Robust and MaxMin Optimization under Matroid and Knapsack Uncertainty Sets

ANUPAM GUPTA, Carnegie Mellon University
and VISWANATH NAGARAJAN, University of Michigan
and R. RAVI, Carnegie Mellon University

Consider the following problem: given a set system (U, Ω) and an edge-weighted graph $G = (U, E)$ on the same universe U, find the set $A \in \Omega$ such that the Steiner tree cost with terminals A is as large as possible—"which set in Ω is the most difficult to connect up?" This is an example of a *max-min problem*: find the set $A \in \Omega$ such that the value of some minimization (covering) problem is as large as possible.

In this paper, we show that for certain covering problems which admit good deterministic online algorithms, we can give good algorithms for max-min optimization when the set system Ω is given by a p-system or knapsack constraints or both. This result is similar to results for constrained maximization of submodular functions. Although many natural covering problems are not even approximately submodular, we show that one can use properties of the online algorithm as a surrogate for submodularity.

Moreover, we give stronger connections between max-min optimization and two-stage robust optimization, and hence give improved algorithms for robust versions of various covering problems, for cases where the uncertainty sets are given by p-systems and knapsack constraints.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Robust Optimization, Submodularity, Approximation Algorithms, Online Algorithms

1. INTRODUCTION

Recent years have seen a considerable body of work on the problem of constrained submodular maximization: you are given a universe U of elements, a collection $\Omega \subseteq 2^U$ of "independent" sets and a submodular function $f : 2^U \to \mathbb{R}_{\geq 0}$, and the goal is to solve the optimization problem of maximizing f over the "independent" sets:

$$\max_{S \in \Omega} f(S). \qquad \text{(Max-}f\text{)}$$

An extended abstract containing the results of this paper and of [Gupta et al. 2014] appeared jointly in *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP), 2010*.

A. Gupta's research supported in part by NSF awards CCF-0448095 and CCF-0729022, and an Alfred P. Sloan Fellowship. R. Ravi's research supported in part by NSF grants CCF-0728841 and CCF-1218382.

Author's addresses: A. Gupta, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, email: anupamg@cs.cmu.edu; V. Nagarajan, Industrial and Operations Engineering Department, University of Michigan, Ann Arbor, MI 48105, email: viswa@umich.edu; R. Ravi, Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213, email: ravi@cmu.edu.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

It is a classical result that when f is a linear function and (U, Ω) is a matroid, the greedy algorithm solves this exactly. Furthermore, results from the mid-1970s [Nemhauser et al. 1978; Fisher et al. 1978] tell us that when f is monotone submodular and (U, Ω) is a partition matroid, the problem becomes NP-hard, but the greedy algorithm is an $\frac{e}{e-1}$-approximation—in fact, greedy is a 2-approximation for monotone submodular maximization subject to *any* matroid constraint. Recent results have shed more light on this problem: it is now known that when f is a monotone submodular function and (U, Ω) is any matroid, there exists an $\frac{e}{e-1}$-approximation algorithm [Călinescu et al. 2011]. We can remove the constraint of monotonicity, and also generalize the constraint Ω substantially: the most general results [Călinescu et al. 2011; Kulik et al. 2013; Lee et al. 2010a; Lee et al. 2010b; Gupta et al. 2010] say that if f is a non-negative submodular function, and if Ω is a *p-system*,[1] then one can approximate Max-f to within a factor of $O(p)$; moreover, if Ω is the intersection of $O(1)$ knapsack constraints then one can approximate Max-f to within a constant factor.

Given this situation, it is natural to ask: *for which broad classes of functions can we approximately solve the Max-f problem efficiently?* Say, subject to constraints Ω that correspond to a p-system, or a small number q of knapsack constraints, or both. Clearly this class of functions includes submodular functions. Does this class contain other interesting classes of functions which may be far from being submodular?

In this paper we consider the case of "max-min optimization": here f is a monotone subadditive function defined by a minimization covering problem, a natural subset of all subadditive functions. We show conditions under which we can do constrained maximization over such functions f. For example, given a set system $(U, \mathcal{F})$, define the "set cover" function $f_{SC} : 2^U \to \mathbb{N}$, where $f_{SC}(S)$ is the minimum number of sets from $\mathcal{F}$ that cover the elements in S. This function f_{SC} is not submodular, and in fact, we can show that there is no submodular function g such that $g(S) \leq f_{SC}(S) \leq \alpha\ g(S)$ for sub-polynomial α. (See Section 6.) Moreover, note that in general we cannot even evaluate $f_{SC}(S)$ to better than an $O(\log n)$-approximation factor in polynomial time. However, our results imply that $\max_{S \in \Omega} f_{SC}(S)$ can indeed be approximated well. In fact, the result that one could approximately maximize f_{SC} subject to a cardinality constraint was given by [Feige et al. 2007]; our results should be seen as building on their ideas.

At a high level, our results imply that if a monotone function f is defined by a (minimization) covering problem, if f is subadditive, and if the underlying covering problem admits good deterministic online algorithms, then there exist good approximation algorithms for Max-f subject to p-systems and q knapsacks. All these terms will be made formal shortly. The resulting approximation guarantee for the max-min problem depends on the competitive ratio of the online algorithm, p and q. Moreover, the approximation ratio improves if there is a better algorithm for the offline minimization problem, or if there is a better online algorithm for a fractional version of the online minimization problem.

Robust Optimization. Our techniques and results imply approximation algorithms for covering problems in the framework of robust optimization as well. In particular, we consider the setting of two-stage demand-robust optimization. Here we are given a set system (U, Ω) (called the *uncertainty set*) representing possible demands, and an inflation parameter $\lambda \geq 1$. The actual demand is some set $A \in \Omega$. We want an algorithm that performs actions in two stages: the first stage is before the demand realization and the second stage is after the actual demand $A \in \Omega$ is observed. The action costs in the

[1] A p-system is similar to, but more general than, the intersection of p matroids; it is formally defined in Section 2.5.

second stage are a factor of λ larger than those in the first stage. The objective is to minimize the worst-case cost, i.e.

$$\text{(cost of first-stage actions)} \quad + \quad \lambda \cdot \max_{A \in \Omega} \text{(cost of second-stage actions for } A)$$

subject to the constraint that the two sets of actions "cover" the demand set $A \in \Omega$. As an example, in robust set cover, one is given another set system $(U, \mathcal{F})$: the allowed actions in the first and second stage are to pick some sub-collections $\mathcal{F}_1 \subseteq \mathcal{F}$ and $\mathcal{F}_2 \subseteq \mathcal{F}$ respectively; the notion of "coverage" is that the union of the sets in $\mathcal{F}_1 \cup \mathcal{F}_2$ must contain A. If $\lambda > 1$, actions are costlier in the second stage, and hence there is a natural tension between waiting for the identity of A, and over-anticipating in the first stage without any information about A.

Note that the robust and max-min objectives are related, at least in one direction: if $\lambda = 1$, there is no incentive to perform any actions in the first stage, in which case the robust objective degenerates into a max-min objective. In this paper, we show a reduction in the other direction as well—if one can approximate the max-min problem well and if the covering problem admits a good deterministic online algorithm, then we get an algorithm for the robust optimization version of the covering problem as well. Previously [Feige et al. 2007] gave a reduction from the robust set-cover problem to the max-min set cover problem, in the special case when $\Omega = \binom{U}{k}$; this result was based on a suitable LP-relaxation. Our reduction extends this in two ways: (a) the constraint sets Ω can now be intersections of p-systems and q knapsack constraints, and (b) more importantly, the reduction applies not only to set cover, but to many sub-additive covering problems (those with deterministic online algorithms).

Our Results and Techniques. Our algorithm for the max-min problem is based on the observation that the cost of a deterministic online algorithm for the underlying covering problem defining f can be used as a surrogate for submodularity. Specifically, we show that the greedy algorithm that repeatedly picks an element maintaining membership in Ω and maximizing the cost of the online algorithm gives us a good approximation to the max-min objective function, as long as Ω is a p-system. This result appears in Section 3.1.

We also show how to reduce the problem of maximizing such a function over the intersection of q knapsacks to $n^{O(1/\epsilon^2)}$ runs of maximizing the function over a single partition matroid, at a loss of a $q(1+\epsilon)$ factor. A variant of this reduction involves $n^{O(q/\epsilon^2)}$ runs of maximizing over a partition matroid, at a loss of a $(1+\epsilon)$ factor. These reductions are fairly general and likely to be of interest in other contexts as well. These appear in Section 3.2.

We then turn to robust optimization. In Section 4, we show that given a deterministic online algorithm for the covering function f, and a max-min algorithm for f over a family Ω, we get an algorithm for the two-stage robust version of the underlying covering problem with uncertainty set Ω—the approximation guarantee depends on the competitive ratio of the online algorithm as well as the approximation guarantee of the max-min problem.

Note that we can obtain approximation algorithms for robust optimization problems by combining the two results above (i) using online algorithms to obtain max-min algorithms and (ii) using max-min algorithms for robust optimization. In Section 5, we give a more careful analysis that gives better approximation ratios than that obtained by simply combining the results above.

Finally, in Section 6, we show that some common covering problems (vertex cover, set cover and multicut) give rise to functions f that cannot be well-approximated (in

a multiplicative sense) by *any* submodular function. Still, these max-min problems admit good approximation algorithms by our results in Section 3.

1.1. Related Work

Constrained submodular maximization problems have been very widely studied [Nemhauser et al. 1978; Fisher et al. 1978; Sviridenko 2004; Călinescu et al. 2011; Vondrák 2008; Kulik et al. 2013]. However, as we mentioned above, some of the covering problems we consider are far from submodular. Interestingly, in a recent paper on testing submodularity [Seshadhri and Vondrák 2011], the authors conjectured that the success of greedy maximization algorithms may depend on a more general property than submodularity; this work provides further corroboration for this, since we show that in our context online algorithms can serve as surrogates for submodularity.

The study of approximation algorithms for robust optimization was initiated by [Dhamdhere et al. 2005; Golovin et al. 2006]: they studied the case when the scenarios were *explicitly* listed, and gave constant-factor approximations for several combinatorial optimization problems. See also the surveys [A. Ben-Tal et al. 2009; Bertsimas et al. 2011] on robust optimization.

[Feige et al. 2007] introduced the k-max-min and k-robust set cover problems, which correspond to $\Omega = \binom{U}{k}$, i.e. "cardinality-constrained" uncertainty sets. This was the first setting with an exponential number of scenarios in the uncertainty set, for which good approximation algorithms were obtained. They gave an $O(\log m \log n)$-approximation algorithm for both versions, where m and n are the numbers of sets and elements. They also showed an $\Omega(\frac{\log m}{\log \log m})$ hardness of approximation for k-max-min (and k-robust) set cover. The algorithm for k-max-min set cover [Feige et al. 2007] used the connection to online algorithms; our results for max-min optimization build on this by handling more general covering problems and sets Ω. To the best of our knowledge, none of the max-min problems other than min-cut have been studied earlier; note that the min-cut function is submodular, and hence the associated max-min problem can be solved using submodular maximization. The algorithm for k-robust set cover [Feige et al. 2007] used their k-max-min algorithm within an LP-rounding approach (see also [Shmoys and Swamy 2004]) to get the same $O(\log m \log n)$-approximation guarantee. As mentioned earlier, our approach is different (we use online algorithms directly for robust optimization) and hence it applies to many other covering problems.

