



# Randomized Strategies for Robust Combinatorial Optimization with Approximate Separation

Yasushi Kawase<sup>1</sup> · Hanna Sumita<sup>2</sup>

Received: 10 March 2023 / Accepted: 20 September 2023 / Published online: 10 October 2023  
© The Author(s) 2023

## Abstract

In this paper, we study the following robust optimization problem. Given a set family representing feasibility and candidate objective functions, we choose a feasible set, and then an adversary chooses one objective function, knowing our choice. The goal is to find a randomized strategy (i.e., a probability distribution over the feasible sets) that maximizes the expected objective value in the worst case. This problem is fundamental in wide areas such as artificial intelligence, machine learning, game theory, and optimization. To solve the problem, we provide a general framework based on the dual linear programming problem. In the framework, we utilize the ellipsoid algorithm with the approximate separation algorithm. We prove that there exists an  $\alpha$ -approximation algorithm for our robust optimization problem if there exists an  $\alpha$ -approximation algorithm for finding a (deterministic) feasible set that maximizes a nonnegative linear combination of the candidate objective functions. Using our result, we provide approximation algorithms for the max–min fair randomized allocation problem and the maximum cardinality robustness problem with a knapsack constraint.

**Keywords** Robust optimization · Ellipsoid method · Max–min fair allocation · Maximum cardinality robustness problem

---

Portions of this paper are based on [18, 19].

---

✉ Yasushi Kawase  
kawase@mist.i.u-tokyo.ac.jp

✉ Hanna Sumita  
sumita@c.titech.ac.jp

<sup>1</sup> The University of Tokyo, Tokyo, Japan

<sup>2</sup> Tokyo Institute of Technology, Tokyo, Japan

## 1 Introduction

This paper addresses robust combinatorial optimization. Let  $E$  be a finite ground set, and let  $n$  be a positive integer. Suppose that we are given  $n$  set functions  $f_1, \dots, f_n: 2^E \rightarrow \mathbb{R}_+$  and a feasibility constraint  $\mathcal{I} \subseteq 2^E$ . Here,  $\mathbb{R}_+$  is the set of nonnegative reals. The functions  $f_1, \dots, f_n$  represent possible scenarios. We assume that each function is given by a *value oracle*, i.e., for a given  $X \subseteq E$ , we can query the oracles about the values  $f_1(X), \dots, f_n(X)$ . For each  $k = 1, \dots, n$ , we denote by  $X_k^*$  an optimal solution for the  $k$ th function  $f_k$  (i.e.,  $X_k^* \in \arg \max_{X \in \mathcal{I}} f_k(X)$ ), and assume that  $f_k(X_k^*) > 0$ .

Throughout this paper, we denote  $[n] = \{1, \dots, n\}$ . The worst case value for  $X \in \mathcal{I}$  across all scenarios is defined as  $\min_{k \in [n]} f_k(X)$ . We focus on a randomized strategy for the robust optimization problem, i.e., a probability distribution over  $\mathcal{I}$ . Let  $\Delta(\mathcal{I}) \subseteq [0, 1]^{\mathcal{I}}$  and  $\Delta_n \subseteq [0, 1]^n$  denote the set of probability distributions over  $\mathcal{I}$  and  $[n]$ , respectively. The worst case value for a randomized strategy  $p \in \Delta(\mathcal{I})$  is defined as  $\min_{k \in [n]} \sum_{X \in \mathcal{I}} p_X \cdot f_k(X)$ . The aim of this paper is to solve the following robust optimization problem:

$$\max_{p \in \Delta(\mathcal{I})} \min_{k \in [n]} \sum_{X \in \mathcal{I}} p_X \cdot f_k(X). \tag{1}$$

This problem is regarded as the problem of computing the *game value* in a two-person zero-sum game where one player (Algorithm) selects a feasible solution and the other player (Adversary) selects an objective function.

