

Robust Combinatorial Optimization with Exponential Scenarios

Uriel Feige¹, Kamal Jain¹, Mohammad Mahdian², and Vahab Mirrokni¹

¹ Microsoft Research, {urifeige,kjain,mirrokn}@microsoft.com

² Yahoo! Research, mahdian@yahoo-inc.com

Abstract. Following the well-studied two-stage optimization framework for stochastic optimization [15, 18], we study approximation algorithms for robust two-stage optimization problems with an exponential number of scenarios. Prior to this work, Dhamdhere et al. [8] introduced approximation algorithms for two-stage robust optimization problems with explicitly given scenarios. In this paper, we assume the set of possible scenarios is given implicitly, for example by an upper bound on the number of active clients. In two-stage robust optimization, we need to pre-purchase some resources in the first stage before the adversary’s action. In the second stage, after the adversary chooses the clients that need to be covered, we need to complement our solution by purchasing additional resources at an inflated price. The goal is to minimize the cost in the worst-case scenario. We give a general approach for solving such problems using LP rounding. Our approach uncovers an interesting connection between robust optimization and online competitive algorithms. We use this approach, together with known online algorithms, to develop approximation algorithms for several robust covering problems, such as set cover, vertex cover, and edge cover. We also study a simple *buy-at-once* algorithm that either covers all items in the first stage or does nothing in the first stage and waits to build the complete solution in the second stage. We show that this algorithm gives tight approximation factors for unweighted variants of these covering problems, but performs poorly for general weighted problems.

1 Introduction

In many combinatorial optimization problems, the objective is to minimize the cost of building an installation to serve a number of clients. In classical optimization problems, it is often assumed that the parameters of the system are known in advance. However, in reality, it is almost always impossible or costly to obtain accurate data about various parameters of the optimization problem at the time of planning. For example, the cost of acquiring a resource or the set of clients that need to be serviced might be unknown. The goal of the fields of *stochastic optimization* and *robust optimization* is to provide algorithms for minimizing the cost in presence of uncertainty.

In *stochastic optimization* [6, 7], it is assumed that we have information about the probability distribution governing the data. Given this information, the goal is to plan ahead to minimize the expected cost. In particular, in *two-stage stochastic optimization*, a solution is built in two stages: in the first stage, we need to decide which resources to purchase given only distributional information about the instance. In the second stage, the exact information about the data is revealed and we are allowed to complement the solution built in the first stage by purchasing extra resources at an inflated cost.

Robust optimization [4, 5, 17, 3] can be considered the worst-case analogue of the stochastic optimization. In a robust optimization problem, we are given bounds on various parameters of the system, and the goal is to find a solution that minimizes the cost in a worst-case scenario (or be feasible in a worst-case scenario). A *two-stage robust optimization* problem (introduced in [3, 8]) is similar to a two-stage stochastic problem except instead of a distribution, we have a set of possible scenarios (given either explicitly, or implicitly by giving bounds on various parameters), and instead of expectation, we would like to minimize the maximum cost of the solution, where maximum is taken over the set of all possible scenarios.

During the past few years, stochastic optimization (and in particular, two-stage stochastic optimization) has received considerable attention from the perspective of approximation algorithms. Efficient approximation algorithms are given for a wide class of optimization problems, both for cases where the distribution is given explicitly by listing the set of all possible scenarios and the corresponding probabilities [18, 13], and in the more general case where the distribution is given implicitly, as the product of a number of independent trials, or by an oracle [15, 20, 12, 8].

For robust optimization, Ben-tal et al [3] initiated the study of two-stage robust optimization problems. Dhamdhere et al. [8] introduced the first approximation algorithms for two-stage robust covering problems when the set of scenarios is given *explicitly*. In this paper, we take on the task of studying approximation algorithms for two-stage robust optimization problems, where the set of possible scenarios is given *implicitly*. In particular, we focus on the case where the set of possible scenarios is given by an upper bound on the number of active clients, and give approximation algorithms for the robust version of several classical covering problems such as set cover, vertex cover, and edge cover.

1.1 The model

In this section we give a formal definition of the robust optimization model that will be studied in this paper. This model is a generalization of the model introduced by Dhamdhere et al. [8] (in the case of explicitly listed scenarios), and is motivated by similar models for two-stage stochastic optimization [15, 13].

In a *covering problem*, we have a set \mathcal{C} of potential *clients*, and a set \mathcal{R} of *resources*. Each resource $r \in \mathcal{R}$ can be *purchased* at a cost c_r . In order to serve

a set of clients, a set of resources must be purchased. The collection of all sets of resources which can serve the set $S \subset \mathcal{C}$ of clients is denoted by $sol(S)$. In covering problems the collection $sol(S)$ is an upper ideal, i.e., if a set of resources can serve S , so can any superset of this set. An *unweighted* covering problem is a covering problem in which all c_r 's are equal to 1.