[Khandekar et al. 2013] noted that this LP-based technique does not imply good results for k-robust Steiner tree, and developed new combinatorial constant-factor approximations for k-robust versions of Steiner tree, Steiner forest on trees and facility location, again for the cardinality-constrained case.

We investigate many of these covering problems in the cardinality-constrained case of both the max-min and robust models in the companion paper [Gupta et al. 2014], and obtain approximation ratios significantly better than the online competitive factors. The algorithms and analysis in [Gupta et al. 2014] rely crucially on the structure of the cardinality uncertainty sets; and it is not clear if they can be extended to the setting of this paper. On the other hand, the goal in this paper is to give a framework for robust and max-min optimization under general uncertainty sets.

2. PRELIMINARIES

2.1. Deterministic covering problems

A covering problem Π has a ground-set E of elements with costs $c : E \to \mathbb{R}_+$, and n covering requirements (often called demands or clients), where the solutions to the i-th requirement are specified—possibly implicitly—by a family $\mathcal{R}_i \subseteq 2^E$ which is *upwards*

closed.[2] Requirement i is *satisfied* by solution $\mathcal{F} \subseteq E$ if and only if $\mathcal{F} \in \mathcal{R}_i$. We use $[n] := \{1, 2, \ldots, n\}$ to denote the set of covering requirements. The covering problem $\Pi = \langle E, c, \{\mathcal{R}_i\}_{i=1}^n \rangle$ involves computing a solution $\mathcal{F} \subseteq E$ satisfying all n requirements and having minimum cost $\sum_{e \in \mathcal{F}} c_e$. E.g., in set cover, "requirements" are items to be covered, and "elements" are sets to cover them with. In Steiner tree, requirements are terminals to connect to the root and elements are the edges; in multicut, requirements are terminal pairs to be separated, and elements are edges to be cut.

The min-cost covering function associated with Π is:

$$f_\Pi(S) := \min \left\{ \sum_{e \in \mathcal{F}} c_e \ : \ \mathcal{F} \in \mathcal{R}_i \text{ for all } i \in S \right\}, \qquad \forall\, S \subseteq [n].$$

2.2. Max-min problems

Given a covering problem Π and a collection $\Omega \subseteq 2^{[n]}$ of "independent sets", the *max-min* problem $\mathsf{MaxMin}(\Pi)$ involves finding a set $\omega \in \Omega$ for which the cost of the min-cost solution to ω is maximized,

$$\max_{\omega \in \Omega} \ f_\Pi(\omega).$$

2.3. Robust covering problems

This problem, denoted $\mathsf{Robust}(\Pi)$, is a *two-stage optimization* problem, where elements are chosen in either the first stage (at the given cost) or the second stage (at cost λ times higher). In the second stage, some subset $\omega \subseteq [n]$ of requirements (also called a *scenario*) materializes, and the elements bought in both stages must collectively satisfy each requirement in ω. Formally, the input to problem $\mathsf{Robust}(\Pi)$ consists of (a) the covering problem $\Pi = \langle E, c, \{\mathcal{R}_i\}_{i=1}^n \rangle$ as above, (b) an uncertainty set $\Omega \subseteq 2^{[n]}$ of scenarios (possibly implicitly given), and (c) an inflation parameter $\lambda \geq 1$. A feasible solution to $\mathsf{Robust}(\Pi)$ is a set of *first stage elements* $E_0 \subseteq E$ (bought without knowledge of the scenario), along with an *augmentation algorithm* that given any $\omega \in \Omega$ outputs $E_\omega \subseteq E$ such that $E_0 \cup E_\omega$ satisfies all requirements in ω. The objective function is to minimize:

$$c(E_0) + \lambda \cdot \max_{\omega \in \Omega} c(E_\omega).$$

Given such a solution, $c(E_0)$ is called the first-stage cost and $\max_{\omega \in \Omega} c(E_\omega)$ is the second-stage cost.

Note that by setting $\lambda = 1$ in any robust covering problem, *the optimal value of the robust problem equals that of its corresponding max-min problem*.

As in [Gupta et al. 2014], our algorithms for robust covering problems are based on the following type of guarantee.

Definition 2.1. An algorithm is $(\alpha_1, \alpha_2, \beta)$-*discriminating* if and only if given as input any instance of $\mathsf{Robust}(\Pi)$ and a threshold T, the algorithm outputs

- a set $\Phi_T \subseteq E$, and
- an algorithm $\mathsf{Augment}_T : \Omega \to 2^E$,

such that:

A. For every scenario $D \in \Omega$,
   (i) the elements in $\Phi_T \cup \mathsf{Augment}_T(D)$ satisfy all requirements in D, and
   (ii) the resulting augmentation cost $c\left(\mathsf{Augment}_T(D)\right) \leq \beta \cdot T$.

[2] A family $\mathcal{R}$ of subsets is called *upwards closed* if, for each $S \in \mathcal{R}$ and $T \supseteq S$ we also have $T \in \mathcal{R}$. Similarly $\mathcal{R}$ is called *downwards closed* if, for each $S \in \mathcal{R}$ and $T \subseteq S$ we also have $T \in \mathcal{R}$.

B. Let Φ^* and T^* (respectively) denote the first-stage and second-stage cost of an optimal solution to the Robust(Π) instance. If the threshold $T \geq T^*$ then the first stage cost $c(\Phi_T) \leq \alpha_1 \cdot \Phi^* + \alpha_2 \cdot T^*$.

LEMMA 2.2 ([GUPTA ET AL. 2014]). *If there is an $(\alpha_1, \alpha_2, \beta)$-discriminating algorithm for a robust covering problem* Robust(Π)*, then for every $\epsilon > 0$ there is a $\left((1+\epsilon) \cdot \max\left\{\alpha_1, \beta + \frac{\alpha_2}{\lambda}\right\}\right)$-approximation algorithm for* Robust(Π).

Although this lemma was only stated for k-robust uncertainty sets in [Gupta et al. 2014], its proof immediately extends to arbitrary uncertainty sets.

2.4. Desirable Properties of the Covering Problem

We now formalize certain properties of the covering problem $\Pi = \langle E, c, \{\mathcal{R}_i\}_{i=1}^n \rangle$ that are useful in obtaining our results. Given a partial solution $S \subseteq E$ and a set $X \subseteq [n]$ of requirements, any set $E_X \subseteq E$ such that $S \cup E_X \in \mathcal{R}_i \; \forall i \in X$ is called an *augmentation* of S for requirements X. Given $X \subseteq [n]$ and $S \subseteq E$, define the min-cost augmentation of S for requirements X as:

$$\mathsf{OptAug}(X \mid S) := \min\{c(E_X) \mid E_X \subseteq E \text{ and } S \cup E_X \in \mathcal{R}_i, \; \forall i \in X\}.$$

Also define $\mathsf{Opt}(X) := \min\{c(E_X) \mid E_X \subseteq E \text{ and } E_X \in \mathcal{R}_i \; \forall i \in X\} = \mathsf{OptAug}(X \mid \emptyset)$, for any $X \subseteq [n]$.

An easy consequence of the fact that costs are non-negative is the following:

PROPERTY 2.3 (MONOTONICITY). *For any requirements $X \subseteq Y \subseteq [n]$ and any solution $S \subseteq E$,* $\mathsf{OptAug}(X|S) \leq \mathsf{OptAug}(Y|S)$. *Similarly, for any $X \subseteq [n]$ and solutions $T \subseteq S \subseteq E$,* $\mathsf{OptAug}(X \mid S) \leq \mathsf{OptAug}(X \mid T)$.

From the definition of coverage of requirements, we obtain:

PROPERTY 2.4 (SUBADDITIVITY). *For any two subsets of requirements $X, Y \subseteq [n]$ and any partial solution $S \subseteq E$, we have:*

$$\mathsf{OptAug}(X \mid S) + \mathsf{OptAug}(Y \mid S) \geq \mathsf{OptAug}(X \cup Y \mid S).$$

To see this property: if $\mathcal{F}_X \subseteq E$ and $\mathcal{F}_Y \subseteq E$ are solutions corresponding to $\mathsf{OptAug}(X \mid S)$ and $\mathsf{OptAug}(Y \mid S)$ respectively, then $\mathcal{F}_X \cup \mathcal{F}_Y \cup S$ covers requirements $X \cup Y$; so $\mathsf{OptAug}(X \cup Y \mid S) \leq c(\mathcal{F}_X \cup \mathcal{F}_Y) \leq c(\mathcal{F}_X) + c(\mathcal{F}_Y) = \mathsf{OptAug}(X \mid S) + \mathsf{OptAug}(Y \mid S)$.

We assume two additional properties of the covering problem.

PROPERTY 2.5 (OFFLINE ALGORITHM). *There is a polynomial time α_{off}-approximation algorithm for the offline covering problem* $\mathsf{OptAug}(X \mid S)$*, for any $S \subseteq E$ and $X \subseteq [n]$.*

An online algorithm for a covering problem $\Pi = \langle E, c, \{\mathcal{R}_i\}_{i=1}^n \rangle$ is the following. It is given in advance the ground-set E, cost function c, and the set $[n]$ of *possible* covering requirements. The actual subset of requirements arrive online one by one. The algorithm must always maintain a feasible solution for the arrived requirements, and the solution must be *monotone* (i.e. elements can only be added to the current solution). The competitive ratio of an online algorithm is defined to be the maximum ratio (over all input sequences τ) of the algorithm's cost to the optimal (offline) cost for covering the requirements τ. We also require:

PROPERTY 2.6 (ONLINE ALGORITHM). *There is a polynomial-time deterministic α_{on}-competitive algorithm for the online version of $\Pi = \langle E, c, \{\mathcal{R}_i\}_{i=1}^n \rangle$.*

2.5. Models of Downward-Closed Families

All covering functions we deal with are monotone non-decreasing. So we may assume, without loss of generality, that the collection Ω in both MaxMin(Π) and Robust(Π) is *downwards-closed*, i.e. $A \subseteq B$ and $B \in \Omega \implies A \in \Omega$. In this paper we consider the following well-studied classes of downward-closed families.

Definition 2.7 (p-*system*). A downward-closed family $\Omega \subseteq 2^{[n]}$ is called a p-system if and only if:

$$\frac{\max_{I \in \overline{\Omega}(A)} |I|}{\min_{J \in \overline{\Omega}(A)} |J|} \leq p, \quad \text{for each } A \subseteq [n],$$

where $\overline{\Omega}(A)$ denotes the collection of *maximal subsets* in $\{S \in \Omega : S \subseteq A\}$.