There are two advantages to adopting a randomized strategy rather than a deterministic one for the problem (1). One is that the randomization improves the worst case value dramatically. Suppose that  $\mathcal{I} = \{\emptyset, \{a\}, \{b\}\}$ ,  $f_1(X) = |X \cap \{a\}|$ , and  $f_2(X) = |X \cap \{b\}|$ . Then, the maximum worst case value among deterministic strategies is  $\max_{X \in \mathcal{I}} \min_{k \in \{1,2\}} f_k(X) = 0$ , while the maximum worst case value among randomized ones is  $\max_{p \in \Delta(\mathcal{I})} \min_{k \in \{1,2\}} \sum_{X \in \mathcal{I}} p_X \cdot f_k(X) = 1/2$ . The other merit is that the optimal randomized strategy can be found easier. It is known that finding an optimal deterministic solution is hard even in a simple setting [1, 17]. In particular, even for an easy case, computing the optimal worst case value among deterministic solutions is NP-hard even to approximate, while we can compute that among randomized solutions in polynomial time. We will show these facts in this paper.

The robust optimization problem also has widespread applications in game theory and combinatorial optimization. One is the problem of computing a *Stackelberg equilibrium* of the (zero-sum) *security games*. This game models interaction between a system *defender* (Algorithm) and a malicious *attacker* (Adversary) to the system. A Stackelberg equilibrium is an optimal solution that maximizes the defender’s utility, taking into account the constraint that the attacker plays a best response to the defender’s action. The model and its game-theoretic solution have various applications in the real world [33]. Another application is the problem of maximizing the *cardinality robustness* for the maximum weight independent set problem [9, 12, 14, 23, 28]. The goal is to choose an independent set of size at most  $k$  with as large total weight as possible, but the cardinality bound  $k$  is not known in advance. We refer

this problem to the *maximum cardinality robustness problem (MCRP)*. We can regard MCRP as the game where Algorithm chooses an independent set  $X$ , and then Adversary chooses  $k$  knowing  $X$ . In addition, dense subgraph discovery in a multilayer network is studied as a robust optimization problem. Randomized solutions for this problem are studied in the paper [20].

One of the most standard ways to solve the robust optimization problem is to use the *linear programming (LP)*. In fact, it is known that we can compute the exact game value in polynomial time with respect to the numbers of deterministic (pure) strategies for both players (see, e.g., [2, 29] for more details). However, in our setting, direct use of the LP formulation is not effective. This is because the set of deterministic strategies for Algorithm has exponentially large cardinality in general, and hence the number of the variables in the LP formulation is exponentially large. We overcome this difficulty by taking the dual and utilizing an (approximate) separation algorithm.

Another approach is to use the multiplicative weights update (MWU) method. The MWU method is an algorithmic technique that maintains a distribution on a certain set of interests and updates it iteratively by multiplying the probability mass of elements by suitably chosen factors based on feedback obtained by running another algorithm on the distribution [16]. MWU is simple but so powerful that it is widely used in game theory, machine learning, computational geometry, optimization, and so on. Freund and Schapire [7] apply the MWU method to calculate the approximate value of a two-person zero-sum game, and showed that if (i) the set of deterministic strategies for Adversary is polynomially sized, and (ii) Algorithm can compute a *best response*, then MWU yields a polynomial-time algorithm to compute the game value up to an additive error of  $\epsilon$  for any fixed constant  $\epsilon > 0$ . Krause [26] and Chen et al. [5] extended this result for the case when Algorithm can compute only an approximately best response. They provided a polynomial-time algorithm that finds an  $\alpha$ -approximation of the game value up to an additive error of  $\epsilon \cdot \max_{k \in [n], X \in \mathcal{I}} f_k(X)$  for any fixed constant  $\epsilon > 0$ . This result leads to no theoretical guarantee in general because the maximum objective value can be arbitrarily large compared with the optimal value. Moreover, obtaining an  $(\alpha - \epsilon')$ -approximate solution of the problem (1) for a fixed constant  $\epsilon' > 0$  by their algorithms requires pseudo-polynomial time. Here, an algorithm is said to be an  $\alpha$ -approximation algorithm if, for any problem instance, it yields a solution whose objective value is at least  $\alpha$  times the optimal value.

## Related Work

While there exist still few papers on randomized strategies of the robust optimization problems, algorithms to find a deterministic strategy have been intensively studied in various settings. See survey papers [1, 17] and references therein for details. Krause et al. [25] focused on solving  $\max_{X \subseteq E, |X| \leq \ell} \min_{k \in [n]} f_k(X)$  where  $f_k$ 's are monotone submodular functions. Those authors showed that this problem is NP-hard even to approximate and provided an algorithm that outputs a set  $X$  of size  $\ell \cdot (1 + \log(\max_{e \in E} \sum_{k \in [n]} f_k(\{e\})))$  whose objective value is at least as good as the optimal value. Orlin et al. [30] provided constant-factor approximate algorithms to

solve  $\max_{X \subseteq E, |X| \leq k} \min_{Z \subseteq X, |Z| \leq \tau} f(X - Z)$ , where  $f$  is a monotone submodular function.

