The Power of Adaptivity for Stochastic Submodular Cover

Rohan Ghuge [1] **Anupam Gupta** [2] **Viswanath Nagarajan** [1]

Abstract

In the stochastic submodular cover problem, the goal is to select a subset of stochastic items of minimum expected cost to cover a submodular function. Solutions in this setting correspond to sequential decision processes that select items one by one "adaptively" (depending on prior observations). While such adaptive solutions achieve the best objective, the inherently sequential nature makes them undesirable in many applications. We ask: *how well can solutions with only a few adaptive rounds approximate fully-adaptive solutions?* We give nearly tight answers for both independent and correlated settings, proving smooth tradeoffs between the number of adaptive rounds and the solution quality, relative to fully adaptive solutions. Experiments on synthetic and real datasets show qualitative improvements in the solutions as we allow more rounds of adaptivity; in practice, solutions with a few rounds of adaptivity are nearly as good as fully adaptive solutions.

1. Introduction

Submodularity is a fundamental notion that arises in applications such as image segmentation, data summarization (Simon et al., 2007; Lin & Bilmes, 2011; Sipos et al., 2012), hypothesis identification (Barinova et al., 2012; Chen et al., 2014), information gathering (Radanovic et al., 2018), and social networks (Kempe et al., 2015) . The *submodular cover* optimization problem requires us to pick a minimum-cost subset S of items to cover a monotone submodular function f. Submodular cover has been extensively studied in machine learning, computer science and operations research (Wolsey, 1982; Golovin & Krause, 2011; Mirzasoleiman et al., 2015; Bateni et al., 2018): here are two examples from sensor deployment and medical diagnosis.

In the sensor deployment setting, we consider the problem of deploying a collection of sensors to monitor some phenomenon (Krause & Guestrin, 2007; Mini et al., 2014; Sun et al., 2019), for example: we may wish to monitor air quality or traffic situations. The area each sensor can cover depends on its sensing range. The goal of the problem is to deploy as few sensors as possible to cover a desired region entirely. The area covered as a function of the sensors deployed is a submodular function, and we can cast the sensor deployment problem as a special case of submodular cover.

In the medical diagnosis example, we have s possible conditions (hypotheses) the patient may suffer from along with the priors on their occurrence, and our goal is to perform tests to identify the correct condition as quickly as possible (Garey & Graham, 1974; Kosaraju et al., 1999; Dasgupta, 2004; Cicalese et al., 2014). This can be cast as submodular cover by viewing each test as eliminating all inconsistent hypotheses. Hence we want a coverage value of $s-1$: once $s-1$ inconsistent hypotheses are eliminated, the remaining one must be correct.

Observe that both these applications involve uncertain data: the precise area covered by a sensor is not known before the sensor is actually setup and the precise outcome (positive/negative) of a test is not known until the action has been taken. This uncertainty can be modeled using *stochastic submodular optimization*, where the items are stochastic. As a simple example of the stochastic nature, each item may be active or inactive (with known probabilities), and only active items contribute to the submodular function.

A solution for stochastic submodular cover is now a *sequential decision process*. At each step, an item is *probed* and its realization (e.g., active or inactive) is observed. The process is typically *adaptive*, where all the information from previously probed items is used to identify the next item to probe. This process continues until the submodular function is covered, and the goal is to minimize the expected cost of probed items. Such adaptive solutions are inherently fully sequential, which is undesirable if probing an item is time-consuming. E.g., in sensor deployment, probing a sensor corresponds to physically deploying a sensor and observing whether it functions as expected, which may take hours or

[1]Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, USA. Research supported in part by NSF grants CMMI-1940766 and CCF-2006778. [2]Department of Computer Science, Carnegie Mellon University, Pittsburgh, USA. Supported in part by NSF awards CCF-1907820, CCF-1955785, and CCF-2006953.. Correspondence to: Rohan Ghuge <rghuge@umich.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

days. Or, probing/performing a test in medical diagnosis may involve a long wait for test results. Therefore, we prefer solutions with only few *rounds* of adaptivity.

Motivated by this, we ask: *how well do solutions with only a few adaptive rounds approximate fully-adaptive solutions for the stochastic submodular cover problem?* We consider both cases where realizations of different items are independent, and where they are allowed to be correlated. For both these situations we give nearly tight answers, with smooth tradeoffs between the number r of adaptive rounds and the solution quality (relative to fully adaptive solutions).

The main contribution of our work is an r-round adaptive solution for stochastic submodular cover in the "set-based" model for adaptive rounds. In this model, a fixed subset of items is probed in parallel every round (and their total cost is incurred). The decisions in the current round can depend on the realizations seen in all previous rounds. However, as noted in (Agarwal et al., 2019), if we require function f to be covered with probability one then the r-round-adaptivity gap turns out to be very large. Therefore, we focus on set-based solutions that are only required to cover the function with high probability.

In designing algorithms, it turns out to be more convenient to work with the "permutation" model for adaptive rounds, where the function is covered with probability one. This model was also used in prior literature (Goemans & Vondrák, 2006; Agarwal et al., 2019). Here, every round of an r-round-adaptive solution specifies an ordering of all remaining items and probes them in this order until some stopping rule. See Definition 2.1 for a formal definition. Moreover, our r-round adaptive algorithm in the permutation model can be transformed into an r-round adaptive algorithm in the set-based model. We obtain algorithms in the set-based model that:

- for any $\eta \in (0,1)$, finds an r-round adaptive solution that has expected cost at most $\frac{r\alpha}{\eta} \cdot \texttt{OPT}$ and covers the function with probability at least $1-\eta$.
- finds a $2r$-round adaptive solution that has expected cost at most $O(\alpha) \cdot \texttt{OPT}$ and covers the function with probability at least $1 - e^{-\Omega(r)}$.

Here OPT is the cost of an optimal fully-adaptive solution and α is the approximation ratio of our algorithm in the permutation model. The first algorithm above is for the case where r, the number of rounds of adaptivity, is small (say, a constant). In this, we keep the number of rounds the same, but we lose a factor r in the expected cost. The second algorithm is for the case that r is large, e.g., more than a constant. Here, the number of set-based rounds increases by a factor 2, but we only lose a constant factor in expected cost. We formalize and prove these results in the full version of the paper. For the rest of the paper, an r-round adaptive algorithm refers to an an r-round adaptive algorithm in the permutation model (unless specified otherwise).

1.1. Results

Consider a monotone submodular function $f : 2^U \to \mathbb{Z}_{\geq 0}$ with $Q := f(U)$. There are m items, where each item i is a random variable $\mathcal{X}_i$ having cost c_i and corresponding to a random element of U. (Our results extend to the more general setting where each item realizes to a subset of U.) The goal is to select a set of items such that the union S of their corresponding realizations satisfies $f(S) = Q$, and the expected cost is minimized. Our first result is when the items have *independent distributions*.