Sets in Ω are called *independent sets*. We assume access to a membership-oracle, that given any subset $I \subseteq [n]$ returns whether or not $I \in \Omega$.

Definition 2.8 (q-*knapsack*). Given q non-negative functions $w^1, \ldots, w^q : [n] \to \mathbb{R}_+$ and capacities $b_1, \ldots, b_q \in \mathbb{R}_+$, the q-knapsack constrained family is:

$$\Omega = \left\{ A \subseteq [n] \,:\, \sum_{e \in A} w^j(e) \leq b_j, \;\; \text{for all } j \in \{1, \ldots, q\} \right\}.$$

These constraints model a rich class of downward-closed families. Some interesting special cases of p-systems are p-matroid intersection [Schrijver 2003] and p-set packing [Hurkens and Schrijver 1989; Berman 2000]; see the appendix in [Călinescu et al. 2011] for more discussion on p-systems. [Jenkyns 1976] showed that the natural greedy algorithm is a p-approximation for maximizing linear functions over p-systems, which is the best known result. Maximizing a linear function over q-knapsack constraints is the well-studied class of packing integer programs (PIPs), eg. [Srinivasan 1999]. Again, the greedy algorithm is known to achieve an $O(q)$-approximation ratio. When the number of constraints q is constant, there is a PTAS [Frieze and Clarke 1984].

3. ALGORITHMS FOR MAX-MIN OPTIMIZATION

In this section we give approximation algorithms for constrained max-min optimization, i.e. problem (Max-f) where f is given by some underlying covering problem and Ω is the intersection of a p-system and q-knapsack. We first consider the case when Ω is a p-system. Then we show that any knapsack constraint can be reduced to a 1-system (specifically a partition matroid) in a black-box fashion; this enables us to obtain an algorithm for Ω being the intersection of a p-system and q-knapsack. The results of this section assume Properties 2.4 and 2.6.

3.1. Algorithm for p-System Constraints

Algorithm 1 for the max-min problem is a greedy algorithm. However it is relative to the objective of the online algorithm $\mathcal{A}_{on}$ from Property 2.6 rather than the (approximate) function value itself. For a sequence π of covering requirements, we let $\mathcal{A}_{on}(\pi) \subseteq E$ denote the solution constructed by the (deterministic) online algorithm $\mathcal{A}_{on}$ upon seeing the input sequence π.

THEOREM 3.1. *Assuming Properties 2.4 and 2.6 Algorithm 1 is a* $((p+1)\,\alpha_{\mathsf{on}})$-*approximation algorithm for* MaxMin(Π) *under* p-*systems.*

PROOF. The proof of this lemma closely follows that in [Călinescu et al. 2011] for submodular maximization over a p-system. We use slightly more notation than necessary since this proof will be used in the next section as well.

ALGORITHM 1: Algorithm for MaxMin(Π) under p-system

Input: covering instance Π that defines f and p-system Ω.
let current scenario $A_0 \leftarrow \emptyset$, counter $i \leftarrow 0$, input sequence $\sigma \leftarrow \langle\rangle$;
while *($\exists e \in [n] \setminus A_i$ such that $A_i \cup \{e\} \in \Omega$)* **do**
  $a_{i+1} \leftarrow \arg\max \{c(\mathcal{A}_{on}(\sigma \circ e)) - c(\mathcal{A}_{on}(\sigma)) \ : \ e \in [n] \setminus A_i \text{ and } A_i \cup \{e\} \in \Omega\}$;
  let $\sigma \leftarrow \sigma \circ a_{i+1}, \ A_{i+1} \leftarrow A_i \cup \{a_{i+1}\}, \ i \leftarrow i+1$;
end
let $D \leftarrow A_i$ be the independent set constructed by the above loop;
Output: solution D.

Suppose that the algorithm performed k iterations; let $D = \{a_1, \cdots, a_k\}$ be the ordered set of elements added by the algorithm. Define $\sigma = \langle\rangle$, $G_0 := \emptyset$, and $G_i := \mathcal{A}_{on}(\sigma \circ a_1 \cdots a_i)$ for each $i \in [k]$. Note that $G_0 \subseteq G_1 \subseteq \cdots \subseteq G_k$. It suffices to show that:

$$\mathsf{OptAug}(B \mid G_0) \leq (p+1) \cdot c(G_k \setminus G_0) \qquad \text{for every } B \in \Omega. \tag{3.1}$$

This would imply $\mathsf{Opt}(B) \leq (p+1) \cdot c(G_k) \leq (p+1)\,\alpha_{\mathsf{on}} \cdot \mathsf{Opt}(D)$ for every $B \in \Omega$, and hence that D is the desired approximate solution. Also, $c(G_k)/\alpha_{\mathsf{on}}$ is output as the (approximate) MaxMin(Π) value.

We use the following claim proved in [Călinescu et al. 2011], which relies on the properties of a p-system.

CLAIM 3.2 ([CĂLINESCU ET AL. 2011]). *Given p-system Ω, $D = \{a_1, \ldots, a_k\} \in \Omega$ and any $B \in \Omega$, there exists a partition $\{B_i\}_{i=1}^{k}$ of B such that for all $i \in [k]$,*

(1) $|B_i| \leq p$, *and*
(2) For every $e \in B_i$, we have $\{a_1, \cdots, a_{i-1}\} \bigcup \{e\} \in \Omega$.

For any sequence π of requirements and any $e \in [n]$ define $\mathsf{Aug}(e;\pi) := c(\mathcal{A}_{on}(\pi \circ e)) - c(\mathcal{A}_{on}(\pi))$. Note that this function depends on the particular online algorithm. From the second condition in Claim 3.2, it follows that each element of B_i was a feasible augmentation to $\{a_1, \ldots, a_{i-1}\}$ in the i^{th} iteration of the **while** loop. By the greedy choice,

$$\begin{aligned}
c(G_i) - c(G_{i-1}) \ = \ \mathsf{Aug}(a_i; \sigma \circ a_1 \cdots a_{i-1}) \ &\geq \ \max_{e \in B_i} \mathsf{Aug}(e; \sigma \circ a_1 \cdots a_{i-1}) \\
&\geq \ \frac{1}{|B_i|} \sum_{e \in B_i} \mathsf{Aug}(e; \sigma \circ a_1 \cdots a_{i-1}) \\
&\geq \ \frac{1}{|B_i|} \sum_{e \in B_i} \mathsf{OptAug}(\{e\} \mid G_{i-1}) \qquad (3.2) \\
&\geq \ \frac{1}{|B_i|} \cdot \mathsf{OptAug}(B_i \mid G_{i-1}) \qquad (3.3) \\
&\geq \ \frac{1}{p} \cdot \mathsf{OptAug}(B_i \mid G_{i-1}). \qquad (3.4)
\end{aligned}$$

Above equation (3.2) is by the definition of $G_{i-1} = \mathcal{A}_{on}(\sigma \circ a_1 \cdots a_{i-1})$, equation (3.3) uses the subadditivity Property 2.4, and (3.4) is by the first condition in Claim 3.2.

Summing over all iterations $i \in [k]$, we obtain:

$$c(G_k) - c(G_0) = \sum_{i=1}^{k} \mathsf{Aug}(a_i; \sigma \circ a_1 \cdots a_{i-1}) \geq \frac{1}{p} \sum_{i=1}^{k} \mathsf{OptAug}(B_i \mid G_{i-1}) \geq \frac{1}{p} \sum_{i=1}^{k} \mathsf{OptAug}(B_i \mid G_k)$$

where the last inequality follows from monotonicity since $G_{i-1} \subseteq G_k$ for all $i \in [k]$.

Using subadditivity Property 2.4, we get $c(G_k) - c(G_0) \geq \frac{1}{p} \cdot \mathsf{OptAug}(\cup_{i=1}^{k} B_i \mid G_k) = \frac{1}{p} \cdot \mathsf{OptAug}(B \mid G_k)$.

Let $J_B := \arg\min\{c(J') \mid J' \subseteq E, \text{ and } G_k \cup J' \in \mathcal{R}_e, \ \forall e \in B\}$. i.e. $\mathsf{OptAug}(B \mid G_k) = c(J_B)$. Observe that $J_B \cup (G_k \setminus G_0)$ is a feasible augmentation to G_0 that covers requirements B. Thus,

$$\mathsf{OptAug}(B \mid G_0) \leq c(J_B) + c(G_k \setminus G_0) = \mathsf{OptAug}(B \mid G_k) + c(G_k \setminus G_0) \leq (p+1) \cdot c(G_k \setminus G_0).$$

This completes the proof. □

3.2. Reducing knapsack constraints to partition matroids

In this subsection we show that every knapsack constraint can be reduced to a suitable collection of partition matroids. This property is then used to complete the algorithm for $\mathsf{MaxMin}(\Pi)$ when Ω is given by the intersection of a p-system a q-knapsack. Observe that even a single knapsack constraint need not correspond exactly to a small p-system: eg. the knapsack with weights $w_1 = 1$ and $w_2 = w_3 = \cdots = w_n = 1/n$, and capacity one is only an $(n-1)$-system (since both $\{1\}$ and $\{2, 3, \cdots, n\}$ are maximal independent sets). However we show that any knapsack constraint can be *approximately* reduced to a partition matroid (which is a 1-system). The main idea in this reduction is an enumeration method from [Chekuri and Khanna 2005].

A *partition matroid* $\mathcal{P}$ on groundset $[n]$ is given by a partition $\{P_1, \ldots, P_\ell\}$ of $[n]$ into ℓ parts with respective bounds $n_1, \ldots, n_\ell \geq 0$. This implicitly defines a family $\mathcal{I} \subseteq 2^{[n]}$ of *independent sets* where $S \in \mathcal{I}$ if and only if $|S \cap P_j| \leq n_j$ for all $j \in \{1, \ldots, \ell\}$. Notice that independent sets of a partition matroid form a 1-system according to Definition 2.7.

LEMMA 3.3. *Given any knapsack constraint $\sum_{i=1}^{n} w_i \cdot x_i \leq B$ and fixed $0 < \epsilon \leq 1$, there is a polynomial-time computable collection of $T = n^{O(1/\epsilon^2)}$ partition matroids, with $\mathcal{I}_1, \cdots, \mathcal{I}_T$ denoting the respective families of independent sets, such that:*

(1) For every $X \in \cup_{t=1}^{T} \mathcal{I}_t$, we have $\sum_{i \in X} w_i \leq (1+\epsilon) \cdot B$.
(2) $\{X \subseteq [n] \mid \sum_{i \in X} w_i \leq B\} \subseteq \cup_{t=1}^{T} \mathcal{I}_t$.