Generally, the collection $sol(S)$ is given implicitly by specifying a set of constraints. Three examples that we will focus on in this paper are set cover, vertex cover, and edge cover. In the *set cover* problem, each resource $r \in \mathcal{R}$ corresponds to a set of clients, and the collection $sol(S)$ consists of all sets of resources whose union covers S . In the *vertex cover* problem, the set of clients and the set of resources are the edge set and the vertex set of a given graph, respectively, and a set $S \subset \mathcal{C}$ can be served by any set of vertices that contain at least one of the endpoints of each edge in S . In the *edge cover* problem, each resource is an edge and each client is a vertex of a given graph, and a set S of clients can be covered by any set of edges that has at least one edge adjacent to any vertex in S .

In a *two-stage robust covering problem*, we have a collection \mathcal{S} of scenarios, each given by a set of *active* clients (i.e., clients that need to be covered). The objective is to purchase a set of resources in the first stage to minimize the cost of these resources plus a given inflation factor λ times the maximum over scenarios S in \mathcal{S} , of the cost of completing the solution for scenario S . In other words, after we purchase a set of resources in the first stage, an adversary decides in which scenario we are. After that, we need to complete the solution by purchasing more resources at costs inflated by a factor λ (or more generally, by an inflation factors λ_r^S which depends on the resource $r \in \mathcal{R}$ and the scenario $S \in \mathcal{S}$).

The robust optimization problem can be studied in several different models, depending on how the list of scenarios \mathcal{S} is given to the algorithm. One model, studied by Dhamdhere et al. [8] and Golovin et al [11], is to assume that the list of all possible scenarios is given explicitly. This model is suitable for situations where the number of possible scenarios is not very large. An alternative model, motivated by the independent trials model of stochastic optimization, is to assume that the list of scenarios is given implicitly by an upper bound on the maximum number of active clients. More formally, in this model an integer k is given and \mathcal{S} is defined as $\{S \subseteq \mathcal{C} : |S| \leq k\}$.³ Finally, motivated by the oracle model in stochastic optimization, we define an oracle model for robust optimization where the list of possible scenarios is given by an oracle which, given the set of resources purchased in the first stage, outputs the worst-case (or approximately the worst-case) scenario for the second stage.

An important distinction between our oracle model and the oracle model for stochastic optimization is that in our model, the problem the oracle needs to

³ More generally, we can consider a model where the set of clients is partitioned into subsets $\mathcal{C}_1, \dots, \mathcal{C}_t$, and the set of scenarios is the collection of all sets that have at most k_i clients from the set \mathcal{C}_i . Although the results in Sections 2 and 3 work for this more general model, for clarity of exposition we restrict ourselves to the simpler model.

solve is often computationally intractable. For example, if the set of scenarios in a robust set cover problem is given by an upper bound on the number of active clients, the oracle needs to find a subset of k clients whose minimum cost of covering is maximized. We call this problem the *max-min set cover problem*, and will observe that it is computationally hard. Considering the hardness of the oracle problem, the algorithms designed for the oracle model need to be able to work with an *approximate oracle* as well. Furthermore, in order to solve a robust optimization problem in the model where the scenarios are given by an upper bound on the number of active clients, in addition to designing an algorithm for the oracle model, we need to give an approximation algorithm for the max-min version of the problem.

1.2 Our contribution

In this paper, we mostly focus on the model where the scenarios are given implicitly by an upper bound on the number of active scenarios. This is motivated by real-world situations where a good estimate of the total number of clients who will show up is available, but we do not exactly know where they will appear. We will also give a general LP-based algorithm for the oracle model, assuming that the oracle gives a good approximation of the worst-case scenario with respect to the fractional solution.

A naive idea to solve the robust optimization problems is a *buy-at-once* algorithm: either cover all items in the first stage in which case nothing needs to be done in the second stage. Or do nothing in the first stage and construct a solution in the second stage, after the adversary makes its choices. The choice of which of the two options to use is based on a polynomial-time test that is problem specific. We study this algorithm in Section 4 and prove that when the inflation factor is the same for all scenarios, the approximation ratio of this algorithm for robust unweighted set cover, vertex cover, and edge cover problems are $\max(\log m, \log n)$, 2, and 2, respectively. However, the following example shows that for the weighted version of robust vertex cover, any buy-at-once algorithm (even with unbounded computing power) performs poorly. Consider a clique on n vertices, with $k = 1$ and $\lambda = \sqrt{n}$. All vertices have weight 1, except for two vertices that have weight $w = \sqrt{n}$. The buy-at-once algorithm will either pay at least n in the first stage, or at least $\lambda w = n$ in the second stage. However, an optimal algorithm can choose only one of the heavy vertices in the first stage, and then pay at most $w + \lambda k = 2\sqrt{n}$. Hence the approximation ratio of the buy-at-once algorithm for weighted robust vertex cover is no better than $\Omega(\sqrt{n})$. This example indicates the need for more sophisticated approximation algorithms for robust two-stage optimization problems.