Since Hassin and Rubinfeld [12] introduced the notion of the cardinality robustness, many papers have been investigating the value of the maximum cardinality robustness [9, 12, 14, 15, 23]. Kakimura et al. [15] proved that the deterministic version of MCRP is weakly NP-hard but admits an FPTAS. Matuschke et al. [28] introduced randomized strategies for cardinality robustness, and they presented a randomized strategy with  $(1/\ln 4)$ -robustness for a certain class of independence system. However, they did not consider the computational aspect of cardinality robustness. Kobayashi and Takazawa [23] focused on independence systems that are defined from the knapsack problem, and exhibited two randomized strategies with robustness  $\Omega(1/\log \sigma)$  and  $\Omega(1/\log \nu)$ , where  $\sigma$  is the exchangeability of the independence system and  $\nu = \frac{\text{the size of a maximum independent set}}{\text{the size of a minimum dependent set} - 1}$ .

When  $n = 1$ , the deterministic version of the robust optimization problem is exactly the classical optimization problem  $\max_{X \in \mathcal{I}} f(X)$ . It is well-known that the problem is solvable in polynomial-time if  $f$  is additive and  $\mathcal{I}$  is a matroid intersection constraint [31]. If  $f$  is additive and  $\mathcal{I}$  is a  $\mu$ -matroid intersection constraint, there is a  $1/(\mu - 1 + \epsilon)$ -approximation algorithm for any fixed  $\epsilon > 0$  and  $\mu \geq 3$  [27]. For the monotone submodular function maximization problem, there exist  $(1 - 1/e)$ -approximation algorithms under a knapsack constraint [32] or a matroid constraint [3, 6], and there exists a  $1/(\mu + \epsilon)$ -approximation algorithm under a  $\mu$ -matroid intersection constraint for any fixed  $\epsilon > 0$  and  $\mu \geq 2$  [27]. As for the case when the objective function  $f$  is additive, the knapsack problem admits an FPTAS [21].

### Our Results

We provide a general framework based on a dual LP to solve the robust optimization problem. Our framework achieves better performance than the MWU based frameworks [5, 26]. In our framework, we utilize the ellipsoid algorithm with the approximate separation algorithm. The separation problem in the framework is of the form  $\max_{X \in \mathcal{I}} \sum_{k \in [n]} z_k f_k(X)$  where  $z_k$  is a nonnegative real for each  $k \in [n]$ . The framework provides an  $\alpha$ -approximation algorithm for the robust optimization problem (1) if we are given an  $\alpha$ -approximation algorithm to solve  $\max_{X \in \mathcal{I}} \sum_{k \in [n]} z_k f_k(X)$  for any  $z \in \mathbb{R}_+^n$ . Specifically, if  $\max_{X \in \mathcal{I}} \sum_{k \in [n]} z_k f_k(X)$  is polynomial-time solvable for any  $z \in \mathbb{R}_+^n$ , the robust optimization problem is also solvable in polynomial-time.

The above framework yields approximation algorithms according to classes of objective functions and constraints. For example, let us focus on when objective functions  $f_1, \dots, f_n$  are additive. The problem (1) is polynomial-time solvable if the constraint  $\mathcal{I}$  is given by a matroid (intersection), since the separation problem coincides with the maximum weight matroid intersection problem and this is polynomial-time solvable. Similarly, when  $\mathcal{I}$  is a knapsack constraint, we can provide an FPTAS for the robust optimization problem since the knapsack problem admits an FPTAS. For monotone submodular functions  $f_1, \dots, f_n$  and a knapsack or a matroid constraint  $\mathcal{I}$ , we obtain a  $(1 - 1/e)$ -approximation algorithm by using the  $(1 - 1/e)$ -approximation

**Table 1** The approximation ratios for robust optimization problems shown in the present paper

Objective functions	Constraint	Approx. ratio	Reference
additive	matroid intersection	P	Theorem 4
additive	$\mu$ -matroid intersection	$1/(\mu - 1 + \epsilon)$	Theorem 6
additive	knapsack	FPTAS	Theorem 7
additive (shortest path)	$s$ - $t$ path	P	Theorem 8
monotone submodular	matroid/knapsack	$(1 - \frac{1}{e})$ -approx	Theorem 9
monotone submodular	$\mu$ -matroid intersection	$1/(\mu + \epsilon)$ -approx	Theorem 11
MCRP	knapsack	FPTAS	Theorem 14

algorithms for the monotone submodular function maximization problem. For MCRP with a knapsack constraint, we design an FPTAS by constructing an FPTAS for the corresponding separation problem. Our results are summarized in Table 1.