PROOF. Let $\delta = \epsilon/6$ and $\beta = \frac{\delta B}{n}$. Without loss of generality, we assume that $\max_{i=1}^{n} w_i \leq B$. Partition the groundset $[n]$ into $G := \lceil \frac{\log(n/\delta)}{\log(1+\delta)} \rceil$ parts as follows.

$$S_k := \begin{cases} \{i \in [n] : w_i \leq \beta\} & \text{if } k = 0 \\ \left\{i \in [n] : \beta \cdot (1+\delta)^{k-1} < w_i \leq \beta \cdot (1+\delta)^k\right\} & \text{if } 1 \leq k \leq G \end{cases}$$

Let T denote the number of non-negative integer partitions of the number $\lceil G/\delta \rceil$ into G parts. Note that,

$$T := \binom{\lceil G/\delta \rceil + G - 1}{G - 1} \leq \exp(\lceil G/\delta \rceil + G - 1) \leq n^{O(1/\delta^2)}.$$

We will define a collection of T *partition matroids* on $[n]$, each over the partition $\{S_0, S_1, \ldots, S_G\}$. For any integer partition $\tau = \{U_k\}_{k=1}^{G}$ of $\lceil G/\delta \rceil$ (i.e. $U_k \geq 0$ are integers and $\sum_k U_k = \lceil G/\delta \rceil$), define a partition matroid $\mathcal{P}_\tau$ that has bounds $N_k(\tau)$ on each part S_k, where

$$N_k(\tau) := \begin{cases} \infty & \text{if } k = 0 \\ \lfloor \frac{n \cdot (U_k + 1)}{G \cdot (1+\delta)^{k-1}} \rfloor & \text{if } 1 \leq k \leq G \end{cases}$$

We let $\mathcal{I}_\tau$ denote the independent sets of $\mathcal{P}_\tau$. Clearly this collection of partition matroids can be constructed in polynomial time for fixed ϵ. We now show that this collection satisfies the two properties in the lemma.

(1) Consider any $X \in \mathcal{I}_\tau$, an independent set in some partition matroid $\mathcal{P}_\tau$. The total weight of elements $X \cap S_0$ is at most $n \cdot \beta \leq \delta \cdot B$. For any group $1 \leq k \leq G$, the weight of elements $X \cap S_k$ is at most:

$$|X \cap S_k| \cdot \beta\,(1+\delta)^k \leq N_k(\tau) \cdot \beta\,(1+\delta)^k \leq \delta(1+\delta)(U_k+1) \cdot \frac{B}{G}$$

Hence the total weight of all elements in X is at most:

$$\begin{aligned}
\delta B + \delta(1+\delta)\frac{B}{G} \cdot \left(\sum_{k=1}^{G} U_k + G\right) &\leq \delta B + \delta(1+\delta)\frac{B}{G} \cdot \left(\frac{G}{\delta} + 1 + G\right) \\
&\leq \delta B + \delta(1+\delta)\frac{B}{G} \cdot \left(\frac{G}{\delta} + 2G\right) \\
&\leq \delta B + (1+\delta) \cdot (B + 2\delta\, B) \\
&\leq B + 6\delta\, B.
\end{aligned}$$

Above we use $\delta \leq 1$. Finally since $\delta = \epsilon/6$, we obtain the first condition.

(2) Consider any $Y \subseteq [n]$ that satisfies the knapsack constraint, i.e. $\sum_{i \in Y} w_i \leq B$. We will show that Y lies in $\mathcal{I}_\tau$, for some integer partition τ of $\lceil G/\delta \rceil$ as above. For each $1 \leq k \leq G$ let Q_k denote the weight of elements in $Y \cap S_k$, and U_k be the unique integer that satisfies $U_k \cdot \frac{\delta B}{G} \leq Q_k < (U_k+1) \cdot \frac{\delta B}{G}$. Define τ to be the integer partition $\{U_k\}_{k=1}^{G}$. We have $\sum_k U_k \leq G/\delta$, which follows from the fact $B \geq \sum_k Q_k \geq \frac{\delta B}{G} \cdot \sum_k U_k$. By increasing U_ks arbitrarily so that they total to $\lceil G/\delta \rceil$, we obtain a feasible integer partition τ. We now claim that Y lies in $\mathcal{I}_\tau$. Since each element of S_k (for $k \geq 1$) has weight at least $\beta \cdot (1+\delta)^{k-1}$, we have

$$|Y \cap S_k| \leq \frac{Q_k}{\beta\,(1+\delta)^{k-1}} \leq \frac{(U_k+1) \cdot \delta B/G}{(1+\delta)^{k-1} \cdot \delta B/n} = \frac{n \cdot (U_k+1)}{G \cdot (1+\delta)^{k-1}}.$$

Since $|Y \cap S_k|$ is integral, we obtain $|Y \cap S_k| \leq \lfloor \frac{n \cdot (U_k+1)}{G \cdot (1+\delta)^{k-1}} \rfloor \leq N_k(\tau)$. Thus $Y \in \mathcal{I}_\tau$ and we obtain the second condition. □

3.3. Algorithm for p-System and q-Knapsack Constraints

Here we consider MaxMin(Π) when Ω is the intersection of p-system $\mathcal{M}$ and a q-knapsack (as in Definition 2.8). The idea is to reduce the q-knapsack to a single knapsack (losing factor $\approx q$), then use Lemma 3.3 to reduce the knapsack to a 1-system, and finally apply Theorem 3.1 on the resulting $p+1$ system. Details appear below.

By scaling weights in the knapsack constraints, we may assume, without loss of generality, that each knapsack has capacity exactly one; let $w^1, \cdots, w^q$ denote the weights in the q knapsack constraints. We also assume, without loss of generality, that each singleton element satisfies the q-knapsack; otherwise such elements can be dropped from the groundset. The algorithm for MaxMin(Π) under p-system and q-knapsack constraints is as follows.

(1) Approximate the q-knapsack by a single knapsack with weights $\sum_{j=1}^{q} w^j$ and capacity q; applying Lemma 3.3 with $\epsilon = \frac{1}{2}$ on this knapsack, let $\{\mathcal{I}_j\}_{j=1}^{L}$ denote the independent sets of the resulting partition matroids. Note that $L = n^{O(1)}$.
(2) For each $j \in \{1, \ldots, L\}$, define $\Sigma_j := \mathcal{M} \bigcap \mathcal{I}_j$.
(3) Run the algorithm from Theorem 3.1 under each $p+1$ system $\{\Sigma_j\}_{j=1}^{L}$ to obtain solutions $\{E_j \in \Sigma_j\}_{j=1}^{L}$.
(4) Let $j^* \leftarrow \arg\max_{j=1}^{L} c\,(\mathcal{A}_{on}(E_j))$.

(5) Partition E_{j^*} into $\{\omega_i\}_{i=1}^{3q}$ such that each $\omega_i \in \Omega$, as per Claim 3.5.
(6) Output ω_{i^*} where $i^* \leftarrow \arg\max_{i=1}^{3q} c(\mathcal{A}_{off}(\omega_i))$. Here we use the offline algorithm from Property 2.5.

Observe that in Step 2 of this algorithm, each Σ_j is a $(p+1)$-system since it is the intersection of $\mathcal{M}$ which is a p-system and $\mathcal{I}_j$ which is a 1-system (independent sets of a partition matroid). We now establish the approximation ratio of this algorithm.

CLAIM 3.4. $\Omega \subseteq \cup_{j=1}^{L} \Sigma_j$.

PROOF. For any $\omega \in \Omega$, we have $\sum_{e\in\omega} w^i(e) \leq 1$ for all $i \in [q]$. Hence $\sum_{e\in\omega}\sum_{i=1}^{q} w^i(e) \leq q$, i.e. it satisfies the combined knapsack constraint. Now by Lemma 3.3 (2), we obtain $\omega \in \bigcap_{j=1}^{L} \mathcal{I}_j$.
Finally, since $\omega \in \Omega \subseteq \mathcal{M}$, we have $\omega \in \cup_{j=1}^{L}\Sigma_j$. □

CLAIM 3.5. *For each $\tau \in \cup_{j=1}^{L}\Sigma_j$ there is a polynomial-time computable collection $\{\omega_i\}_{i=1}^{3q}$ such that $\tau = \bigcup_{\ell=1}^{3q} \omega_\ell$, and $\omega_\ell \in \Omega$ for all $\ell \in [3q]$.*

PROOF. Consider any $\tau \in \Sigma := \cup_{j=1}^{L}\Sigma_j$. Note that $\tau \in \mathcal{M}$, so any subset of τ is also in $\mathcal{M}$ (which is downwards-closed). We will show that there is a partition of τ into $\{\omega_\ell\}_{\ell=1}^{3q}$ such that each ω_ℓ satisfies the q-knapsack. This suffices to prove the claim. Since $\tau \in \bigcap_{j=1}^{L}\mathcal{I}_j$, by Lemma 3.3 (1) it follows that $\sum_{e\in\tau}\sum_{i=1}^{q} w^i(e) \leq \frac{3}{2}q$. Starting with the trivial partition of τ into singleton elements, greedily merge parts as long as each part satisfies the q-knapsack, until no further merge is possible. Note that the trivial partition is indeed feasible since each element satisfies the q-knapsack. Let $\{\omega_\ell\}_{\ell=1}^{r}$ denote the parts in the final partition; we will show $r \leq 3q$ which would prove the claim. Observe that for any pair $\{\omega, \omega'\}$, it must be that $\omega \cup \omega'$ violates *some* knapsack; so $\sum_{e\in\omega\cup\omega'}\sum_{i=1}^{q} w^i(e) > 1$. Adding this inequality over the r pairs $\{\omega_1,\omega_2\},\{\omega_2,\omega_3\},\cdots\{\omega_r,\omega_1\}$, we obtain $2\sum_{e\in\tau}\sum_{i=1}^{q} w^i(e) > r$. On the other hand, $\sum_{e\in\tau}\sum_{i=1}^{q} w^i(e) \leq \frac{3}{2}q$, which implies $r < 3q$. □

THEOREM 3.6. *Assuming Properties 2.4, 2.5 and 2.6, there is an $O((p+1)(q+1)\,\alpha_{\mathsf{off}}\,\alpha_{\mathsf{on}})$-approximation algorithm for* MaxMin(Π) *under a p-system and q-knapsack constraint.*

PROOF. Let Opt_j denote the optimal value of MaxMin(Π) under $p+1$ system Σ_j, for each $j \in [L]$. By Claim 3.4 we have $\max_{j=1}^{L} \mathsf{Opt}_j \geq \mathsf{Opt}$, the optimal value of MaxMin(Π) under Ω. Observe that Theorem 3.1 actually implies $c(\mathcal{A}_{on}(E_j)) \geq \frac{1}{p+2}\cdot \mathsf{Opt}_j$ for each $j \in [q]$. Thus $c(\mathcal{A}_{on}(E_{j^*})) \geq \frac{1}{p+2}\cdot \mathsf{Opt}$; hence $\mathsf{Opt}(E_{j^*}) \geq \frac{1}{\alpha_{\mathsf{on}}\,(p+2)}\cdot \mathsf{Opt}$. Now consider the partition $\{\omega_i\}_{i=1}^{3q}$ of E_{j^*} from Claim 3.5. By the subadditivity property, $\sum_{i=1}^{3q}\mathsf{Opt}(\omega_i) \geq \mathsf{Opt}(E_{j^*})$; i.e. there is some $i' \in [3q]$ with $\mathsf{Opt}(\omega_{i'}) \geq \frac{1}{\alpha_{\mathsf{on}}\,(p+2)(3q)}\cdot \mathsf{Opt}$. Thus the i^* found using the offline algorithm (Property 2.5) satisfies $\mathsf{Opt}(\omega_{i^*}) \geq \frac{1}{\alpha_{\mathsf{on}}\,\alpha_{\mathsf{off}}\,(p+2)(3q)}\cdot \mathsf{Opt}$. The approximate MaxMin(Π) value is $c(\mathcal{A}_{off}(\omega_{i^*}))/\alpha_{\mathsf{off}}$. □