Theorem 1.1 (Independent Items)**.** *For any integer $r \geq 1$, there is an r-round adaptive algorithm for the stochastic submodular cover problem with cost $O(Q^{1/r} \cdot \log Q)$ times the cost of an optimal adaptive algorithm.*

This improves over an $O(r\, Q^{1/r} \log Q \log(mc_{max}))$ bound from (Agarwal et al., 2019) by eliminating the dependence on the number of items m and the item costs (which could be much larger than Q). Moreover, our result nearly matches the lower bound of $\Omega(\frac{1}{r^3} Q^{1/r})$ from (Agarwal et al., 2019). Setting $r = \log Q$ shows that $O(\log Q)$ adaptive rounds give an $O(\log Q)$-approximation. By transforming this algorithm into a set-based solution, we get:

Corollary 1.2. *There is an $O(\log Q)$ round algorithm for stochastic submodular cover in the set-based model with cost $O(\log Q)$ times the optimal (fully) adaptive cost.*

This approximation ratio of $O(\log Q)$ is the best possible (unless P=NP) even with an arbitrary number ($r = m$) of adaptive rounds. Previously, one could only obtain an $O(\log^2 Q \log(mc_{max}))$-approximation in a logarithmic number of rounds (Agarwal et al., 2019). In the special case of *unit costs*, (Esfandiari et al., 2019) gave an $O(\log(mQ))$-approximation for covering "adaptive submodular" functions using $O(\log m \log(Qm))$ set-based rounds. When costs are arbitrary, their result implies an $O(\log(mQc_{\max}))$-approximation in $O(\log m \log(Qmc_{\max}))$ set-based rounds.

Moreover, Theorem 1.1 (with $r = 1$) implies an $O(Q \log Q)$ adaptivity gap for stochastic set cover (a special case of submodular cover), resolving an open question from (Goemans & Vondrák, 2006) up to an $O(\log Q)$ factor, where Q is the number of objects in set cover.

Our second result is when the items have correlated distributions. Let s denote the support size of the joint distribution $\mathcal{D}$, i.e., the number of *scenarios*.

Theorem 1.3 (Correlated Items)**.** *For any integer $r \geq 1$, there is an r-round adaptive algorithm for scenario sub-*

modular cover with cost $O\left(s^{1/r}(\log s + r \log Q)\right)$ *times the cost of an optimal adaptive algorithm.*

Again, in the set-based setting, we obtain:

Corollary 1.4. *There is an* $O(\log s)$ *round algorithm for scenario submodular cover in the set-based model with cost* $O(\log s \ \log Q)$ *times the optimal (fully) adaptive cost.*

Scenario submodular cover generalizes the classic optimal decision tree problem (Garey & Graham, 1974; Hyafil & Rivest, 1976/77; Loveland, 1985; Kosaraju et al., 1999; Dasgupta, 2004; Adler & Heeringa, 2012; Gupta et al., 2017a). A fully-adaptive $O(\log(sQ))$-approximation for scenario submodular cover was obtained in (Grammel et al., 2016); see also (Navidi et al., 2020) for a more general result. In terms of rounds-of-adaptivity, an $O(\log(mQ\frac{c_{\max}}{p_{\min}}))$-approximation in $O(\log m \ \log(Qm\frac{c_{\max}}{p_{\min}}))$ set-based rounds follows from (Grammel et al., 2016; Esfandiari et al., 2019). Here $p_{\min} \leq \frac{1}{s}$ is the minimum probability of any scenario. Scenario submodular cover is not "adaptive submodular", and so results from (Esfandiari et al., 2019) cannot be used directly. Still, (Grammel et al., 2016) showed an equivalent goal function for scenario submodular cover that is indeed adaptive-submodular, but with a larger "Q value" of $\frac{Q}{p_{min}}$. Then, the algorithm from (Esfandiari et al., 2019) can be applied to this new goal function. We note that when the number of rounds is less than logarithmic, our result provides the first approximation guarantee even in the well-studied special case of optimal decision tree.

The results in Theorem 1.1 and Theorem 1.3 are incomparable: while the independent case has more structure in the distribution $\mathcal{D}$, its support size is exponential. Finally, the dependence on the support size s is necessary in the correlated setting, as our next result shows.

Theorem 1.5 (Lower Bound)**.** *For any integer* $r \geq 1$*, there is an instance of scenario submodular cover with* $Q = 1$ *where the cost of any* r*-round adaptive solution is* $\Omega(\frac{1}{r^2 \log s} \cdot s^{1/r})$ *times the optimal adaptive cost.*

This lower bound is information-theoretic and does not depend on computational assumptions, whereas the upper bound of Theorem 1.3 is given by a polynomial algorithm.

Finally, we note that our algorithms are also easy to implement. We tested both algorithms on synthetic and real datasets that validate the practical performance of our algorithms. Specifically, we test our algorithm for the independent case (Theorem 1.1) on instances of stochastic set cover, and our algorithm for the correlated case (Theorem 1.3) on instances of optimal decision tree. For stochastic set cover, we use real-world datasets to generate instances with $\approx$ 1200 items. We observe a sharp improvement in performance within a few rounds of adaptivity, and that 6-7 rounds of adaptivity are nearly as good as fully adaptive solutions. For optimal decision tree, we use both real-world data and synthetic data. The real-world data has ≈ 400 scenarios and the synthetic data has $10,000$ scenarios. Again, we find that about 6 rounds of adaptivity suffice to obtain solutions as good as fully adaptive ones. We also compared our algorithms' cost to information-theoretic lower bounds for both applications: our costs are typically within 50% of these lower bounds.

1.2. Techniques

The algorithms for the independent and correlated cases are similar, with some crucial differences. In each round of both algorithms, we iteratively compute a "score" for each item and greedily select the item of maximum score. This results in a non-adaptive list of all remaining items, and the items are *probed* in this order until a stopping rule. The PARCA stopping rule in the independent case corresponds to reducing the remaining target (on the function value) by a factor of $Q^{1/r}$, whereas the SPARCA rule involves reducing the number of "compatible scenarios" by an $s^{1/r}$ factor in the correlated case.

The analysis for both Theorems 1.1 and 1.3 follows parallel lines at the beginning. For each $i \geq 0$, we relate the "non-completion" probabilities of the algorithm after cost $\alpha \cdot 2^i$ to the optimal adaptive solution after cost 2^i. The "stretch" factor α corresponds to the approximation ratio, which is different for the independent and correlated cases. In order to relate these non-completion probabilities, we consider the total score G of items selected by the algorithm between cost $\alpha 2^{i-1}$ and $\alpha 2^i$. The crux of the analysis lies in giving lower and upper bounds on the total score G; the arguments here are different for the independent and correlated settings.