In Section 2, we give a general LP-based framework for solving robust covering problems given access to an oracle that solves the fractional *max-min problem* (or the adversary's problem) and another oracle that rounds the LP solution for the classical (i.e., non-robust) optimization problem. For example, for the robust set cover problem, we need an oracle that given an integer k and a collection

of subsets S_1, S_2, \dots, S_m of a universe F each with a cost $c(S_i)$, finds a subset $T \subseteq F$ of size at most k for which the cost of fractional set cover is maximized, and another oracle that rounds a fractional set cover to an integral one. In Section 3, we show how an online algorithm can be used to solve the max-min problem when the set of feasible scenarios are given by an upper bound on the number of active clients. We use this to give an $O(\log m)$ -approximation algorithm for max-min fractional set cover. We also show that the max-min fractional set cover problem is not approximable within a factor better than $\Omega(\frac{\log m}{\log \log m})$ under reasonable complexity assumptions. As a result of this framework, we get an $O(\log n \log m)$ -approximation for the robust set cover problem. Following similar ideas, we design constant-factor approximation algorithms for robust vertex cover and edge cover problems. This framework can be extended easily to more general settings in which the scenarios are given implicitly in more general ways. The main step for these extensions is to design good approximation algorithms for the max-min problems.

Finally, in Section 5 we show that our algorithms for max-min fractional set cover and edge cover achieve essentially the best possible approximation factors, assuming reasonable complexity assumptions.

2 An LP-rounding approach for robust set cover

In this section, we give an LP-based approach for robust set cover. Our techniques work for a more general covering problem where each resource $r \in \mathcal{R}$ can be picked an integer number of times x_r , and a client is covered if a corresponding inequality of the form $\sum_r a_{ir} x_r \geq 1$ (where a_{ir} are given non-negative coefficients) is satisfied. The details of this generalization are omitted here.

We start by giving an LP formulation of two-stage robust set cover.⁴

$$\text{minimize } Z + \sum_{r \in \mathcal{R}} c_r y_r^0 \quad (1)$$

$$\text{subject to } \forall S \in \mathcal{S}, \forall i \in S : \sum_{r: i \in r} (y_r^0 + y_r^S) \geq 1 \quad (2)$$

$$\forall S \in \mathcal{S} : \sum_{r \in \mathcal{R}} \lambda c_r y_r^S \leq Z. \quad (3)$$

The variable y_r^0 in the above LP indicates whether the resource r is purchased in the first stage. Similarly, the variable y_r^S indicates whether this resource is purchased in the second stage, if the adversary selects the set S as the set of active clients. The variable Z indicates the maximum cost of the second stage, where the maximum is taken over all possible scenarios. Clearly, if the variables

⁴ We present this LP in the case that the inflation factor λ does not depend on the resource r or the scenario S . However, it is easy to see that all proofs in this section apply to the more general case.

y_r^0 and y_r^S are restricted to be integers, the above integer program captures the robust set cover problem precisely. Therefore, relaxing the integrality condition gives us a linear program whose solution is a lower bound on the cost of the optimal solution to the robust set cover problem.

The main difficulty with this LP formulation is that it contains an exponential number of constraints *and* an exponential number of variables, and therefore cannot be solved directly using the ellipsoid method. We can deal with this problem using a technique developed by Shmoys and Swamy [20] for stochastic optimization: we consider the projection of the above LP onto the space corresponding to the variables y_r^0 's and Z , and then give a separation oracle for the reduced LP. The projection of the above LP corresponds to the following program.

$$\begin{aligned} & \text{minimize} && Z + \sum_{r \in \mathcal{R}} c_r y_r^0 && (P) \\ & \text{subject to} && \forall S \in \mathcal{S} : Z \geq \text{cost}_2(S, y^0) \end{aligned}$$

Here $\text{cost}_2(S, y^0)$ denotes the cost of the optimal fractional solution for the second stage when the set of active clients is S , given that resource r is already purchased to the extent of y_r^0 in the first stage.

The separation oracle for this LP corresponds to an algorithm that computes the optimal strategy for the adversary of the robust fractional set cover problem. We call this *the max-min fractional set cover problem*. More precisely, the max-min fractional set cover problem is the following: given a fractional first-stage solution (i.e., y_r^0 's), select a scenario (in the example we will focus on in this paper, a set of at most k clients) so that the cost of a fractional solution for the second stage is maximized. The following lemma, proved using a simple application of the ellipsoid method, shows that given an approximation algorithm for the max-min fractional problem, we can compute an approximate solution of the above LP in polynomial time.

Lemma 1. *Assume we have a polynomial time γ -approximation algorithm for the max-min fractional problem. Then, we can compute a γ -approximation to the solution of the linear program (P) in polynomial time.*

The proof, which is omitted here, is based on the ellipsoid algorithm and the techniques used by Shmoys and Swamy [20] in the context of stochastic optimization.

The above lemma requires us to be able to solve the max-min fractional set cover problem given a *fractional* first-stage solution. In other words, for each client i we are given a fractional value θ_i , so that if the adversary chooses i in the set of active clients, we will have to cover i to the extent of θ_i . In the following lemma, we show that it is enough to be able to solve the max-min problem given that θ_i 's are zero or one. In other words, given a subset C' of the clients (corresponding to those with $\theta_i = 1$), we need to be able to find a set of at most k

clients in C' whose minimum fractional covering cost is maximized. We call this problem the max-min fractional set cover problem with integer requirements.