Remark: We can obtain a better approximation guarantee of $O((p+1)(q+1)\,\alpha_{\mathsf{on}})$ in Theorem 3.6 using randomization; but where the algorithm does not output the approximate max-min value. The algorithm is same as the above, except for the last step, where we output ω_ℓ for $\ell \in [3q]$ chosen *uniformly at random*. From the above proof

of Theorem 3.6, it follows that:

$$E[\mathsf{Opt}(\omega_\ell)] = \frac{1}{3q}\sum_{i=1}^{3q}\mathsf{Opt}(\omega_i) \geq \frac{\mathsf{Opt}(E_{j^*})}{3q} \geq \frac{1}{\alpha_{\mathsf{on}}\,(p+2)(3q)}\cdot \mathsf{Opt}.$$

4. GENERAL FRAMEWORK FOR ROBUST COVERING PROBLEMS

In this section we present an abstract framework for robust covering problems under *any uncertainty set* Ω, as long as we are given access to offline, online, and max-min algorithms for the base covering problem. Formally, this requires Properties 2.5, 2.6 and the following additional property (recall the notation from Section 2).

PROPERTY 4.1 (MAX-MIN ALGORITHM). *There is an α_{mm}-approximation algorithm for the max-min problem:*

$$\textit{Given input } S \subseteq E, \quad \mathsf{MaxMin}(S) \;:=\; \max_{X\in\Omega} \min\{c(A) \mid S\cup A \in R_i,\ \forall i \in X\}.$$

THEOREM 4.2. *Under Properties 2.4, 2.5, 2.6 and 4.1, Algorithm 2 is an $O(\alpha_{\mathsf{off}}\cdot \alpha_{\mathsf{on}}\cdot\alpha_{\mathsf{mm}})$-approximation algorithm for the robust covering problem* $\mathsf{Robust}(\Pi) = \langle E, c, \{R_i\}_{i=1}^n, \Omega, \lambda\rangle$.

PROOF. The algorithm proceeds as follows.

ALGORITHM 2: Algorithm Robust-with-General-Uncertainty-Sets

Input: $\mathsf{Robust}(\Pi)$ instance and threshold T.
let counter $t \leftarrow 0$, initial online algorithm's input $\sigma = \langle\rangle$, initial online solution $F_0 \leftarrow \emptyset$;
repeat
  set $t \leftarrow t+1$;
  let $E_t \subseteq [n]$ be the scenario returned by the algorithm of Property 4.1 on $\mathsf{MaxMin}(F_{t-1})$;
  let $\sigma \leftarrow \sigma \circ E_t$, and $F_t \leftarrow \mathcal{A}_{on}(\sigma)$ be the current online solution;
until $c(F_t) - c(F_{t-1}) \leq 2\alpha_{\mathsf{on}}\cdot T$;
set $\tau \leftarrow t-1$;
Output: first-stage solution $\Phi_T := F_\tau$, and second-stage solution $\mathsf{Augment}_T$ where for any $\omega \subseteq [n]$, $\mathsf{Augment}_T(\omega)$ is the solution of the offline algorithm (Property 2.5) for the problem $\mathsf{OptAug}(\omega \mid \Phi_T)$.

Let $\Phi^* \subseteq E$ denote the optimal first stage solution (and its cost), and T^* the optimal second-stage cost; so the optimal value is $\Phi^* + \lambda\cdot T^*$ (notation as in Definition 2.1). We prove the performance guarantee using the following claims.

CLAIM 4.3 (GENERAL 2ND STAGE). *For any $T \geq 0$ and $X \in \Omega$,*
- *elements $\Phi_T \bigcup \mathsf{Augment}_T(X)$ satisfy all the requirements in X, and*
- *$c(\mathsf{Augment}_T(X)) \leq 2\alpha_{\mathsf{off}}\cdot\alpha_{\mathsf{mm}}\cdot\alpha_{\mathsf{on}}\cdot T$.*

PROOF. It is clear that $\Phi_T \bigcup \mathsf{Augment}_T(X)$ satisfy all requirements in X. By the choice of set $E_{\tau+1}$ in the last iteration, for any $X \in \Omega$ we have:

$$\mathsf{OptAug}(X \mid F_\tau) \leq \alpha_{\mathsf{mm}}\cdot \mathsf{OptAug}(E_{\tau+1} \mid F_\tau) \leq \alpha_{\mathsf{mm}}\cdot(c(F_{\tau+1}) - c(F_\tau)) \leq 2\alpha_{\mathsf{mm}}\cdot\alpha_{\mathsf{on}}\cdot T$$

The first inequality is by Property 4.1, the second inequality uses the fact that $F_{\tau+1} \supseteq F_\tau$ (since we use an online algorithm to augment F_{t-1} to F_t),[3] and the last inequality

[3]This is the technical reason we need an online algorithm. If instead we had used an offline algorithm to compute F_t then $F_t \not\supseteq F_{t-1}$, and we could not upper bound the augmentation cost $\mathsf{OptAug}(E_t \mid F_{t-1})$ by $c(F_t) - c(F_{t-1})$.

follows from the termination condition of the repeat loop. Finally, since $\mathsf{Augment}_T(X)$ is an α_{off}-approximation to $\mathsf{OptAug}(X \mid F_\tau)$, we obtain the claim. □

CLAIM 4.4. $\mathsf{Opt}(\cup_{t\le\tau} E_t) \le \tau \cdot T^* + \Phi^*$.

PROOF. Since each $E_t \in \Omega$ (these are solutions to the MaxMin instances), the bound on the second-stage optimal cost gives $\mathsf{OptAug}(E_t \mid \Phi^*) \le T^*$ for all $t \le \tau$. By subadditivity (Property 2.4) we have $\mathsf{OptAug}(\cup_{t\le\tau} E_t \mid \Phi^*) \le \tau \cdot T^*$, which immediately implies the claim. □

CLAIM 4.5. $\mathsf{Opt}(\cup_{t\le\tau} E_t) \ge \frac{1}{\alpha_{\mathsf{on}}} \cdot c(F_\tau)$.

PROOF. This follows directly from the competitiveness of the online algorithm in Property 2.6. □

CLAIM 4.6 (GENERAL 1ST STAGE). *If $T \ge T^*$ then $c(\Phi_T) = c(F_\tau) \le 2\,\alpha_{\mathsf{on}} \cdot \Phi^*$.*

PROOF. We have $c(F_\tau) = \sum_{t=1}^{\tau} [c(F_t) - c(F_{t-1})] > 2\alpha_{\mathsf{on}}\tau \cdot T \ge 2\alpha_{\mathsf{on}}\tau \cdot T^*$ by the termination condition. Combined with Claim 4.5, we have $\mathsf{Opt}(\cup_{t\le\tau} E_t) \ge 2\tau \cdot T^*$. Now using Claim 4.4, we have $\tau \cdot T^* \le \Phi^*$, and hence $\mathsf{Opt}(\cup_{t\le\tau} E_t) \le 2 \cdot \Phi^*$. Finally using Claim 4.5 again, we obtain $c(F_\tau) \le 2\alpha_{\mathsf{on}} \cdot \Phi^*$. □

Claim 4.3 and Claim 4.6 imply that the above algorithm is a $(2\alpha_{\mathsf{on}},\ 0,\ 2\alpha_{\mathsf{mm}}\alpha_{\mathsf{on}}\alpha_{\mathsf{off}})$-discriminating algorithm for the robust problem $\mathsf{Robust}(\Pi) = \langle E, c, \{R_i\}_{i=1}^n, \Omega, \lambda\rangle$. Now using Lemma 2.2 we obtain Theorem 4.2.

Explicit uncertainty sets. An easy consequence of Theorem 4.2 is for the *explicit scenario* model of robust covering problems [Dhamdhere et al. 2005; Golovin et al. 2006], where Ω is specified as a list of possible scenarios. In this case, the MaxMin problem can be solved using the α_{off}-approximation algorithm from Property 2.5 which implies an $O(\alpha_{\mathsf{off}}^2\alpha_{\mathsf{on}})$-approximation for the robust version. In fact, we can do slightly better—observing that in this case, the algorithm for second-stage augmentation is the same as the Max-Min algorithm, we obtain an $O(\alpha_{\mathsf{off}} \cdot \alpha_{\mathsf{on}})$-approximation algorithm for robust covering with explicit scenarios. As an application of this result, we obtain an $O(\log n)$ approximation for robust Steiner forest with explicit scenarios, which is the best result known for this problem.

5. ROBUST COVERING UNDER P-SYSTEM AND Q-KNAPSACK UNCERTAINTY SETS

Recall that any uncertainty set Ω for a robust covering problem can be assumed, without loss of generality, to be *downward-closed*, i.e. $X \in \Omega$ and $Y \subseteq X$ implies $Y \in \Omega$. Eg., in the k-robust model [Feige et al. 2007], $\Omega = \{S \subseteq [n] : |S| \le k\}$. Hence it is of interest to obtain good approximation algorithms for robust covering when Ω is specified by means of general models for downward-closed families. In this section, we consider the two well-studied models of p-systems and q-knapsacks (Definitions 2.7 and 2.8).

The result of this section says the following: *if we can solve both the offline and online versions of a covering problem well, we get good algorithms for* Robust(Π) *under uncertainty sets given by the intersection of p-systems and q-knapsack constraints*. Naturally, the performance depends on p and q; we note that this is unavoidable due to complexity considerations. Based on Theorem 4.2 it suffices to give an approximation algorithm for the max-min problem under p-systems and q-knapsack constraints; so Theorem 3.6 combined with Theorem 4.2 implies an $O\left((p+1)(q+1)\,\alpha_{\mathsf{on}}^2\,\alpha_{\mathsf{off}}^2\right)$-approximation ratio. However, we can obtain a better guarantee by considering the algorithm for Robust(Π) directly. Formally we show that:

THEOREM 5.1. *Under Properties 2.4, 2.5 and 2.6, the robust covering problem* $\mathsf{Robust}(\Pi)\langle E, c, \{\mathcal{R}_i\}_{i=1}^m, \Omega, \lambda\rangle$ *admits an* $O\left((p+1)\cdot(q+1)\cdot\alpha_{\mathsf{off}}\cdot\alpha_{\mathsf{on}}\right)$*-approximation guarantee when* Ω *is given by the intersection of a* p*-system and* q*-knapsack constraints.*

The outline of the proof is same as for Theorem 3.6. We first consider the case when the uncertainty set is a p-system (Subsection 5.1); then using the reduction in Lemma 3.3 we solve a suitable instance of $\mathsf{Robust}(\Pi)$ under a $(p+1)$-system uncertainty set.