In the independent case, the score of any item $\mathcal{X}_e$ is an estimate of its relative marginal gain, where we take an expectation over all previous items as well as $\mathcal{X}_e$. We also normalize this gain by the item's cost. See Equation (1) for the definition. In lower bounding the total score G, we use a variant of a sampling lemma from (Agarwal et al., 2019) as well as the constant-factor adaptivity gap for submodular maximization (Bradac et al., 2019). We also need to partition the outcome space (of all previous realizations) into "good" and bad outcomes: conditional on a good outcome, OPT has a high probability of completing before cost 2^i. Good outcomes are necessary in our proof of the sampling lemma, but luckily the total probability of bad outcomes is small (and they can be effectively ignored). In upper bounding G, we consider the total score as a sum over decision/sample paths and use the fact that the sum of relative gains corresponds to a harmonic series.

In the correlated case, the score of any item $\mathcal{X}_e$ is the sum of two terms (i) its expected relative marginal gain as in the independent case, and (ii) an estimate of the probability on

"eliminated" scenarios. Both terms are needed because the algorithm needs to balance (i) covering the function and (ii) identifying the realized scenario (after which it is trivial to cover f). Again, we normalize by the item's cost. See Equation (2). In lower bounding the total score G, we partition the outcome space into good/okay/bad outcomes that correspond to a high conditional probability of OPT (i) covering function f by cost 2^i, (ii) eliminating a constant fraction of scenarios by cost 2^i, or (iii) neither of the two cases. Further, by restricting to outcomes that have a "large" number of compatible scenarios (else, the algorithm's stopping rule would apply), we can bound the number of "relevant" outcomes by $s^{1/r}$. Then we consider OPT (up to cost 2^i) conditional on all good/okay outcomes and show that one of these items has a high score. To upper bound G, we again consider the total score as a sum over decision paths.

1.3. Other related work

The role of adaptivity has been extensively studied for various stochastic "maximization" problems such as knapsack (Dean et al., 2008; Bhalgat et al., 2011), matching (Bansal et al., 2012; Behnezhad et al., 2020), matroid-constrained probing (Gupta & Nagarajan, 2013) and submodular-maximization (Asadpour & Nazerzadeh, 2016; Gupta et al., 2017b; Bradac et al., 2019). In all these cases, constant-factor adaptivity gaps are known.

Recently, there have been several results examining the role of adaptivity in (deterministic) submodular optimization (Balkanski & Singer, 2018a; Balkanski et al., 2018; Balkanski & Singer, 2018b; Balkanski et al., 2019; Chekuri & Quanrud, 2019). The motivation in these works was parallelizing function queries that are often expensive. In many settings, there are now algorithms using (poly)logarithmic number of rounds that nearly match the best sequential (or fully adaptive) approximation algorithms. While our motivation is similar (in parallelizing the expensive probing steps), the techniques used are completely different.

2. Definitions

In the stochastic submodular cover problem, the input is a collection of m random variables (or *items*) $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_m\}$. Each item $\mathcal{X}_i$ has a cost $c_i \in \mathbb{R}_+$, and realizes to a random element of groundset U. Let the joint distribution of $\mathcal{X}$ be denoted by $\mathcal{D}$. The random variables $\mathcal{X}_i$ may or may not be independent; we discuss this issue in more detail below. The realization of item $\mathcal{X}_i$ is denoted by $X_i \in U$; this realization is only known when $\mathcal{X}_i$ is *probed* at a cost of c_i. Extending this notation, given a subset of items $\mathcal{S} \subseteq \mathcal{X}$, its realization is denoted $S = \{X_i : \mathcal{X}_i \in \mathcal{S}\} \subseteq U$.

In addition, we are given an integer-valued monotone submodular function $f : 2^U \to \mathbb{Z}_+$ with $f(U) = Q$. A realization S of items $\mathcal{S} \subseteq \mathcal{X}$ is *feasible* if and only if $f(S) = Q$ the maximal value; in this case, we also say that $\mathcal{S}$ *covers* f. The goal is to probe (possibly adaptively) a subset $\mathcal{S} \subseteq \mathcal{X}$ of items that gets realized to a feasible set. We use the shorthand $c(\mathcal{S}) := \sum_{i:\mathcal{X}_i \in \mathcal{S}} c_i$ to denote the total cost of items in $\mathcal{S} \subseteq \mathcal{X}$. The objective is to minimize the expected cost of probed items, where the expectation is taken over the randomness in $\mathcal{X}$. We consider the following types of solutions.

Definition 2.1. *For an integer $r \geq 1$, an r-**round-adaptive** solution proceeds in r rounds of adaptivity. In each round $k \in \{1, \ldots, r\}$, the solution specifies an ordering of all remaining items and probes them in this order until some stopping rule. The decisions in round k can depend on the realizations seen in all previous rounds $1, \ldots, k-1$.*

Setting $r = 1$ gives us a *non-adaptive* solution, and setting $r = m$ gives us a *(fully) adaptive* solution. Having more rounds leads to a smaller objective value, so adaptive solutions have the least objective value. Our performance guarantees are relative to an optimal fully adaptive solution; let OPT denote this solution and its cost. The *r-round-adaptivity gap* is defined as follows:

$$\sup_{\text{instance } I} \frac{\mathbb{E}[\text{cost of best } r\text{-round adaptive solution on } I]}{\mathbb{E}[\text{cost of best adaptive solution on } I]}$$

Setting $r = 1$ gives the *adaptivity gap*.

Independent and Correlated Distributions We first study the case where the random variables $\mathcal{X}$ are *independent*. In keeping with existing literature (Im et al., 2016; Deshpande et al., 2016; Agarwal et al., 2019), we refer to this problem simply as the *stochastic submodular cover* problem. We then consider the case when the random variables $\mathcal{X}$ are correlated with a joint distribution of polynomial size, and refer to it as the *scenario submodular cover* problem (Grammel et al., 2016).

3. Stochastic Submodular Cover

We now consider the (independent) stochastic submodular cover problem and prove Theorem 1.1. For simplicity, we assume that costs c_i are integers. Our results also hold for arbitrary costs (by replacing certain summations in the analysis by integrals).

We find it convenient to solve a *partial cover* version of the stochastic submodular cover problem. In this partial cover version, we are given a parameter $\delta \in [0, 1]$ and the goal is to probe some $\mathcal{R}$ that realizes to a set R achieving value $f(R) > Q(1 - \delta)$. We are interested in a *non adaptive* algorithm for this problem. Clearly, if $\delta = 1/Q$, the integrality of the function f means that $f(R) = Q$, and we solve the original (full coverage) problem. Moreover, we can use this

algorithm with different thresholds to also get the r-round version of the problem. The main result of this section is:

Theorem 3.1. *There is a non-adaptive algorithm for the partial cover version of stochastic submodular cover with cost $O\left(\frac{\ln 1/\delta}{\delta}\right)$ times the optimal adaptive cost for the (full) submodular cover.*

The algorithm first creates an ordering/list L of the items non-adaptively; that is, without knowing the realizations of the items. To do so, at each step we pick a new item that maximizes a carefully-defined score function (Equation (1)). The score of an item cannot depend on the realizations of previous items on the list (since we are non-adaptive). However, it depends on the *marginal relative increase* for a random draw from the previously chosen items. Once this ordering L is specified, the algorithm starts probing and realizing the items in this order, and does so until the realized value exceeds $(1-\delta)Q$.