Lemma 2. *Assume we have a polynomial time γ -approximation algorithm A for the max-min fractional set cover problem with integer requirements. Then, we can compute a $(\gamma+1)$ -approximation to the solution of the linear program (P) in polynomial time.*

Proof. We iteratively run the ellipsoid algorithm to check whether (P) has a solution with an objective function value of at most R , and use binary search to find the smallest value of R for which the ellipsoid algorithm declares that there is such a solution. For the separation oracle, we do the following: let $C' = \{i \in \mathcal{C} : \sum_{r:i \in r} y_r^0 < 1/(\gamma+1)\}$ denote the set of clients covered by the fractional first-stage solution to an extent less than $1/(\gamma+1)$. We run algorithm A to find the max-min fractional set cover among clients in C' . Let T denote the cost of the solution returned by A . This means that there is at least one scenario $S^* \in \mathcal{S}$ such that the cost of minimum fractional cover for $S^* \cap C'$ is at least T , and for every scenario $S \in \mathcal{S}$, the cost of the minimum fractional cover for $S \cap C'$ is at most γT . Our separation oracle accepts (y^0) if the cost of the first stage solution y^0 plus $\frac{\gamma}{\gamma+1}T$ is at most R ; otherwise it rejects (i.e., declares that there is no solution with a first stage solution of y^0 of total cost at most R).

First, we show that if the above separation oracle rejects y^0 , then an exact separation oracle would do the same. This is because with a first stage solution of y^0 , clients in C' need to be covered to the extent of at least $\frac{\gamma}{\gamma+1}$ in the second stage, and therefore the cost of the second stage in scenario S^* cannot be less than $\frac{\gamma}{\gamma+1}$ times the cost of the minimum fractional cover for $S^* \cap C'$, which, by definition, is at least T .

Next, we prove that if our separation oracle accepts y^0 , then we can build a feasible solution for (P) of cost at most $(\gamma+1)R$. This is done by multiplying y^0 by $(\gamma+1)$. The set of clients not covered by this inflated first stage solution is a subset of C' . Therefore, the cost of the second stage is at most γT . The overall cost of this solution is at most $(\gamma+1) \sum_{r \in \mathcal{R}} c_r y_r^0 + \gamma T \leq (\gamma+1)R$, where the latter inequality follows from the fact that our separation oracle accepts y^0 .

Now, let R^* be the smallest value of R for which our algorithm decides that the linear program has a solution. This means that for $R < R^*$, our separation oracle never accepts any first stage solution y^0 . By our first observation, the ellipsoid algorithm with an exact separation oracle would return the same answer. Hence, R^* is a lower bound on the solution of (P). Since the ellipsoid algorithm for $R = R^*$ finds a y^0 which our separation oracle accepts, our second observation implies that there is a solution of value $(\gamma+1)R^*$ for (P). This is a $(\gamma+1)$ -approximate solution for the program (P). \square

The solution obtained by solving the linear program (P) can be rounded into an integral solution using an LP-based algorithm that solves the (non-robust) optimization problem. Combining this with Lemma 2, we obtain the following.

Theorem 1. *Assume there is an α -approximation algorithm A_1 for the max-min fractional set cover problem with integer requirements, and an algorithm A_2 that given a subset S of clients, finds an integral solution that covers the clients in S and whose cost is at most β times the minimum cost of fractionally covering S . Then there is a $(\alpha + 1)\beta$ -approximation algorithm for the robust set cover problem.*

Proof. We run the algorithm described in the proof of Lemma 2 to compute an $(\alpha + 1)$ -approximate solution to the LP (P). The solution that this algorithm finds corresponds to $(\alpha + 1)y^0$, where y^0 is a first stage solution accepted by our separation oracle. As in the separation oracle, we define $C' = \{i \in C : \sum_{r:i \in r} y_r^0 < 1/(\alpha + 1)\}$. Now, we run the algorithm A_2 to find an integral set cover solution that covers clients in C' . The cost of this first stage solution is at most $(\alpha + 1)\beta$ times the cost of y^0 . Also, for every scenario in the second stage, we use A_2 to find a β -approximation to the optimal fractional second stage cost. This defines a $(\alpha + 1)\beta$ approximation algorithm for robust set cover. \square

By the above theorem, the main ingredient in solving a robust optimization problem with implicitly given scenarios is the algorithm for the max-min problem. In the next section, we show how an online algorithm for the underlying optimization problem can be used to approximately solve the max-min problem.

3 The max-min problems

The results of the previous section show that in order to solve the LP relaxation of the robust set cover problem, we need to consider the max-min problem. In this section, we design an $O(\log m)$ -approximation algorithm for max-min fractional set cover problem. In fact, we present a general framework to design approximation algorithms for a max-min problem using online competitive algorithms for the underlying optimization problem. Note that the max-min problems that we need to solve for approximating the robust covering problems are the fractional variants of the problems.

Given a universe F of clients and a subset $T \subseteq F$, let $\text{opt}(T)$ be the cost of an optimal (fractional) solution to cover all clients in T . Let \mathcal{A} be an α -competitive online algorithm for a covering problem. Namely, upon the arrival of any client a_k to an existing set of clients a_1, a_2, \dots, a_{k-1} , \mathcal{A} augments the current solution to a feasible solution for a_1, \dots, a_{k-1}, a_k . The algorithm is α -competitive if for every sequence of clients a_1, \dots, a_k the cost of the online solution produced by \mathcal{A} is at most α times the cost of the optimal (offline) solution for a_1, a_2, \dots, a_k . Let $\mathcal{A}(b|a_1, a_2, \dots, a_k)$ denote the *marginal increase* in the cost of the solution constructed by algorithm \mathcal{A} when we add a new element b to an existing sequence of clients (a_1, \dots, a_k) .