5.1. p-System Uncertainty Sets

In this subsection, we consider $\mathsf{Robust}(\Pi)$ when the uncertainty set Ω is some p-system. The algorithm is a combination of the ones in Theorem 4.2 and Theorem 3.1. We start with an empty solution, and use the online algorithm to greedily try and build a scenario of large cost. If we do find a scenario which has high cost then we augment our current solution to cover this scenario (again using the online algorithm), and continue. The algorithm is given as Algorithm 3 below.

ALGORITHM 3: Algorithm Robust-with-p-system-Uncertainty-Sets

Input: $\mathsf{Robust}(\Pi)$ instance and bound T.
let counter $t \leftarrow 0$, initial online algorithm's input $\sigma = \langle\rangle$, initial online solution $F_0 \leftarrow \emptyset$;
repeat
  set $t \leftarrow t+1$;
  let current scenario $A_0^t \leftarrow \emptyset$, counter $i \leftarrow 0$;
  while *(*$\exists e \in [n] \setminus A_i^t$ *such that* $A_i^t \cup \{e\} \in \Omega$*)* **do**
    $a_{i+1} \leftarrow \arg\max\{c(\mathcal{A}_{on}(\sigma \circ e)) - c(\mathcal{A}_{on}(\sigma)) \mid e \in [n] \setminus A_i^t \text{ and } A_i^t \cup \{e\} \in \Omega\}$;
    let $\sigma \leftarrow \sigma \circ a_{i+1}$, $A_{i+1}^t \leftarrow A_i^t \cup \{a_{i+1}\}, i \leftarrow i+1$;
  end
  let $E_t \leftarrow A_i^t$ be the scenario constructed by the above loop;
  let $F_t \leftarrow \mathcal{A}_{on}(\sigma)$ be the current online solution;
until $c(F_t) - c(F_{t-1}) \le 2\alpha_{\mathsf{on}} \cdot T$;
set $\tau \leftarrow t-1$;
Output: first-stage solution $\Phi_T := F_\tau$, and second-stage solution $\mathsf{Augment}_T$ where for any $\omega \subseteq [n]$, $\mathsf{Augment}_T(\omega)$ is the solution of the offline algorithm (Property 2.5) for the problem $\mathsf{OptAug}(\omega \mid \Phi_T)$.

We first prove a useful lemma about the behavior of the **while** loop.

LEMMA 5.2 (MAX-MIN LEMMA). *For any iteration* t *of the* **repeat** *loop, the scenario* $E_t \in \Omega$ *has the property that for any other scenario* $B \in \Omega$, $\mathsf{OptAug}(B \mid F_{t-1}) \le (p+1)\cdot c(F_t \setminus F_{t-1})$.

PROOF. The proof is almost identical to that of Theorem 3.1.

Consider any iteration t of the **repeat** loop in Algorithm 3 that starts with a sequence σ of elements (that have been fed to the online algorithm $\mathcal{A}_{on}$). Let $A = \{a_1, \cdots, a_k\}$ be the ordered set of elements added by the algorithm in this iteration. Define $G_0 := \mathcal{A}_{on}(\sigma)$, and $G_i := \mathcal{A}_{on}(\sigma \circ a_1 \cdots a_i)$ for each $i \in [k]$. Note that $F_{t-1} = G_0$ and $F_t = G_k$, and $G_0 \subseteq G_1 \subseteq \cdots \subseteq G_k$. It suffices to show that $\mathsf{OptAug}(B \mid G_0) \le (p+1)\cdot c(G_k \setminus G_0)$ for every $B \in \Omega$. But this is precisely Equation (3.1) from the proof of Theorem 3.1. □

COROLLARY 5.3 (SECOND STAGE). *For any* $T \ge 0$ *and* $B \in \Omega$,
- *elements* $\Phi_T \bigcup \mathsf{Augment}_T(B)$ *satisfy all the requirements in* B*, and*
- $c(\mathsf{Augment}_T(B)) \le 2\alpha_{\mathsf{off}} \cdot \alpha_{\mathsf{on}} \cdot (p+1) \cdot T$.

PROOF. This is identical to Claim 4.3. Observe that $\Phi_T = F_\tau = \mathcal{A}_{on}(\sigma)$, so the first part of the corollary follows from the definition of $\mathsf{Augment}_T$. By Lemma 5.2 and the termination condition, we have $\mathsf{OptAug}(B \mid F_\tau) \le (p+1) \cdot (c(F_{\tau+1}) - c(F_\tau)) \le 2(p+1)\alpha_{\mathsf{on}}\, T$. Now Property 2.5 guarantees that the solution $\mathsf{Augment}_T(B)$ found by this approximation algorithm has cost at most $2\alpha_{\mathsf{off}} \cdot \alpha_{\mathsf{on}} \cdot (p+1)\, T$. □

It just remains to bound the cost of the first-stage solution F_τ. Below Φ^* denotes the optimal first-stage solution (and its cost); and T^* is the optimal second-stage cost (as in Definition 2.1). So the optimal cost is $\Phi^* + \lambda \cdot T^*$.

LEMMA 5.4 (FIRST STAGE). *If $T \ge T^*$ then $c(\Phi_T) = c(F_\tau) \le 2\alpha_{\mathsf{on}} \cdot \Phi^*$.*

PROOF. This is identical to Claim 4.6. For any set $X \subseteq [n]$ of requirements, recall that $\mathsf{Opt}(X)$ denotes the minimum cost to satisfy X. Firstly, observe that $\mathsf{Opt}(\cup_{t \le \tau} E_t) \le \tau \cdot T^* + \Phi^*$. This follows from the fact that each of the τ scenarios E_t are in Ω, so the bound on the second-stage optimal cost gives $\mathsf{OptAug}(E_t \mid \Phi^*) \le T^*$ for all $t \le \tau$. By subadditivity (Property 2.4) we have $\mathsf{OptAug}(\cup_{t \le \tau} E_t \mid \Phi^*) \le \tau \cdot T^*$, which immediately implies the inequality. Now, we claim that

$$\mathsf{Opt}(\cup_{t \le \tau} E_t) \ge \tfrac{1}{\alpha_{\mathsf{on}}} \cdot c(F_\tau) \ge \tfrac{1}{\alpha_{\mathsf{on}}} \cdot 2\alpha_{\mathsf{on}} \tau \cdot T^* = 2\tau \cdot T^*. \tag{5.5}$$

The first inequality follows directly from the competitiveness of the online algorithm in Property 2.6. For the second inequality, we have $c(F_\tau) = \sum_{t=1}^{\tau} [c(F_t) - c(F_{t-1})] > 2\alpha_{\mathsf{on}}\tau \cdot T \ge 2\alpha_{\mathsf{on}}\tau \cdot T^*$ by the termination condition. Putting the upper and lower bounds on $\mathsf{Opt}(\cup_{t \le \tau} E_t)$ together, we have $\tau \cdot T^* \le \Phi^*$, and hence $\mathsf{Opt}(\cup_{t \le \tau} E_t) \le 2 \cdot \Phi^*$. Using the competitiveness of the online algorithm again, we obtain $c(F_\tau) \le 2\alpha_{\mathsf{on}} \cdot \Phi^*$. □

From Corollary 5.3 and Lemma 5.4, it follows that Algorithm 3 is $(2\alpha_{\mathsf{on}}, 0, 2\alpha_{\mathsf{off}}\, \alpha_{\mathsf{on}} \cdot (p+1))$-discriminating (Definition 2.1) to $\mathsf{Robust}(\Pi)$. Thus we obtain Theorem 5.1 for the case $q = 0$.

5.2. Algorithm for p-Systems *and* q-Knapsacks

Here we consider $\mathsf{Robust}(\Pi)$ when the uncertainty set Ω is the intersection of p-system $\mathcal{M}$ and a q-knapsack. The algorithm is similar to that in Subsection 3.3. Again, by scaling weights in the knapsack constraints, we may assume, without loss of generality, that each knapsack has capacity exactly one; let $w^1, \cdots, w^q$ denote the weight functions in the q knapsack constraints. We also assume, without loss of generality, that each singleton element satisfies the q-knapsack. The algorithm for $\mathsf{Robust}(\Pi)$ under Ω works as follows.

(1) Consider a modified uncertainty set that is given by the intersection of $\mathcal{M}$ and the *single knapsack* with weight function $\sum_{j=1}^{q} w^j$ and capacity q.
(2) Applying the algorithm in Lemma 3.3 to this single knapsack with $\epsilon = 1$, let $\{\mathcal{I}_j\}_{j=1}^{L}$ denote the independent sets of the resulting partition matroids. Note $L = n^{O(1)}$.
(3) For each $j \in [L]$, define uncertainty-set $\Sigma_j := \mathcal{M} \bigcap \mathcal{I}_j$.
(4) Let $\Sigma \leftarrow \cup_{j=1}^{L} \Sigma_j$. Solve $\mathsf{Robust}(\Pi)$ under Σ using the algorithm of Theorem 5.6.

As in the algorithm from Subsection 3.3, observe that here too each Σ_j is a $(p+1)$-system. Recall Claims 3.4 and 3.5 which hold here as well.

LEMMA 5.5. *Any α-approximate solution to $\mathsf{Robust}(\Pi)$ under uncertainty set Σ is a $(3q\alpha)$-approximate solution to $\mathsf{Robust}(\Pi)$ under uncertainty-set Ω.*

PROOF. As before, let Φ^* denote the optimal first-stage solution to $\mathsf{Robust}(\Pi)$ under Ω (and its cost), and the T^* denote the optimal second-stage cost. The optimal value $\mathsf{Opt} = \Phi^* + \lambda \cdot T^*$. Let $\tau \in \Sigma$ be any scenario, with partition $\{\omega_i\}_{i=1}^{3q}$ given by Claim 3.5.