## 2 Preliminaries

For given objective functions  $f_1, \dots, f_n: 2^E \rightarrow \mathbb{R}_+$  and a feasibility constraint  $\mathcal{I} \subseteq 2^E$ , we consider problem (1).

### Objective functions

Throughout this paper, we consider set functions  $f$  with  $f(\emptyset) = 0$ . We say that a set function  $f: 2^E \rightarrow \mathbb{R}$  is *submodular* if  $f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$  holds for all  $X, Y \subseteq E$  [8, 24]. In particular, a set function  $f: 2^E \rightarrow \mathbb{R}$  is called *additive* (or *modular*) if  $f(X) + f(Y) = f(X \cup Y) + f(X \cap Y)$  holds for all  $X, Y \subseteq E$ . An additive function  $f$  is represented as  $f(X) = \sum_{e \in X} w_e$  for some  $w \in \mathbb{R}^E$ . A function  $f$  is said to be *monotone* if  $f(X) \leq f(Y)$  for all  $X \subseteq Y \subseteq E$ . An additive function  $f(X) = \sum_{e \in X} w_e$  is monotone if and only if  $w_e \geq 0$  ( $\forall e \in E$ ).

### Feasibility constraints

Let  $E$  be a finite ground set. An *independence system* is a set system  $(E, \mathcal{I})$  with the following properties: (I1)  $\emptyset \in \mathcal{I}$ , and (I2)  $X \subseteq Y \in \mathcal{I}$  implies  $X \in \mathcal{I}$ . A set  $X \subseteq \mathcal{I}$  is said to be *independent*, and an inclusion-wise maximal independent set is called a *base*.

A *matroid* is an independence system  $(E, \mathcal{I})$  satisfying that (I3) if  $X, Y \in \mathcal{I}$  and  $|X| < |Y|$  then there exists  $e \in Y \setminus X$  such that  $X \cup \{e\} \in \mathcal{I}$ . All bases of a matroid have the same cardinality, which is called the *rank* of the matroid and is denoted by  $\rho(\mathcal{I})$ . An example of matroids is a uniform matroid  $(E, \mathcal{I})$ , where  $\mathcal{I} = \{X \subseteq E : |X| \leq r\}$  for some  $r$ . Note that the rank of this uniform matroid is  $r$ . Given two matroids  $\mathcal{M}_1 = (E, \mathcal{I}_1)$  and  $\mathcal{M}_2 = (E, \mathcal{I}_2)$ , the *matroid intersection* of  $\mathcal{M}_1$  and  $\mathcal{M}_2$  is defined by  $(E, \mathcal{I}_1 \cap \mathcal{I}_2)$ . By definition, a matroid is a matroid intersection in which one matroid is a uniform matroid with rank  $r = |E|$ . Similarly, given  $\mu$  matroids  $(E, \mathcal{I}_1), \dots, (E, \mathcal{I}_\mu)$ , the  $\mu$ -matroid intersection is defined by  $(E, \bigcap_{i=1}^{\mu} \mathcal{I}_i)$ .

We assume that the independence oracle for each matroid is given when we consider a matroid intersection or a  $\mu$ -matroid intersection constraint.

Given an item set  $E$  with size  $s(e)$  and value  $v(e)$  for each  $e \in E$ , and a capacity  $c \in \mathbb{R}_+$ , the *knapsack problem* is to find a subset  $X$  of  $E$  that maximizes the total value  $\sum_{e \in X} v(e)$  subject to a knapsack constraint  $\sum_{e \in X} s(e) \leq c$ . Each subset satisfying the knapsack constraint is called a knapsack solution. Let  $\mathcal{I} = \{X \subseteq E : \sum_{e \in X} s(e) \leq c\}$  be the family of knapsack solutions. Then,  $(E, \mathcal{I})$  is an independence system.