Consider two solutions w and w' for a fractional covering problem. Solution w' dominates solution w if for each set S the fractional value given to its respective

variable in w' is at least as large as that given in w . We say that the covering problem satisfies the *monotonicity* property, if for any two given solutions w and w' such that w' dominates w and any element a , the optimal marginal increase in expanding w' to cover a is not more than the optimal marginal increase in expanding w to cover a . It is not hard to prove that the set cover problem and its special cases satisfy this property.

The following theorem presents a relation between competitive online algorithms and approximation algorithms for the max-min problem.

Theorem 2. *Let \mathcal{A} be an α -competitive online algorithm for a covering problem. If the covering problem satisfies the monotonicity property then the corresponding max-min problem admits an $(\frac{e}{e-1})\alpha$ -approximation algorithm.*

Proof. Given the online algorithm \mathcal{A} for the covering problem, we prove that the following algorithm \mathcal{B} is a $(1 - \frac{1}{e})\alpha$ -approximation algorithm for the max-min problem:

1. $T = \emptyset$.
2. for $i = 1, \dots, k$ do
 - (a) Find a client a_i that maximizes $\mathcal{A}(a_i|a_1, a_2, \dots, a_{i-1})$ and add it to T .

Let the optimal solution to the max-min problem be the set $\{b_1, \dots, b_k\}$ of clients. Let OPT^* be the optimal cost of covering $\{b_1, \dots, b_k\}$. Let W_i be the cost of the solution of the online algorithm after i elements a_1, \dots, a_i have arrived and $L_i = \max[0, \text{OPT}^* - W_i]$. We prove that $L_i \leq (1 - \frac{1}{k})L_{i-1}$. Consider expanding the solution of the online algorithm for $\{a_1, \dots, a_{i-1}\}$ in the optimal way so that it covers $\{b_1, b_2, \dots, b_k\}$. The cost of this new solution is at least OPT^* . Hence there is some item b_j (with $1 \leq j \leq k$) such that there is difference of at least $\frac{\text{OPT}^* - W_{i-1}}{k}$ in cost between the case in which the clients b_1, \dots, b_{j-1} are added (or no clients at all, if $j = 1$) and the case in which the clients b_1, \dots, b_j are added. Since the covering problem satisfies the monotonicity property, adding b_j alone to $\{a_1, \dots, a_{i-1}\}$ requires an increase in cost of at least $\frac{\text{OPT}^* - W_{i-1}}{k}$ compared to the cost of \mathcal{A} covering $\{a_1, \dots, a_{i-1}\}$. Hence $W_{i-1} + (\frac{\text{OPT}^* - W_{i-1}}{k})$ is a lower bound on the cost of \mathcal{A} for covering $\{a_1, \dots, a_{i-1}, b_j\}$. Since algorithm \mathcal{B} chooses in the i th step the a_i that maximizes the marginal increase in the cost of \mathcal{A} , we will indeed have that $W_i \geq W_{i-1} + \frac{\text{OPT}^* - W_{i-1}}{k}$, and thus, $L_i \leq L_{i-1}(1 - \frac{1}{k})$. Thus $L_i \leq (1 - \frac{1}{k})^i L_0$. Therefore, $L_k \leq (1 - \frac{1}{k})^k \text{OPT}^* \leq \frac{1}{e} \text{OPT}^*$. This shows that $W_k \geq (1 - \frac{1}{e})\text{OPT}^*$. Since \mathcal{A} is an α -competitive algorithm, the true cost of covering $\{a_1, \dots, a_k\}$ is at least W_k/α , and algorithm \mathcal{B} is a $(\frac{e}{e-1})\alpha$ -approximation algorithm for the max-min problem. \square

Using Theorem 2 and known online algorithms, we can design approximation algorithms for the max-min problems. For example, an $O(\log m)$ -competitive algorithm for the online fractional set cover problem by Alon et al. [1] and Theorem 2 implies an $O(\log m)$ -approximation algorithm for the max-min fractional

set cover problem. In Section 5, we show that this result is nearly best possible (assuming certain complexity theoretic assumptions). This algorithm, together with the $O(\log n)$ randomized rounding algorithm for set cover and Theorem 1, imply the following.

Theorem 3. *There exists an $O(\log m \log n)$ -approximation algorithm for the robust two-stage set cover problem.*

Using the ideas of the 2-approximation algorithm for vertex cover by Bar-Yehuda and Even [2], we can design a 2-competitive online algorithm for vertex cover problem as follows. In the online algorithm, we keep track of a value $r(u)$ for each vertex u of the graph. We initialize these values to $r(u) = w(u)$. Upon the arrival of a new edge $e = uv$, the online algorithm sets $r_u = r_u - \min(r_u, r_v)$ and $r_v = r_v - \min(r_u, r_v)$. Observe that after this update either $r(u) = 0$ or $r(v) = 0$. At any moment, the fractional vertex cover solution is to pick $1 - \frac{r(u)}{w(u)}$ fraction of each vertex u . This means that we fully pick u or v for edge $e = uv$ and this solution is a feasible fractional vertex cover. Similar to the proof of Bar-Yehuda and Even [2], we can prove that this algorithm is a 2-competitive online algorithm. Using Theorem 2, this 2-competitive online algorithm implies a $(\frac{2e}{e-1})$ -approximation algorithm for the max-min fractional vertex cover problem. Applying the above results and the 2-approximate rounding procedure for the vertex cover problem, we get a $2(\frac{2e}{e-1} + 1)$ -approximation algorithm for the robust (weighted) vertex cover problem. The details are omitted.