ALGORITHM 4: Modification to Algorithm 3 for unions of p-systems.

set $t \leftarrow t+1$;
for *(*$j \in [L]$*)* **do**
  let current scenario $A_j \leftarrow \emptyset$;
  while *(*$\exists e \in [n] \setminus A_j$ *such that* $A_j \cup \{e\} \in \Sigma_j$*)* **do**
    $e^* \leftarrow \arg\max\{c(\mathcal{A}_{on}(\sigma \circ A_j \circ e)) - c(\mathcal{A}_{on}(\sigma \circ A_j)) \mid e \in [n] \setminus A_j \text{ and } A_j \cup \{e\} \in \Sigma_j\}$;
    $A_j \leftarrow A_j \cup \{e^*\}$;
  end
  Let $\Delta_j \leftarrow c(\mathcal{A}_{on}(\sigma \circ A_j)) - c(\mathcal{A}_{on}(\sigma))$;
end
let $j^* \leftarrow \arg\max\{\Delta_j \mid j \in [L]\}$, and $E_t \leftarrow A_{j^*}$;
let $\sigma \leftarrow \sigma \circ E_t$ and $F_t \leftarrow \mathcal{A}_{on}(\sigma)$ be the current online solution ;

Using the subadditivity Property 2.4, we have $\mathsf{OptAug}(\tau|\Phi^*) \leq \sum_{\ell=1}^{3q} \mathsf{OptAug}(\omega_\ell|\Phi^*) \leq (3q) \cdot T^*$. Thus, using Φ^* as the first stage solution to Robust(Π) under uncertainty-set Σ, results in an objective value at most $\Phi^* + \lambda \cdot (3q)\, T^* \leq 3q \cdot \mathsf{Opt}$. In particular, the optimal value of Robust(Π) under Σ is at most $3q\,\mathsf{Opt}$.

Finally Claim 3.4 implies that $\Omega \subseteq \Sigma$; so any solution to Robust(Π) under Σ is also a solution to Robust(Π) under Ω, where the objective value may only decrease. Thus the lemma follows. □

For solving Robust(Π) under Σ, note that although Σ itself is not any p'-system, it is the *union of polynomially-many* $(p+1)$-systems. We show below that a simple extension of the algorithm in Subsection 5.1 also works for unions of p-systems; this would solve Robust(Π) under Σ.

THEOREM 5.6. *There is an* $O((p+1)\,\alpha_{\mathsf{off}}\,\alpha_{\mathsf{on}})$*-approximation algorithm for* Robust(Π) *when the uncertainty set is the union of polynomially many* p*-systems.*

PROOF. Let $\Sigma = \cup_{j=1}^{L} \Sigma_j$ denote the uncertainty set where each Σ_j is a p-system. The algorithm for Robust(Π) under Σ is just Algorithm 3 where we replace the body of the repeat-loop by Algorithm 4.

Fix any iteration t of the repeat loop. By Lemma 5.2 applied to each p-system Σ_j,

CLAIM 5.7. *For each* $j \in [L]$*, we have* $\mathsf{OptAug}(B|F_{t-1}) \leq (p+1) \cdot \Delta_j$ *for every* $B \in \Sigma_j$.

By the choice of scenario E_t and since $\Sigma = \bigcup_{j=1}^{L} \Sigma_j$, we obtain:

CLAIM 5.8. *For any iteration* t *of the repeat loop and any* $B \in \Sigma$*,* $\mathsf{OptAug}(B|F_{t-1}) \leq (p+1) \cdot c(F_t \setminus F_{t-1})$.

Based on these claims and proofs identical to Corollary 5.3 and Lemma 5.4, we obtain the same bounds on the first and second stage costs of the final solution F_τ. Thus our algorithm is $(2\alpha_{\mathsf{on}}, 0, 2\alpha_{\mathsf{off}}\,\alpha_{\mathsf{on}} \cdot (p+1))$-discriminating, which by Lemma 2.2 implies the theorem. □

Finally, combining Lemma 5.5 and Theorem 5.6 we obtain Theorem 5.1.

Remark: In Theorem 5.1, the dependence on the number of constraints describing the uncertainty set Ω is necessary (under some complexity assumptions). We show that the independent set problem on graphs is a special case of the robust-covering problem (with large p or q). Recall that in the *independent set* problem, we are given an undirected graph G on n vertices with edge set $F \subseteq \binom{[n]}{2}$, and the objective is to find a maximum cardinality subset $S \subseteq [n]$ such that no edge has both end-points in S. Consider a very special case of the robust covering problem on ground-set $E :=$

$\{e_i\}_{i=1}^n$ and requirements $[n]$. The requirement $i \in [n]$ is satisfied if and only if the solution contains e_i. The cost function c on E is all ones, and the inflation parameter $\lambda = 1$. The uncertainty set Ω is given by the intersection of $p = |F|$ different cardinality constraints, where for each edge $(u, v) \in F$ there is a constraint that "at most one of the requirements $\{u, v\}$ can appear". Notice that Ω is precisely the set of independent sets in graph G. In this case, the optimal value of the robust covering problem (which is also the optimal max-min value) is exactly the optimal value of the independent set instance G. The hardness result from [Håstad 1999] now implies that the objective value of this robust covering problem is $\Omega(p^{\frac{1}{2}-\epsilon})$ hard to approximate. We note that this hardness applies only to algorithms having running time that is polynomial in both $|E|$ and p; this is indeed the case for our algorithm.

Results for p-System and q-Knapsack Uncertainty Sets. We now list some specific results for robust covering under uncertainty sets described by p-systems and knapsack constraints; these follow directly from Theorem 5.1 using known offline and (deterministic) online algorithms for the relevant problems.

Problem	Offline ratio	Online ratio	Robust under p-system, q-knapsack
Set Cover	$O(\log m)$	$O(\log m \cdot \log n)$	$O(pq \cdot \log^2 m \cdot \log n)$
Steiner Tree Steiner Forest	2	$O(\log n)$	$O(pq \cdot \log n)$
Minimum Cut	1	$O(\log^3 n \cdot \log\log n)$	$O(pq \cdot \log^3 n \cdot \log\log n)$
Multicut	$O(\log n)$	$O(\log^3 n \cdot \log\log n)$	$O(pq \cdot \log^4 n \cdot \log\log n)$

The offline algorithms are: Steiner forest [Agrawal et al. 1995; Goemans and Williamson 1995] and multicut [Garg et al. 1996]. The online algorithms are: set cover [Alon et al. 2009], Steiner tree/forest [Imase and Waxman 1991; Berman and Coulston 1997], and min-cut/multicut [Alon et al. 2006; Harrelson et al. 2003].

6. NON-SUBMODULARITY OF SOME COVERING FUNCTIONS

In this section we show that some natural covering functions cannot be approximated well (pointwise) by any submodular function. Formally, for $f : 2^U \to \mathbb{R}_{\geq 0}$ being a monotone subadditive function, we define f to be *α-approximately submodular* if there exists a submodular function $g : 2^U \to \mathbb{R}_{\geq 0}$ with $g(S) \leq f(S) \leq \alpha \cdot g(S)$ for all $S \subseteq U$.

Recall the definition of a min-cost covering function from Section 2. We use $[n]$ to denote the set of covering requirements. Given a set system with universe $[n]$ and collection $\mathcal{C}$ of subsets of $[n]$, define the *min-set-cover function* f_{SC} as

$$f_{SC}(S) := \text{ minimum number of subsets from } \mathcal{C} \text{ required to cover } S, \qquad \forall S \subseteq [n].$$

Similarly, given an undirected graph with $[n]$ denoting a collection of terminal-pairs, define the *min-multicut function* f_{MMC} which maps subset $S \subseteq [n]$ to the minimum number of edges whose deletion separates each pair of terminals in S.

PROPOSITION 6.1 ([IMMORLICA ET AL. 2008]). *The min-set-cover and min-multicut functions are not $o(n^{1/4})$-approximately submodular, where n is the number of covering requirements.*

PROOF. We show this result for the vertex cover problem, which is a special case of set-cover as well as multicut (even on a star graph). The proof follows as a corollary of

a result from [Immorlica et al. 2008] which gives a lower bound on the *budget-balance* for *cross-monotone cost allocations* (the reader is not required to know the definition of these terms). For the sake of completeness, we give a direct proof of the vertex-cover covering function not being approximately submodular by adapting the proof from [Immorlica et al. 2008].

(For readers familiar with the notions of budget-balance and cross-monotonicity, here is a shorter proof. [Immorlica et al. 2008] showed that there is no sub-polynomial approximately-budget-balanced cross-monotone cost allocation for the vertex-cover game. On the other hand it is known (see Chapter 15.4.1 in [Nisan et al. 2007]) that any submodular-cost game admits a budget-balanced cross-monotone cost allocation. This also implies that any α-approximately submodular cost function admits an α-approximate budget-balanced cross-monotone cost allocation. Thus the vertex-cover covering function is not approximately submodular, the precise bound following from [Immorlica et al. 2008].)

Now for the self-contained proof. Recall the vertex cover problem where, given an undirected graph, the goal is to find a minimum cardinality set of vertices that covers all edges (an edge is covered if either of its end-points is chosen). Consider the vertex-cover covering function f_{VC} on a t-vertex complete graph, which is defined as follows. The covering requirements are the $n := \binom{t}{2}$ edges of the complete graph. For any subset S of edges, define function $f_{VC}(S)$ to be the minimum cardinality of a vertex cover for the edges S. Choose parameter m and set $\ell := m^2$ and $t := m + 2\ell$.

Suppose that g is any submodular function satisfying $g(S) \leq f_{VC}(S) \leq \alpha \cdot g(S)$ for all subsets S of edges. We will show that $\alpha \geq \frac{m}{3}$. Below, for any subsets X and Y, we denote $g_X(Y) = g(X \cup Y) - g(X)$. By submodularity,

$$g_X(Y \cup Z) \quad \leq \quad g_X(Y) + g_X(Z), \qquad \forall\ X, Y, Z \subseteq [n],\ Y \cap Z = \emptyset. \tag{6.6}$$

$$g_X(Z) \quad \leq \quad g_Y(Z), \qquad \forall\ Y \subseteq X \subseteq [n],\ Z \subseteq [n] \setminus X. \tag{6.7}$$

Consider the following random process. Pick a permutation π of $[t]$ uniformly at random. Let A denote the first m vertices in π, B the set of the next ℓ vertices, and C the remaining ℓ vertices. Let $b_1, b_2, \ldots, b_\ell$ denote the ordered list of vertices in B; similarly $c_1, c_2, \ldots, c_\ell$ the vertices in C. Define $T = \{(a, b) : a \in A,\ b \in B\}$, edges $e_i = (b_i, c_i)$ for all $i \in [\ell]$, and $M = \{e_1, e_2, \ldots, e_\ell\}$. Note that T and M are random sets.

By (6.6) we have $g_T(M) \leq \sum_{i=1}^{\ell} g_T(\{e_i\})$ for every possible T and M. Also,

$$g_T(M) \quad = \quad g(T \cup M) - g(T) \quad \geq \quad \frac{f_{VC}(T \cup M)}{\alpha} - f_{VC}(T) \quad \geq \quad \frac{\ell}{\alpha} - m \tag{6.8}$$

The first inequality uses the fact that g is an α-approximation to f_{VC}. The second inequality follows from (i) $f_{VC}(T) \leq m$, since A is a feasible vertex cover, and (ii) $f_{VC}(T \cup M) \geq \ell$ since M is a matching of size ℓ.