Algorithm 1 PARtial Covering Algorithm PARCA$(\mathcal{X}, f, Q, \delta)$

$\mathcal{S} \leftarrow \emptyset$, list $L \leftarrow \langle\rangle$, $\tau \leftarrow Q(1-\delta)$
while $\mathcal{S} \neq \mathcal{X}$ **do** ▷ Building the list non-adaptively
  select an item $\mathcal{X}_e \in \mathcal{X} \setminus \mathcal{S}$ that maximizes:

$$\text{score}(\mathcal{X}_e) := \frac{1}{c_e} \cdot \sum_{S \sim \mathcal{S}: f(S) \leq \tau} \mathbf{P}(\mathcal{S} = S) \cdot \mathbb{E}_{X_e \sim \mathcal{X}_e}\left[\frac{f(S \cup X_e) - f(S)}{Q - f(S)}\right] \quad (1)$$

  $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathcal{X}_e\}$ and list $L \leftarrow L \circ \mathcal{X}_e$
$\mathcal{R} \leftarrow \emptyset$, $R \leftarrow \emptyset$
while $f(R) \leq \tau$ **do** ▷ Probing items on the list
  $\mathcal{X}_e \leftarrow$ first r.v. in list L not in $\mathcal{R}$
  $X_e \in U$ be the realization of $\mathcal{X}_e$
  $R \leftarrow R \cup \{X_e\}, \mathcal{R} \leftarrow \mathcal{R} \cup \{\mathcal{X}_e\}$
return probed items $\mathcal{R}$ and their realizations R.

Given this partial covering algorithm we immediately get an algorithm for the r-round version of the problem, where we are allowed to make r rounds of adaptive decisions. Indeed, we can first set $\delta = Q^{-1/r}$ and solve the partial covering problem with this value of δ. Suppose we probe variables $\mathcal{R}$ and their realizations are given by the set $R \subseteq U$. Then we can condition on these values to get the marginal value function f_R (which is submodular), and inductively get an $r-1$-round solution for this problem. The following algorithm formalizes this.

Theorem 3.2. *Algorithm 2 is an r-round adaptive algorithm for stochastic submodular cover with cost $O(Q^{1/r} \log Q)$ times the optimal adaptive cost.*

The proofs of Theorem 3.1 and Theorem 3.2 can be found in the full version of the paper.

Algorithm 2 r-round adaptive algorithm for stochastic submodular cover SSC$(r, \mathcal{X}, f)$

run PARCA $(\mathcal{X}, f, Q, Q^{-1/r})$ for round #1.
let $\mathcal{R}$ (resp. R) denote the probed items (resp. their realizations) in PARCA.
define residual submodular function $\widehat{f} := f_R$.
recursively solve SSC$(r-1, \mathcal{X} \setminus \mathcal{R}, \widehat{f})$.

Remark: Assuming that the scores (1) can be computed in polynomial time, it is clear that our entire algorithm can be implemented in polynomial time. In particular, if T denotes the time taken to calculate the score of one item, then the overall algorithm runs in time $poly(m, T)$ where m is the number of items. We are not aware of a closed-form expression for the scores (for arbitrary submodular functions f). However, as discussed in the full version, we can use sampling to estimate these scores to within a constant factor. Moreover, our analysis works even if we only choose an approximate maximizer for (1) at each step. It turns out that $T = poly(m, c_{max})$ many samples suffice to estimate these scores. So, the final runtime is $poly(m, c_{max})$; note that this does not depend on the number $|U|$ of elements. We note that even the previous algorithms (Agarwal et al., 2019; Esfandiari et al., 2019) have a polynomial dependence on c_{max} in their runtime. In the following analysis we will assume that the scores (1) are computed exactly.

3.1. Analysis for a Call to PARCA

We now prove Theorem 3.1. We denote by OPT an optimal adaptive solution for the covering problem on f. Now we analyze the cost incurred by following the non-adaptive strategy (which we call NA): probe variables according to the order given by the list L generated by PARCA, and stop when the realized coverage exceeds $\tau := Q(1-\delta)$ (see Algorithm 1 for details). We consider the expected cost of this strategy, and relate it to the cost of OPT.

We refer to the cumulative cost incurred (either by OPT or by NA) until any point in a solution as *time* elapsing. We say that OPT is in phase i when it is in the time interval $[2^i, 2^{i+1})$ for $i \geq 0$. Let $\alpha := O\left(\frac{\ln 1/\delta}{\delta}\right)$. We say that NA is in *phase* i when it is in time interval $[\alpha \cdot 2^{i-1}, \alpha \cdot 2^i)$ for $i \geq 1$; phase 0 refers to the interval $[1, \alpha)$. Define

- u_i: probability that NA goes beyond phase i; that is, has cost at least $\alpha \cdot 2^i$.
- u_i^*: probability that OPT goes beyond phase $i-1$; that is, costs at least 2^i.

Since all costs are integers, $u_0^* = 1$. For ease of notation, we also use OPT and NA to denote the *random* cost incurred

by OPT and NA respectively. The following lemma forms the crux of the analysis.

Lemma 3.3. *For any phase $i \geq 1$, $u_i \leq \frac{u_{i-1}}{4} + \frac{6}{5}u_i^*$.*

This lemma implies Theorem 3.1. We include the proofs of Theorem 3.1 and Lemma 3.3 in the full version of the paper.

4. Scenario Submodular Cover

In this section, we describe an r-round adaptive algorithm for the *scenario submodular cover* problem. In contrast to the independent case, the stochastic items here are correlated, and their joint distribution $\mathcal{D}$ is given as input.

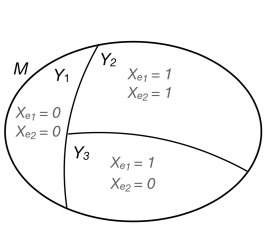
The joint distribution $\mathcal{D}$ specifies the (joint) probability that $\mathcal{X}$ realizes to any outcome $X \in U^m$. We refer to the realizations $X \in U^m$ that have a non-zero probability of occurrence as *scenarios*. Let $s = |\mathcal{D}|$ denote the number of scenarios in $\mathcal{D}$. The set of scenarios is denoted $M = \{1, \cdots, s\}$ and p_ω denotes the probability of each scenario $\omega \in M$. Note that $\sum_{\omega=1}^{s} p_\omega = 1$. For each scenario $\omega \in M$ and item $\mathcal{X}_e$, we denote by $X_e(\omega) \in U$ the realization of $\mathcal{X}_e$ in scenario ω. The distribution $\mathcal{D}$ can be viewed as selecting a random *realized scenario* $\omega^* \in M$ according to the probabilities $\{p_\omega\}$, after which the item realizations are deterministically set to $\langle X_1(\omega^*), \cdots, X_m(\omega^*)\rangle$. However, an algorithm does not know the realized scenario ω^*: it only knows the realizations of the probed items (using which it can infer a posterior distribution for ω^*). As stated in §2, our performance guarantee in this case depends on the support-size s. We will also show that such a dependence is necessary (even when Q is small).