For the edge cover problem, a simple 2-competitive online algorithm is to cover every arriving vertex with the cheapest edge incident to it. This, together with Theorems 2 and 1, give a constant-factor approximation algorithm for the robust edge cover problem. We do not optimize the constants of the approximation ratio for the weighted problems. However, in Section 4 we show a tight buy-at-once 2-approximation for unweighted vertex cover and edge cover.

4 Improved algorithms for unweighted problems

In this section, we give buy-at-once approximation algorithms for unweighted variants of robust set cover, vertex cover, and edge cover.

4.1 Robust unweighted set cover

In the robust unweighted set cover problem, all sets have unit cost. The input of the problem consists of n items, a collection of m sets, a parameter k (for number of items to be chosen by adversary), and an inflation factor $\lambda > 1$. To simplify notation, we assume here that parameters such as k, m and n are sufficiently large, and hence we shall ignore effects such as rounding $\ln m$ to the nearest integer. They affect the approximation ratio only by low order terms. The buy-at-once approximation algorithm for robust set cover is as follows:

1. Compute a minimum fractional set cover that covers all potential clients and let t be its size.
2. If $t < \frac{\lambda k}{\ln n}$, use the greedy algorithm to find a set cover. It will be of size at most $t \ln n$. Nothing needs to be done in the second stage.
3. If $t \geq \frac{\lambda k}{\ln n}$, do nothing in first stage. In the second stage, use a greedy algorithm to cover the items chosen by the adversary.

Theorem 4. *The above buy-at-once algorithm achieves an approximation ratio no worse than $\max(\ln n, \ln m)$ (up to low order terms) for unweighted robust set cover.*

Proof. Observe that by duality, t is the size of the maximum fractional packing. Let αt be the number of sets chosen by the optimal solution (to the robust set cover problem) in the first stage. Removing all items covered by these sets, the remaining set cover instance still has a fractional packing of value at least $(1 - \alpha)t$. (We may assume that $\alpha \leq 1$, as otherwise the analysis becomes even simpler.) Pick a set T of items, where each item is selected into T independently, with probability equal to its fractional value in the maximum fractional packing. The expected size of T is at least $(1 - \alpha)t$. In fact, known bounds by Siegel [19] imply that $|T| \geq \lfloor (1 - \alpha)t \rfloor$ with probability at least $1/2$. Moreover, every set is expected to contain at most one item from T , and Chernoff bound implies that with probability at least $1/2$, no set will contain more than $\ln 2m$ items. For simplicity of notation, we shall assume that T contains exactly $(1 - \alpha)t$ items, and no set contains more than $\ln m$ items from T . (This assumption affects only low order terms in the approximation ratio.) Hence in the second stage opt will pay at least $\min((1 - \alpha)t, k) \frac{\lambda}{\ln m}$, and in the two stages combined opt pays at least $\alpha t + \min((1 - \alpha)t, k) \frac{\lambda}{\ln m}$. This is a piecewise linear function in α , and its minimum is achieved when α is either 0 or 1, or when $(1 - \alpha)t = k$. It follows that opt pays at least $\min(t, \frac{k\lambda}{\ln m})$.

Now we can analyze the approximation ratio of our algorithm. When $t < \frac{\lambda k}{\ln n}$, the algorithm pays at most $t \ln n \leq \lambda k$, which is a factor of $\ln n$ larger than t , and at most a factor of $\ln m$ larger than $\frac{k\lambda}{\ln m}$. Hence the approximation ratio in this case is at most $\max(\ln m, \ln n)$.

When $t \geq \frac{\lambda k}{\ln n}$, the algorithm pays nothing in the first stage, and at most $\lambda k \leq t \ln n$ in the second stage. Again, the approximation ratio can be seen to be at most $\max(\ln m, \ln n)$. \square

The following example shows that the above analysis for the buy-at-once algorithm is tight up to a $\log \log m$ factor: consider an instance of the two-stage robust set cover where the ground set consists of $n + n^{1/4}$ elements and $k = n^{1/4}$ and $\lambda = n^{1/4}$. The family of subsets in the set cover instance is the family of all subsets of size $n^{1/4}$ of set $\{1, 2, \dots, n\}$ and all singleton sets $\{n + 1\}, \{n + 2\}, \dots, \{n + n^{1/4}\}$. The optimal solution is to buy all singleton sets in advance and wait for the scenario. Since the adversary should choose a set of size $n^{1/4}$ of $\{1..n\}$ and this set is in the family of sets in the set cover instance,

we can cover any scenario by buying one set at cost $\lambda = n^{1/4}$ later. Thus, the cost of the optimal solution is at most $2n^{1/4}$. If we do not buy any set in advance, the adversary selects $\{n+1, n+2, \dots, n+n^{1/4}\}$ and we should pay $\lambda k = n^{1/2}$ in the second stage. On the other hand, in order to cover all elements in the first stage, we need to buy at least $n^{3/4}$ sets. Both of these cases are more than a factor of $n^{1/4} = \Omega(\frac{\log m}{\log \log m})$ larger than the optimal solution.