Next we upper bound the expectation $\mathbb{E}[\sum_{i=1}^{\ell} g_T(\{e_i\})]$. First consider any term $g_T(\{e_i\})$ which by (6.7) is at most $g_{T_i}(\{e_i\})$, where $T_i = \{(a, b_i) : a \in A\} \subseteq T$. Notice that by symmetry, $\langle T_i, e_i \rangle$ has an identical distribution for all $i \in [\ell]$, which is the following:

> Pick $b \in [t]$ and $A' \subseteq [t] \setminus \{b\}$ of size $m + 1$ randomly; then choose $c \in A'$ randomly. Set $T_i = \{(a, b) : a \in A' \setminus \{c\}\}$ and $e_i = (b, c)$.

So $\mathbb{E}[\sum_{i=1}^{\ell} g_{T_i}(\{e_i\})] = \ell \cdot \mathbb{E}[g_{T_1}(\{e_1\})]$. Call a subset $R \subseteq [n]$ of $m+1$ edges an $(m+1)$-star if all edges in R have a common end point; in particular, the vertex cover value $f_{VC}(R) = 1$. Let N denote the number of $(m+1)$-stars. Note that for any $(m+1)$-star R and any edge $f \in R$, we have $\Pr[T_1 = (R \setminus \{f\}) \wedge e_1 = f] = p := \frac{1}{(m+1)N}$. For a ***fixed***

$(m+1)$-star R, grouping those terms in $\mathbb{E}[g_{T_1}(\{e_1\})]$ that have $T_1 \cup \{e_1\} = R$ gives us:

$$p \cdot \sum_{f \in R} g_{R \setminus f}(\{f\}) \quad \leq \quad p \cdot g(R) \quad \leq \quad p \tag{6.9}$$

The second inequality is due to $g(R) \leq f_{VC}(R) = 1$ since R is a star. To see the first inequality, let $R = \{f_1, \ldots, f_{m+1}\}$; then by (6.7),

$$\sum_{f \in R} g_{R \setminus f}(\{f\}) \quad \leq \quad \sum_{j=0}^{m} g_{\{f_1,\ldots,f_j\}}(\{f_{j+1}\}) \quad = \quad g(R).$$

Using (6.9) we obtain:

$$\mathbb{E}\left[\sum_{i=1}^{\ell} g_{T_i}(\{e_i\})\right] \;=\; \ell \cdot \mathbb{E}[g_{T_1}(\{e_1\})] \;=\; \ell \cdot \sum_{R:\text{(m+1)-star}} p \cdot \sum_{f \in R} g_{R \setminus f}(\{f\}) \;\leq\; \ell \cdot p \cdot N \;=\; \frac{\ell}{m+1}$$

Recall that by (6.6) we have:

$$\mathbb{E}\left[\sum_{i=1}^{\ell} g_{T_i}(\{e_i\})\right] \quad \geq \quad \mathbb{E}[g_T(M)] \quad \geq \quad \frac{\ell}{\alpha} - m,$$

the last inequality uses (6.8) which holds for every outcome. Combining the above two inequalities, it follows that $\alpha \geq \frac{\ell m + \ell}{\ell + m^2 + m} \geq m/3$. Recall that the number of covering requirements in the vertex cover instance $n = O(t^2) = O(m^4)$; so $\alpha = \Omega(n^{1/4})$. Finally, since vertex cover is a special case of both set-cover and multicut, the claim follows. □

On the other hand, some other covering functions we considered are indeed approximately submodular.

— The minimum-cut function ($f_{MC}(S) =$ minimum cost cut separating vertices S from a fixed root) is in fact submodular due to submodularity of cuts in graphs.
— The min-Steiner-tree ($f_{ST}(S) =$ minimum length tree that connects vertices S to a fixed root) and min-Steiner-forest ($f_{SF}(S) =$ minimum length forest connecting the terminal-pairs in S) functions are $O(\log n)$-approximately submodular. When the underlying metric is a tree, these functions are submodular—in this case they reduce to weighted coverage functions. Using probabilistic approximation of general metrics by trees, we can write $g(S) = E_{T \in \mathcal{T}}[f^T(S)]$ where $\mathcal{T}$ is the distribution on dominating tree-metrics (from [Fakcharoenphol et al. 2004]) and f^T is the Steiner-tree/Steiner-forest function on tree T. Clearly g is submodular. Since there exists $\mathcal{T}$ that approximates distances in the original metric within factor $O(\log n)$ [Fakcharoenphol et al. 2004], it follows that g also $O(\log n)$-approximates f_{ST} (resp. f_{SF}).

While approximate submodularity of the covering problem Π (eg. minimum-cut or Steiner-tree) yields direct approximation algorithms for MaxMin(Π), it is unclear whether they help in solving Robust(Π) (even under cardinality-constrained uncertainty sets [Gupta et al. 2014]). On the other hand, the online-algorithms based approach in this paper solves both MaxMin(Π) and Robust(Π), for a large class of uncertainty sets: arising from p-systems and q-knapsacks.

REFERENCES

A. Ben-Tal, L.El Ghaoui, and A. Nemirovski. 2009. *Robust Optimization*. Princeton University Press.

Ajit Agrawal, Philip Klein, and R. Ravi. 1995. When trees collide : An approximation algorithm for the generalized Steiner problem on networks. *SIAM J. Comput.* 24(3) (1995), 445–456.

Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. 2006. A general approach to online network optimization problems. *ACM Transactions on Algorithms* 2, 4 (2006), 640–660.

Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. 2009. The Online Set Cover Problem. *SIAM J. Comput.* 39, 2 (2009), 361–370.

Piotr Berman. 2000. A $d/2$ approximation for maximum weight independent set in d-claw free graphs. *Nordic J. of Computing* 7, 3 (2000), 178–184.

Piotr Berman and Chris Coulston. 1997. On-line algorithms for Steiner tree problems. In *Proceedings of the Symposium on Theory of Computing (STOC)*. 344–353.

Dimitris Bertsimas, David B. Brown, and Constantine Caramanis. 2011. Theory and Applications of Robust Optimization. *SIAM Rev.* 53, 3 (2011), 464–501.

Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. 2011. Maximizing a Monotone Submodular Function Subject to a Matroid Constraint. *SIAM J. Comput.* 40, 6 (2011), 1740–1766.

Chandra Chekuri and Sanjeev Khanna. 2005. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM J. Comput.* 35, 3 (2005), 713–728.

Kedar Dhamdhere, Vineet Goyal, R. Ravi, and Mohit Singh. 2005. How to Pay, Come What May: Approximation Algorithms for Demand-Robust Covering Problems. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 367–378.

Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. 2004. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. System Sci.* 69, 3 (2004), 485–497.

Uriel Feige, Kamal Jain, Mohammad Mahdian, and Vahab S. Mirrokni. 2007. Robust Combinatorial Optimization with Exponential Scenarios. In *Proceedings of the Integer Programming and Combinatorial Optimization Conference (IPCO)*. 439–453.

M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. 1978. An analysis of approximations for maximizing submodular set functions II. *Mathematical Programming Study* 8 (1978), 73–87.

A.M. Frieze and M.R.B. Clarke. 1984. Approximation algorithms for the m-dimensional 0-1 knapsack problem: worst case and probabilistic analyses. *European Journal of Operational Research* 15, 1 (1984), 100–109.

Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. 1996. Approximate Max-Flow Min-(Multi)Cut Theorems and Their Applications. *SIAM J. Comput.* 25(2) (1996), 235–251.

Michel X. Goemans and David P. Williamson. 1995. A General Approximation Technique for Constrained Forest Problems. *SIAM J. Comput.* 24, 2 (1995), 296–317.

Daniel Golovin, Vineet Goyal, and R. Ravi. 2006. Pay today for a rainy day: improved approximation algorithms for demand-robust min-cut and shortest path problems. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS)*. Lecture Notes in Comput. Sci., Vol. 3884. Springer, Berlin, 206–217.

Anupam Gupta, Viswanath Nagarajan, and R. Ravi. 2014. Thresholded covering algorithms for robust and max-min optimization. *Math. Program.* 146, 1-2 (2014), 583–615.

Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. 2010. Constrained Non-monotone Submodular Maximization: Offline and Secretary Algorithms. In *Proceedings of the Workshop on Internet and Network Economics (WINE)*. 246–257.

Chris Harrelson, Kirsten Hildrum, and Satish Rao. 2003. A polynomial-time tree decomposition to minimize congestion. In *Proceedings of the Symposium on Parallelism in Algorithms and Architectures (SPAA)*. 34–43.

J. Håstad. 1999. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica* 182 (1999), 105–142.

Cor A. J. Hurkens and Alexander Schrijver. 1989. On the Size of Systems of Sets Every t of Which Have an SDR, with an Application to the Worst-Case Ratio of Heuristics for Packing Problems. *SIAM J. Discrete Math.* 2, 1 (1989), 68–72.

M. Imase and B.M. Waxman. 1991. Dynamic Steiner tree problem. *SIAM J. on Discrete Mathematics* 4(3) (1991), 369–384.

Nicole Immorlica, Mohammad Mahdian, and Vahab S. Mirrokni. 2008. Limitations of cross-monotonic cost-sharing schemes. *ACM Trans. Algorithms* 4, 2 (2008), 1–25.

T. A. Jenkyns. 1976. The efficiency of the "greedy" algorithm. In *7th South Eastern Conference on Combinatorics, Graph Theory and Computing*. 341–350.

Rohit Khandekar, Guy Kortsarz, Vahab S. Mirrokni, and Mohammad R. Salavatipour. 2013. Two-stage Robust Network Design with Exponential Scenarios. *Algorithmica* 65, 2 (2013), 391–408.

Ariel Kulik, Hadas Shachnai, and Tami Tamir. 2013. Approximations for Monotone and Nonmonotone Submodular Maximization with Knapsack Constraints. *Math. Oper. Res.* 38, 4 (2013), 729–739.

Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. 2010a. Maximizing Nonmonotone Submodular Functions under Matroid or Knapsack Constraints. *SIAM J. Discrete Math.* 23, 4 (2010), 2053–2078.

Jon Lee, Maxim Sviridenko, and Jan Vondrák. 2010b. Submodular Maximization over Multiple Matroids via Generalized Exchange Properties. *Math. Oper. Res.* 35, 4 (2010), 795–806.

G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming* 14 (1978), 265–294.

Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. 2007. *Algorithmic Game Theory*. Cambridge University Press.

A. Schrijver. 2003. *Combinatorial Optimization*. Springer.

C. Seshadhri and Jan Vondrák. 2011. Is Submodularity Testable?. In *Proceedings of the Innovations in Computer Science*. 195–210.

D. Shmoys and C. Swamy. 2004. Stochastic Optimization is (almost) as Easy as Deterministic Optimization. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 228–237.

Aravind Srinivasan. 1999. Improved Approximation Guarantees for Packing and Covering Integer Programs. *SIAM J. Comput.* 29, 2 (1999), 648–670.

M. Sviridenko. 2004. A note on maximizing a submodular set function subject to knapsack constraint. *Operations Research Letters* 32 (2004), 41–43.

J. Vondrák. 2008. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the Symposium on Theory of Computing (STOC)*. 67–74.