For any subset $\mathcal{S} \subseteq \mathcal{X}$ of items, we denote by $S(\omega)$, the realizations for items in $\mathcal{S}$ under scenario ω. We say that scenario ω is *compatible* with a realization of $\mathcal{S} \subseteq \mathcal{X}$ if and only if, $\mathcal{X}_e$ realizes to $X_e(\omega)$ for all items $\mathcal{X}_e \in \mathcal{S}$.

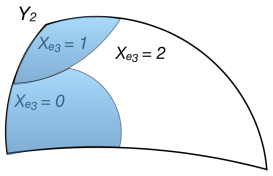
4.1. The Algorithm

Similar to the algorithm for the independent case, it is convenient to solve a *partial cover* version of the scenario submodular cover problem. However, the notion of partial progress is different: we will use the *number of compatible scenarios* instead of function value. Formally, in the partial version, we are given a parameter $\delta \in [0, 1]$ and the goal is to probe some items $\mathcal{R}$ that realize to a set R such that either (i) the number of compatible scenarios is less than δs or (ii) the function f is fully covered. Clearly, if $\delta = 1/s$ then case (i) cannot happen (it corresponds to zero compatible scenarios), so the function f must be fully covered. We will use this algorithm with different parameters δ to solve the r-round version of the problem. The main result of this section is:

Theorem 4.1. *There is a non-adaptive algorithm for the partial cover version of scenario submodular cover with cost $O\left(\frac{1}{\delta}(\ln\frac{1}{\delta} + \log Q)\right)$ times the optimal adaptive cost for the (full) submodular cover.*



(a) We have $\mathcal{S} = \{\mathcal{X}_{e_1}, \mathcal{X}_{e_2}\}$, and we partition the set of scenarios M based on outcomes of $\mathcal{S}$ to get $\mathcal{H}(S) = \{Y_1, Y_2, Y_3\}$.



(b) We further partition scenarios Y_2 based on realizations of $\mathcal{X}_{e_3}$.The part of Y_2 compatible with outcome $X_{e_3} = 2$ is the largest cardinality part, that is, $B_{e_3}(Y_2)$. The shaded region represents $L_{e_3}(Y_2)$.

Figure 1. Illustrations of Key Definitions

The algorithm first creates an ordering/list L of the items non-adaptively; that is, without knowing the realizations of the items. To do so, at each step we pick a new item that maximizes a carefully-defined score function (Equation (2)). The score of an item depends on an estimate of progress towards (i) eliminating scenarios and (ii) covering function f. Before we state this score formally, we need some definitions.

Definition 4.1. *For any $\mathcal{S} \subseteq \mathcal{X}$ let $\mathcal{H}(\mathcal{S})$ denote the partition $\{Y_1, \cdots, Y_\ell\}$ of the scenarios M where all scenarios in a part have the same realization for items in $\mathcal{S}$. Let $\mathcal{Z} := \{Y \in \mathcal{H}(\mathcal{S}) : |Y| \geq \delta s\}$ be the set of "large" parts having size at least δs.*

In other words, scenarios ω and σ lie in the same part of $\mathcal{H}(\mathcal{S})$ if and only if $S(\omega) = S(\sigma)$. Note that partition $\mathcal{H}(\mathcal{S})$ does not depend on the realization of $\mathcal{S}$. Moreover, after probing and realizing items $\mathcal{S}$, the set of compatible scenarios must be one of the parts in $\mathcal{H}(\mathcal{S})$. Also, the number of "large" parts $|\mathcal{Z}| \leq \frac{s}{\delta s} = \frac{1}{\delta}$ as the number of scenarios $|M| = s$. See Figure 1a for an example.

Definition 4.2. *For any $\mathcal{X}_e \in \mathcal{X}$ and subset $Z \subseteq M$ of scenarios, consider the partition of Z based on the realization of $\mathcal{X}_e$. Let $B_e(Z) \subseteq Z$ be the largest cardinality part, and define $L_e(Z) := Z \setminus B_e(Z)$.*

Note that $L_e(Z)$ is comprised of several parts of the above partition of Z. If the realized scenario $\omega^* \in L_e(Z)$ and $\mathcal{X}_e$ is selected then, at least half the scenarios in Z will be eliminated (as being incompatible with X_e). Figure 1b illustrates these definitions.

For any $Z \in \mathcal{H}(\mathcal{S})$, note that the realizations $S(\omega)$ are identical for all $\omega \in Z$: we use $S(Z) \subseteq U$ to denote the realization of $\mathcal{S}$ under each scenario in Z.

If $\mathcal{S}$ denotes the previously added items in list L, the score

(2) involves a term for each part $Z \in \mathcal{Z}$, which itself comes from two sources:

- *Information gain* $\sum_{\omega \in L_e(Z)} p_\omega$, the total probability of scenarios in $L_e(Z)$.
- *Relative function gain*

$$\sum_{\omega \in Z} p_\omega \cdot \frac{f(S(Z) \cup X_e(\omega)) - f(S(Z))}{Q - f(S(Z))},$$

the expected relative gain obtained by including element X_e, where the expectation is over scenarios Z.

The overall score of item $\mathcal{X}_e$ is the sum of these terms (over all parts in $\mathcal{Z}$) normalized by the cost c_e of item $\mathcal{X}_e$. In defining the score, we only focus on the "large" parts $\mathcal{Z}$. If the realization of $\mathcal{S}$ corresponds to any other part then the number of compatible scenarios would be less than δs (and the partial-cover algorithm would have terminated).

Once the list L is specified, the algorithm starts probing and realizing the items in this order, and does so until either (i) the number of compatible scenarios drops below δs, or (ii) the realized function value equals Q. Note that in case (ii), the function is fully covered. See Algorithm 3 for a formal description of the non-adaptive partial-cover algorithm.