Also, observe that the term $\ln n$ in the approximation ratio cannot be improved by any polynomial-time algorithm (e.g., when $\lambda = \infty$), due to the hardness of approximating minimum set cover [10]. It is not clear whether the term $\ln m$ can be improved.

4.2 Robust unweighted vertex cover

Consider the following buy-at-once algorithm for the robust unweighted vertex cover problem: compute a maximum matching M in G , and let $|M|$ denote its size. If $|M| < \lambda k$, then we pick a vertex cover of size no larger than $2|M|$ in the first stage (for example, by picking both endpoints of every edge in M) and nothing needs to be done in the second stage. If $|M| \geq \lambda k$, we do nothing in the first stage and in the second stage, for each edge that is present in the realized scenario, we pick one of its endpoints arbitrarily.

Theorem 5. *The above algorithm achieves an approximation ratio of 2 for the unweighted robust vertex cover problem.*

Proof. Let OPT denote an optimal algorithm, and let x denote the number of vertices purchased by this algorithm in the first stage. Thus, at least $|M| - x$ of the edges in M are not covered by OPT in the first stage. Consider the scenario where the adversary picks $\min(k, |M| - x)$ of these edges in the second stage. The overall cost of OPT in this scenario is $T := x + \lambda \min(k, |M| - x)$. Since $\lambda \geq 1$, we have $T \geq \min(\lambda k, |M|)$.

Now, we show that the cost of our algorithm is always at most $2T$. We consider two cases: if $|M| < \lambda k$, our algorithm buys a vertex cover of cost at most $2|M|$ in the first stage. Therefore, the cost of our algorithm is at most $2|M| = 2 \min(\lambda k, |M|) \leq 2T$. If $|M| \geq \lambda k$, our algorithm incurs a cost of $\lambda k = \min(\lambda k, |M|) \leq T$. Therefore, in this case our algorithm is actually optimal. \square

Note that any algorithm that approximates unweighted robust vertex cover within a ratio better than 2 can be used to approximate the minimum vertex cover problem within a ratio better than 2 (e.g., by setting $\lambda = \infty$), and achieving this would resolve a long standing open problem.

4.3 Robust unweighted edge cover

The input of the unweighted edge cover problem is a graph with n vertices, m edges, a parameter k (for number of vertices to be chosen by adversary), and

an inflation factor $\lambda > 1$. All edges have unit cost. Observe that the number of edges needed to cover ℓ vertices is always between $\ell/2$ and ℓ . This fact serves as a basis for a tight 2-approximation for the robust unweighted edge cover problem. The algorithm and the proof are left to the full version of the paper. Moreover, we can prove that if $P \neq NP$, then the max-min variant and the robust two-stage variant of the edge cover problem cannot be approximated within a factor better than 2.

5 Hardness of max-min problems

In this section, we give a strong inapproximability result for the max-min (fractional) set cover problem. First, we show a hardness result for the max-min (fractional) edge cover problem.

Theorem 6. *For every $\epsilon > 0$, it is NP-hard to approximate the max-min unweighted edge cover problem within a ratio better than $2 - \epsilon$.*

Proof. The proof is by reduction from the maximum independent set problem. As shown in [14], for every sufficiently small $\epsilon > 0$, it is NP-hard to distinguish between the following two classes of graphs:

Yes instances: graphs on n vertices that contain an independent set of size ϵn .

No instances: graphs on n vertices with no independent set of size $\epsilon^5 n$.

In order to distinguish whether a given graph G is a yes instance or a no instance, we construct an instance of the max-min edge cover problem that consists of G and $k = \epsilon n$. On yes instances, one can select k vertices that form an independent set in G , and then k edges are needed in order to cover them. On no instances, whenever there are more than $\epsilon^5 k$ vertices, two of them share an edge in G . It follows that any selection of k vertices can be covered by $\epsilon^5 k + (1 - \epsilon^5)k/2 < k/(2 - \epsilon)$ edges. Therefore, any algorithm that approximates the max-min unweighted edge cover problem within a factor better than $2 - \epsilon$ can be used to distinguish between these classes. \square

The proof of Theorem 6 can be adopted easily for the max-min fractional edge cover problem. This implies that for any $\epsilon > 0$, it is NP-hard to approximate the max-min fractional edge cover problem within a factor better than $2 - \epsilon$. This hardness ratio can be strengthened to nearly logarithmic factors for the fractional set cover problem, but proving this using current techniques seems to require assumptions stronger than $P \neq NP$. Picking $p(n) = \sqrt{n}$ in Theorem 7 shows that the max-min (fractional) set cover problem cannot be approximated within a ratio better than $\Omega(\frac{\log N}{\log \log N})$ (on instances of size N) unless 3SAT can be solved in time $2^{O(\sqrt{n})}$ (on instances of size n).