**Hardness of the problem**

We show that computing the optimal worst case value among deterministic solutions is strongly NP-hard even to approximate, as we mentioned in Introduction. To prove this, we reduce from the *hitting set problem*, which is known to be NP-hard [10]. Given  $n$  subsets  $S_k \subseteq E$  ( $k \in [n]$ ) on a ground set  $E$  and an integer  $r$ , the hitting set problem is to find a subset  $A \subseteq E$  such that  $|A| \leq r$  and  $S_k \cap A \neq \emptyset$  for all  $k \in [n]$ , if such a subset exists.

**Theorem 1** *It is NP-hard to compute the value*

$$\max_{X \in \mathcal{I}} \min_{k \in [n]} f_k(X) \tag{2}$$

*even when the objective functions  $f_1, \dots, f_k$  are additive and  $\mathcal{I}$  is given by a uniform matroid. Moreover, there exists no approximation algorithm for the problem unless  $P=NP$ .*

**Proof** Let  $(E, \{S_1, \dots, S_n\}, r)$  be an instance of the hitting set problem. We construct an instance of the problem (2) as follows. The constraint  $\mathcal{I}$  is defined as the uniform matroid over  $E$  with rank  $r$ , i.e.,  $\mathcal{I} = \{X \subseteq E : |X| \leq r\}$ . Each objective function  $f_k$  ( $k \in [n]$ ) is defined by  $f_k(X) = |X \cap S_k|$  ( $X \subseteq E$ ), which is additive.

If there exists a hitting set  $X \in \mathcal{I}$ , then  $\min_{k \in [n]} f_k(X) \geq 1$ , which implies that the optimal value of (2) is at least 1. On the other hand, if any  $X \in \mathcal{I}$  is not a hitting set, then  $\min_{k \in [n]} f_k(X) = 0$  for all  $X \in \mathcal{I}$ , meaning that the optimal value of (2) is 0. Therefore, even deciding whether the optimal value of (2) is positive or zero is NP-hard. Thus, there exists no approximation algorithm to the problem unless  $P = NP$ . □

Note that the randomized version of the problem (2) is solvable in polynomial time by Theorem 4.

We conclude this section by presenting details about the applications of (1).

**Security game**

In a security game, we are given a set  $E$  of  $n$  targets. One player called defender selects a set of targets  $X \in \mathcal{I} \subseteq 2^E$ , and then the other player called attacker selects one target  $i \in E$ . A typical example of the constraint is  $\mathcal{I} = \{X \subseteq E : |X| \leq \ell\}$ , which means that the defender can protect at most  $\ell$  targets at the same time. The utility of defender is  $r_i$  if  $i \in X$  and  $c_i$  if  $i \notin X$ . Then, we can interpret the game as the robust optimization with  $f_i(X) = c_i + \sum_{j \in X} w_{ij}$  ( $i \in E$ ), where  $w_{ii} = r_i - c_i$  and  $w_{ij} = 0$  for  $j \neq i$ . Note that all of the functions are additive. Then the problem of computing the Stackelberg equilibrium is equivalent to (1).

### Max–min fair allocation

Suppose that we are given indivisible goods  $E$  and agents  $N = [n]$ , where each agent  $k \in N$  has a utility function  $u_k: 2^E \rightarrow \mathbb{R}_+$ . Our goal is to allocate goods to agents so as to make the least happy agent as happy as possible. Such an allocation is said to achieve the max–min fairness. Let  $\mathcal{I} = \{X \subseteq [n] \times E : |X \cap \{(k, e) : e \in E\}| \leq 1 \ (\forall k \in [n])\}$  and  $f_k(X) = u_k(\{e \in E : (k, e) \in X\})$ . Then, each  $X \in \mathcal{I}$  corresponds to an allocation such that agent  $k \in N$  receives a bundle  $\{e \in E : (k, e) \in X\}$ . Then, the max–min fair randomized allocation problem is to solve  $\max_{p \in \Delta(\mathcal{I})} \min_{k \in [n]} \sum_{X \in \mathcal{I}} p_X \cdot f_k(X)$ .