Algorithm 3 Scenario PARtial Covering Algorithm SPARCA$(\mathcal{X}, M, f, Q, \delta)$

$\mathcal{S} \leftarrow \emptyset$ and list $L \leftarrow \langle\rangle$.
while $\mathcal{S} \neq \mathcal{X}$ **do** ▷ Building the list non-adaptively
  define $\mathcal{Z}$ and $L_e(Z)$ as in Definitions 4.1 and 4.2.
  select an item $\mathcal{X}_e \in \mathcal{X} \setminus \mathcal{S}$ that maximizes:

$$\text{score}(\mathcal{X}_e) = \frac{1}{c_e} \cdot \sum_{Z \in \mathcal{Z}} \left(\sum_{\omega \in L_e(Z)} p_\omega + \sum_{\omega \in Z} p_\omega \cdot \frac{f(S(Z) \cup X_e(\omega)) - f(S(Z))}{Q - f(S(Z))} \right) \quad (2)$$

  $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathcal{X}_e\}$ and list $L \leftarrow L \circ \mathcal{X}_e$
$\mathcal{R} \leftarrow \emptyset$, $R \leftarrow \emptyset$, $H \leftarrow M$.
while $|H| \geq \delta |M|$ and $f(R) < Q$ **do**
  $\mathcal{X}_e \leftarrow$ first r.v. in list L not in $\mathcal{R}$
  $X_e = v \in U$ be the realization of $\mathcal{X}_e$.
  $R \leftarrow R \cup \{v\}, \mathcal{R} \leftarrow \mathcal{R} \cup \{\mathcal{X}_e\}$
  $H \leftarrow \{\omega \in H : X_e(\omega) = v\}$
return probed items $\mathcal{R}$, realizations R and compatible scenarios H.

Given this partial covering algorithm we immediately get an algorithm for the r-round version of the problem, where we are allowed to make r rounds of adaptive decisions. Indeed, we can first set $\delta = s^{-1/r}$ and solve the partial covering problem. Suppose we probe the items $\mathcal{R}$ (with realizations $R \subseteq U$) and are left with compatible scenarios $H \subseteq M$. Then we can *condition* on scenarios H and the marginal value function f_R (which is submodular), and inductively get an $r-1$-round solution for this problem. The following algorithm and result formalizes this.

Algorithm 4 r-round adaptive algorithm for scenario submodular cover NSC$(r, \mathcal{X}, M, f)$

run SPARCA$(\mathcal{X}, M, f, Q, |M|^{-1/r})$ for round one. Let $\mathcal{R}$ denote the probed items, R their realizations, and $H \subseteq M$ the compatible scenarios returned by SPARCA.
define residual submodular function $\widehat{f} := f_R$.
recursively solve NSC$(r-1, \mathcal{X} \setminus \mathcal{R}, H, \widehat{f})$.

Theorem 4.2. *Algorithm 4 is an r-round adaptive algorithm for scenario submodular cover with cost $O\left(s^{1/r}(\log s + r \log Q)\right)$ times the optimal adaptive cost, where s is the number of scenarios.*

We defer the proofs of Theorems 4.1 and 4.2 to the full version of the paper.

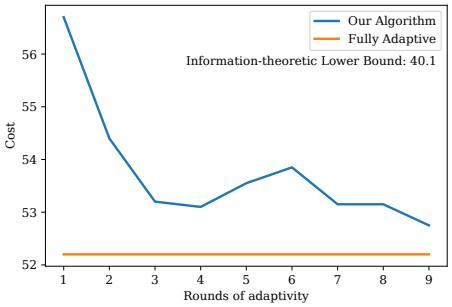
5. Computational Results

We provide a summary of computational results of our r-round adaptive algorithms for the stochastic set cover and optimal decision tree problems. We conducted all of our computational experiments using Python 3.8 and Gurobi 8.1 with a 2.3 Ghz Intel Core $i5$ processor and 16 GB 2133 MHz LPDDR3 memory.

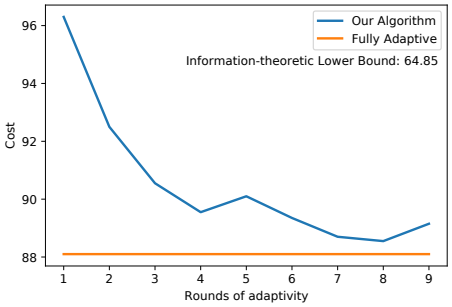
5.1. Stochastic Set Cover

Instances. We use the Epinions network[1] and a Facebook messages dataset described in (Rossi & Ahmed, 2015) to generate instances of the stochastic set cover problem. The Epinions network consists of 75 879 nodes and 508 837 directed edges. We compute the subgraph induced by the top 1000 nodes with the highest out-degree (this subgraph has 57 092 directed edges) and use this to generate the stochastic set cover instance. The Facebook messages dataset consists 1 266 nodes and 6 451 directed edges. Given an underlying directed graph, we generate an instance of the stochastic set cover problem as follows. We associate the ground set U with the set of nodes of the underlying graph. We associate an item with each node. Let $\Gamma(u)$ denote the out-neighbors of u. We sample a subset of $\Gamma(u)$ using the binomial distribution with $p = 0.1$ for 500 times. Let $S \subseteq \Gamma(u)$ be sampled α_S times: then $\mathcal{X}_u$ realizes to $S \cup \{u\}$ with

[1] http://snap.stanford.edu/



(a) Epinions network



(b) Facebook messages network

Figure 2. Computational results for Stochastic Set Cover

probability $\alpha_S/500$. This ensures that $\mathcal{X}_u$ always covers u. We set f to be the coverage function and set $Q = \delta n$ where n represents the number of nodes in the underlying network. We use $\delta = 0.5$ for the Epinions network. However, since the Facebook messages network is sparse, we set $\delta = 0.25$ in the second instance.