Theorem 7. *For every $0 < \delta < 1$ and $p(n) = n^\delta$, the max-min fractional set cover problem cannot be approximated within a ratio better than $\Omega(\frac{p(n)}{\log p(n)})$ on*

instances of size $N = 2^{O(p(n))}$ (in time polynomial in N), unless NP problems (say 3SAT) can be solved in time $2^{O(p(n))}$.

Proof. The proof is presented for the integral set cover problem, but the approximation hardness applies also to the max-min fractional set cover problem, because in the *yes* instance the cover is disjoint. The proof is based on the structure of instances of set cover that are generated by the reduction described in [10], and specifically, on the parameters given in Section 6 in [10]. Here we only sketch the proof.

Recall that in [10], the hardness of approximation result is based on a certain multiple-prover proof system. We shall need the number of provers (denoted in [10] by k) to be $p(n)$. (Hence one cannot use here the earlier [16] instead of [10].) In [10] it suffices that the number of parallel repetitions ℓ is logarithmic in the number of provers, hence we can have $\ell = O(\log(p(n)))$. (Remark: later work [9] used a version of a multilayered PCP which is somewhat simpler than the multiple prover system of [10]. This simpler version requires ℓ to grow at a faster rate than $p(n)$, and would result in weaker conclusions if used in the proof of Theorem 7.) This results in a set cover instance with $2^{O(p(n))}$ clients and sets.

Each subset in [10] would be an item in the max-min set cover problem. Each item in [10] would be a set in the max-min set cover problem. Note that in [10] all sets are of the same size, and there is a disjoint set cover for *yes* instances, say, by t sets. We shall set k for the max-min set cover problem to be equal to this t . Hence *yes* instances of [10] correspond to *yes* instances of max-min set cover for which k clients can be selected that require k sets in order to be covered.

The property of *no* instances of [10] that we shall use is the following: for every $q < p(n)$, for every collection of $tq/p(n)$ sets, there is some item that belongs to $O(p(n)/q)$ of the sets. Extensions of the analysis in [10] can be used in order to prove this property, but this is omitted from the current paper.

The property above implies that for *no* instances in [10], for every collection of t sets there are $O(t \frac{\log(p(n))}{p(n)})$ clients that hit all the sets. This implies that in *no* instances of the max-min set cover problem, the optimum solution has value $O(t \frac{\log(p(n))}{p(n)})$. \square

Acknowledgements

This work benefited greatly from insightful comments that anonymous reviewers provided on previous versions of this manuscript.

References

1. N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. A general approach to online network optimization problems. In *SODA*, pages 577–586, 2004.
2. R. Bar-Yehuda and S. Even. A linear time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2:198–203, 1981.

3. A. Ben-Tal, A. Goryashko, E. Guslitzer, A. Nemirovski. Adjustable robust solutions of uncertain linear programs. 351-376 *Mathematical Programming*, 99:2:351-376, 2004.
4. D. Bertsimas and M. Sim. The price of robustness. *Operation Research*, 52:35–53.
5. D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming Series B*, 98:49–71.
6. J. Birge and F. Louveaux. Introduction to stochastic programming. *Springer*, Berlin, 1997.
7. G. B. Dantzig. Linear programming under uncertainty. *Management Science*, 1:197–206, 1955.
8. K. Dhamdhere, V. Goyal, R. Ravi, and M. Singh. How to pay, come what may: Approximation algorithms for demand-robust covering problems. *FOCS*, 2005.
9. I. Dinur, V. Guruswami, S. Khot, and O. Regev. New multilayered pcp and the hardness of hypergraph vertex cover. *SIAM Journal of Computing*, 34(5):1129–1146, 2005.
10. U. Feige. A threshold of $\ln n$ for approximating set cover. *JACM*, 45(4):634–652, 1998.
11. D. Golovin, V. Goyal, and R. Ravi. Pay today for a rainy day: Improved approximation algorithms for demand-robust min-cut and shortest path problems. *STACS*, 2006.
12. A. Gupta, M. Pal, R. Ravi, and A. Sinha. Boosted sampling: Approximation algorithms for stochastic optimization. In *STOC*, pages 170–178, 2004.
13. A. Gupta, R. Ravi, and A. Sinha. An edge in time saves nine: Lp rounding approximation algorithms for stochastic network design. *FOCS*, 45, 2004.
14. J. Hastad. Clique is hard to approximate. In *FOCS*, pages 627–636, 1996.
15. N. Immerlica, D. Karger, M. Minkoff, and V. S. Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *SODA*, 2004.
16. C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *JACM*, 41(5):960–981, 1994.
17. Y. Nikulin. Robustness in combinatorial optimization and scheduling theory: An annotated bibliography. *Technical Report SOR-91-13, Statistics and Operation Research*, http://www.optimization-online.org/DB_FILE/2004/11/995.pdf, 2004.
18. R. Ravi and A. Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. *IPCO*, pages 101–115, 2004.
19. A. Siegel, Median Bounds and Their Application. *J. Algorithms* 38(1): 184-236 (2001).
20. D. Shmoys and S. Swamy. Stochastic optimization is (almost) as easy as deterministic optimization. In *FOCS*, 2004.