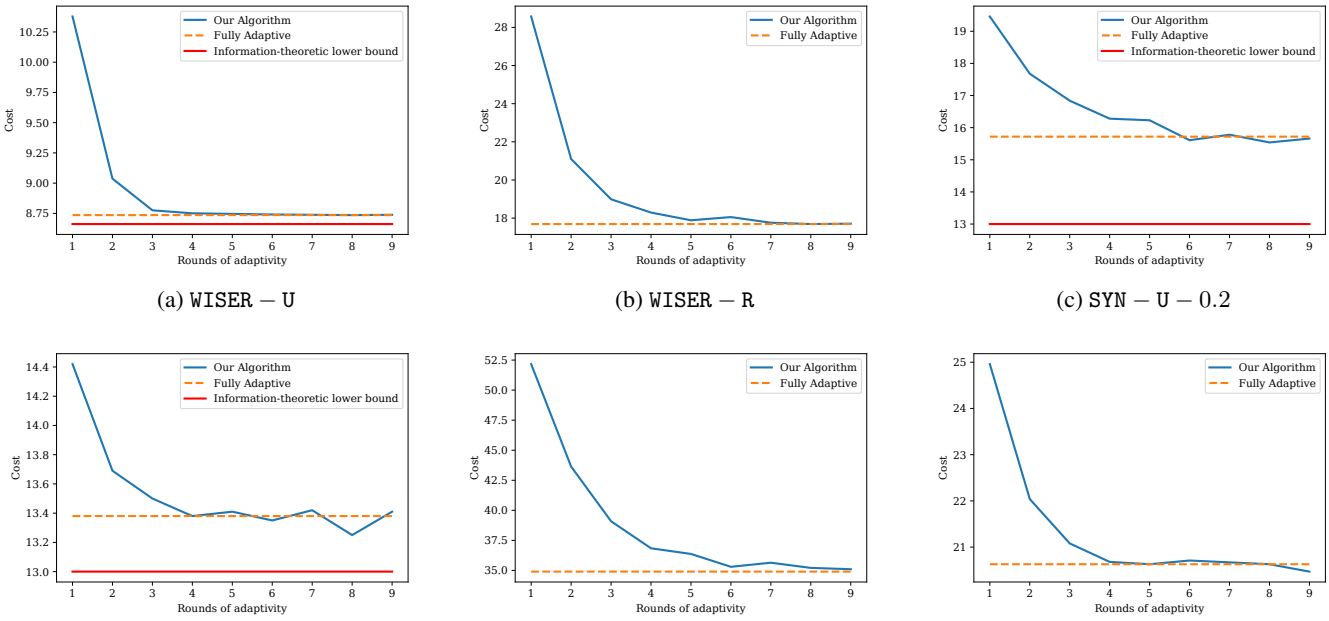
Results. We test our r-round adaptive algorithm on the two kinds of instances described above. We vary r over integers in the interval $[1, \log n]$, where $n \approx 1000$. To compute an estimate of the expected cost, we sample new realizations 20 times and take the average cost incurred by the algorithm over these trials. In each trial, we also solve an integer linear program (using the Gurobi solver) to determine the optimal *offline* cost to cover Q elements for the given realization: we use the average cost over the trials as an estimate on an information-theoretic lower bound: no adaptive policy can do better than this lower bound. In fact, the gap between this information-theoretic lower bound and an optimal adaptive solution may be as large as $\Omega(Q)$ on worst-case instances. We observe that in both cases, the expected cost of our solution after only a few rounds of adaptivity is within 50% of the information-theoretic lower bound. Moreover, in $6-7$ rounds of adaptivity we notice a decrease of $\approx 8\%$ in the expected cost and these solutions are nearly as good as fully adaptive solutions. We plot this trend in Figure 2. Finally, note that the increase in expected cost with rounds of adaptivity (see Figure 2a) can be attributed to the the probabilistic nature of our algorithm (and the experimental setup). We also notice this in the next batch of experiments.

5.2. Optimal Decision Tree

Instances. We use a real-world dataset called WISER [2] for our experiments. The WISER dataset describes symptoms that one may suffer from after being exposed to certain chemicals. It contains data corresponding to 415 chemicals (scenarios for ODT) and 79 symptoms (elements with binary outcomes). Given a patient exhibiting certain symptoms, the goal is to identify the chemical that the patient has been exposed to (by testing as few symptoms as possible). This dataset has been used for evaluating algorithms for similar problems in other papers (Bhavnani et al., 2007; Bellala et al., 2011; Bellala et al., 2012; Navidi et al., 2020). For each symptom-chemical pair, the data specifies whether or not someone exposed to the chemical exhibits the given symptom. However, the WISER data has 'unknown' entries for some pairs. In order to obtain instances for ODT from this, we generate 10 different datasets by assigning random binary values to the 'unknown' entries. Then we remove all identical scenarios: to ensure that the ODT instance is feasible. We use the uniform probability distribution for the scenarios. Given these 10 datasets, we consider two cases. The first case considered is one where all tests have unit cost. We refer to this as the WISER $-$ U case. For the second case, we generate costs randomly for each test from $\{1, 4, 7, 10\}$ according to the weight vector $[0.1, 0.2, 0.4, 0.3]$; for example, with probability 0.4, a test is assigned cost 7. Varying cost captures the setting where tests may have different costs, and we may not want to schedule an expensive test without sufficient information. We refer to this as WISER$-$R case.

We also test our algorithm on synthetic data which we generate as follows. We set $s = 10000$ and $m = 100$. For each $y \in [s]$, we randomly generate a binary sequence of outcomes which corresponds to how y reacts to the tests. We do this in two ways: for test e, we set $y \in T_e$ with probability $p \in \{0.2, 0.5\}$. If a sequence of outcomes is repeated, we discard the scenario to ensure feasibility of the ODT instance. We assign equal probability to each scenario. We generate instances using unit costs or by assigning a random cost from $\{1, 4, 7, 10\}$ to each test according to the weight vector $[0.1, 0.2, 0.4, 0.3]$ (as described above). Thus, we generate 4 types of instances with synthetic data. We refer to the instance generated with $p = 0.2$ and unit costs as SYN $-$ U $- 0.2$. We similarly name the other instances.

[2] http://wiser.nlm.nih.gov/



(a) WISER − U (b) WISER − R (c) SYN − U − 0.2

(d) SYN − U − 0.5 (e) SYN − R − 0.2 (f) SYN − R − 0.5

Figure 3. Performance of our algorithm for the Optimal Decision Tree problem on the WISER and synthetic datasets.

Results. We test our r-round adaptive algorithm on all of the above mentioned datasets. We vary r over integers in the interval $[1, \log s]$. For the WISER − U datasets, we compute the expected cost of our algorithm over *all* scenarios. We plot the expected costs over all rounds, and compare it to $\log(s)$ which is an information-theoretic lower bound for the ODT problem with unit costs and uniform probabilities over the scenarios. We observe that our algorithm gets very close to this lower bound in only 3 rounds of adaptivity. See Figure 3a. In the case of WISER − R, we compute the expected cost incurred by our r-round adaptive algorithm for r varying over integers in the interval $[1, \log s]$ (expectation taken over *all* scenarios). We observe a sharp decrease in costs within 4 rounds of adaptivity. We plot the results in Figure 3b.

For the synthetic data, we sample scenarios over 100 trials (since $s = 10000$, computing expectation over all s would be very slow). As in the previous case, for each trial, we compute the cost incurred by our r-round adaptive algorithm for r varying over integers in the interval $[1, \log s]$. Then, we average all these costs and use it as an estimate for the expected cost. For SYN − U − 0.2 and SYN − U − 0.5, we compare the results to $\log(s)$ which is an information-theoretic lower bound for the instances (since scenarios are sampled uniformly at random). We observe a improvement in the costs within 6 rounds of adaptivity. Beyond this however, the costs do not improve (and we exclude it from our plot). We observe a similar trend for the case of SYN − R − 0.2 and SYN − R − 0.5. We plot the results in Figures 3c-3f.

Note that we only plot results related to the first dataset for the WISER − U and WISER − R cases. We include plots for all 10 datasets in the full version of the paper.

References

Adler, M. and Heeringa, B. Approximating optimal binary decision trees. *Algorithmica*, 62(3-4):1112–1121, 2012.

Agarwal, A., Assadi, S., and Khanna, S. Stochastic submodular cover with limited adaptivity. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 323–342, 2019.

Asadpour, A. and Nazerzadeh, H. Maximizing stochastic monotone submodular functions. *Manag. Sci.*, 62(8): 2374–2391, 2016.

Balkanski, E. and Singer, Y. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1138–1151, 2018a.

Balkanski, E. and Singer, Y. Approximation guarantees for adaptive sampling. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 393–402, 2018b.

Balkanski, E., Breuer, A., and Singer, Y. Non-monotone submodular maximization in exponentially fewer iterations. In *Advances in Neural Information Processing Systems*, pp. 2359–2370, 2018.

Balkanski, E., Rubinstein, A., and Singer, Y. An exponential speedup in parallel running time for submodular maximization without loss in approximation. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 283–302, 2019.

Bansal, N., Gupta, A., Li, J., Mestre, J., Nagarajan, V., and Rudra, A. When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

Barinova, O., Lempitsky, V., and Kholi, P. On detection of multiple object instances using hough transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1773–1784, 2012.

Bateni, M., Esfandiari, H., and Mirrokni, V. S. Optimal distributed submodular optimization via sketching. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1138–1147, 2018.

Behnezhad, S., Derakhshan, M., and Hajiaghayi, M. Stochastic matching with few queries: (1-ϵ) approximation. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1111–1124, 2020.

Bellala, G., Bhavnani, S., and Scott, C. Active diagnosis under persistent noise with unknown noise distribution: A rank-based approach. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15, pp. 155–163, 2011.

Bellala, G., Bhavnani, S. K., and Scott, C. Group-based active query selection for rapid diagnosis in time-critical situations. *IEEE Transactions on Information Theory*, 58 (1):459–478, 2012.

Bhalgat, A., Goel, A., and Khanna, S. Improved approximation results for stochastic knapsack problems. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1647–1665, 2011.

Bhavnani, S. K., Abraham, A., Demeniuk, C., Gebrekristos, M., Gong, A., Nainwal, S., Vallabha, G. K., and Richardson, R. J. Network analysis of toxic chemicals and symptoms: implications for designing first-responder systems. *AMIA Annual Symposium Proceedings*, pp. 51–55, 2007.

Bradac, D., Singla, S., and Zuzic, G. (Near) Optimal Adaptivity Gaps for Stochastic Multi-Value Probing. In *Approximation, Randomization, and Combinatorial Optimization*, volume 145, pp. 49:1–49:21, 2019.

Chekuri, C. and Quanrud, K. Parallelizing greedy for submodular set function maximization in matroids and beyond. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 78–89, 2019.

Chen, Y., Shio, H., Montesinos, C. A. F., Koh, L. P., Wich, S., and Krause, A. Active detection via adaptive submodularity. In *Proceedings of the 31st International Conference on Machine Learning*, pp. I–55–I–63, 2014.

Cicalese, F., Laber, E. S., and Saettler, A. M. Diagnosis determination: decision trees optimizing simultaneously worst and expected testing cost. In *Proceedings of the 31th International Conference on Machine Learning*, pp. 414–422, 2014.

Dasgupta, S. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems*, pp. 337–344, 2004.

Dean, B. C., Goemans, M. X., and Vondrák, J. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.*, 33(4):945–964, 2008.

Deshpande, A., Hellerstein, L., and Kletenik, D. Approximation algorithms for stochastic submodular set cover with applications to boolean function evaluation and min-knapsack. *ACM Trans. Algorithms*, 12(3), April 2016.

Esfandiari, H., Karbasi, A., and Mirrokni, V. S. Adaptivity in adaptive submodularity. *CoRR*, abs/1911.03620, 2019.

Garey, M. and Graham, R. Performance bounds on the splitting algorithm for binary testing. *Acta Informatica*, 3:347–355, 1974.

Goemans, M. and Vondrák, J. Stochastic covering and adaptivity. In *LATIN 2006: Theoretical Informatics*, pp. 532–543. Springer Berlin Heidelberg, 2006.

Golovin, D. and Krause, A. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res. (JAIR)*, 42:427–486, 2011.

Grammel, N., Hellerstein, L., Kletenik, D., and Lin, P. Scenario submodular cover. In *International Workshop on Approximation and Online Algorithms*, pp. 116–128. Springer, 2016.

Gupta, A. and Nagarajan, V. A stochastic probing problem with applications. In *Integer Programming and Combinatorial Optimization - 16th International Conference*, pp. 205–216, 2013.

Gupta, A., Nagarajan, V., and Ravi, R. Approximation algorithms for optimal decision trees and adaptive TSP problems. *Math. Oper. Res.*, 42(3):876–896, 2017a.

Gupta, A., Nagarajan, V., and Singla, S. Adaptivity gaps for stochastic probing: Submodular and XOS functions. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1688–1702, 2017b.

Hyafil, L. and Rivest, R. L. Constructing optimal binary decision trees is NP-complete. *Information Processing Lett.*, 5(1):15–17, 1976/77.

Im, S., Nagarajan, V., and van der Zwaan, R. Minimum latency submodular cover. *ACM Trans. Algorithms*, 13 (1):13:1–13:28, 2016.

Kempe, D., Kleinberg, J. M., and Tardos, É. Maximizing the spread of influence through a social network. *Theory of Computing*, 11:105–147, 2015.

Kosaraju, S. R., Przytycka, T. M., and Borgstrom, R. S. On an Optimal Split Tree Problem. In *Proceedings of the 6th International Workshop on Algorithms and Data Structures*, pp. 157–168, 1999.

Krause, A. and Guestrin, C. Near-optimal observation selection using submodular functions. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 2*, pp. 1650–1654. AAAI Press, 2007.

Lin, H. and Bilmes, J. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pp. 510–520, 2011.

Loveland, D. W. Performance bounds for binary testing with arbitrary weights. *Acta Inform.*, 22(1):101–114, 1985.

Mini, S., Udgata, S. K., and Sabat, S. L. Sensor deployment and scheduling for target coverage problem in wireless sensor networks. *IEEE Sensors Journal*, 14(3):636–644, 2014.

Mirzasoleiman, B., Karbasi, A., Badanidiyuru, A., and Krause, A. Distributed submodular cover: Succinctly summarizing massive data. In *Advances in Neural Information Processing Systems*, pp. 2881–2889, 2015.

Navidi, F., Kambadur, P., and Nagarajan, V. Adaptive submodular ranking and routing. *Oper. Res.*, 68(3):856–877, 2020.

Radanovic, G., Singla, A., Krause, A., and Faltings, B. Information gathering with peers: Submodular optimization with peer-prediction constraints. In *Proc. Conference on Artificial Intelligence (AAAI)*, Feburary 2018.

Rossi, R. A. and Ahmed, N. K. The network data repository with interactive graph analytics and visualization. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015.

Simon, I., Snavely, N., and Seitz, S. M. Scene summarization for online image collections. In *11th International Conference on Computer Vision*, pp. 1–8, 2007.

Sipos, R., Swaminathan, A., Shivaswamy, P., and Joachims, T. Temporal corpus summarization using submodular word coverage. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pp. 754–763, 2012.

Sun, C., Li, V. O. K., Lam, J. C. K., and Leslie, I. Optimal citizen-centric sensor placement for air quality monitoring: A case study of city of cambridge, the united kingdom. *IEEE Access*, 7:47390–47400, 2019.

Wolsey, L. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4): 385–393, 1982.