### MCRP

Consider that given an independence system  $(E, \mathcal{I})$  with weights of elements in  $E$ , we are required to choose  $X \in \mathcal{I}$  of size at most  $k$  with as large total weight as possible, but  $k$  is not known in advance. For each  $X \in \mathcal{I}$ , we denote the total weight of the  $k$  heaviest elements in  $X$  by  $v_{\leq k}(X)$ . For  $\alpha \in [0, 1]$ , an independent set  $X \in \mathcal{I}$  is said to be  $\alpha$ -robust if  $v_{\leq k}(X) \geq \alpha \cdot \max_{Y \in \mathcal{I}} v_{\leq k}(Y)$  for any  $k \in [n]$ . Then, MCRP is a problem to find a randomized strategy that maximizes the robustness  $\alpha$ , i.e., solve

$$\max_{p \in \Delta(\mathcal{I})} \min_{k \in [n]} \frac{\sum_{X \in \mathcal{I}} p_X \cdot v_{\leq k}(X)}{\max_{Y \in \mathcal{I}} v_{\leq k}(Y)}.$$

This is formulated as problem (1) by setting  $f_k(X) = \frac{v_{\leq k}(X)}{\max_{Y \in \mathcal{I}} v_{\leq k}(Y)}$ .

## 3 Dual-Based Framework

In this section, we propose a computation framework for the robust optimization problem (1). The problem can be described as the following LP:

$$\begin{aligned} \max \quad & t \\ \text{s.t.} \quad & t \leq \sum_{X \in \mathcal{I}} f_i(X) p_X \quad (\forall i \in [n]), \\ & \sum_{X \in \mathcal{I}} p_X = 1, \\ & p_X \geq 0 \quad (\forall X \in \mathcal{I}). \end{aligned} \quad (3)$$

We may assume that the optimal value  $t$  is positive because a solution  $(p, t)$  defined by  $p_X = |\{i \in [n] : X = X_i^*\}|/n$  ( $X \in \mathcal{I}$ ) and  $t = \sum_{i \in [n]} f_i(X_i^*)/n > 0$  is feasible. Here, recall that  $X_k^* \in \arg \max_{X \in \mathcal{I}} f_k(X)$  and  $f_k(X_k^*) > 0$  is assumed for every  $k \in [n]$ . By setting  $q_X = p_X/t$  for each  $X \in \mathcal{I}$  and changing the objective to minimize  $1/t$ , we obtain the following equivalent and simpler LP:

$$\begin{aligned} \min \quad & \sum_{X \in \mathcal{I}} q_X \\ \text{s.t.} \quad & \sum_{X \in \mathcal{I}} f_i(X) q_X \geq 1 \quad (\forall i \in [n]), \\ & q_X \geq 0 \quad (\forall X \in \mathcal{I}). \end{aligned} \quad (4)$$

Then, the dual LP is given as follows:

$$\begin{aligned} \max \quad & \sum_{i \in [n]} z_i \\ \text{s.t.} \quad & \sum_{i \in [n]} f_i(X) z_i \leq 1 \quad (\forall X \in \mathcal{I}), \\ & z_i \geq 0 \quad (\forall i \in [n]). \end{aligned} \tag{5}$$

In the following, we aim to solve problem (4) via solving (5) by the ellipsoid algorithm. This algorithm works when we have a separation algorithm to solve the separation problem [11]. For a polyhedron  $P \subseteq \mathbb{R}^n$ , the *separation problem* for  $P$  receives a vector  $y$  and either asserts  $y \in P$  or finds a vector  $d$  such that  $d^\top x > d^\top y$  for all  $x \in P$ . In our setting, we need a separation algorithm for the feasible region of (5). Suppose that  $\hat{z} \in \mathbb{R}^n$  is an input of the separation problem. If  $\hat{z}_i < 0$  for some  $i \in [n]$ , then  $\hat{z}$  is infeasible and  $-\hat{z}_i > 0 \geq -z_i$  for all feasible solutions  $z \in \mathbb{R}^n$ . Hence, assume that  $\hat{z}_i \geq 0$  for all  $i \in [n]$ . We observe that checking the feasibility of a given input  $\hat{z} \in \mathbb{R}^n$  can be done by solving  $\max \sum_{i \in [n]} f_i(X) \hat{z}_i$ . For  $z \in \mathbb{R}^n$ , we define a function  $f^z: 2^E \rightarrow \mathbb{R}_+$  by

$$f^z(X) = \sum_{i \in [n]} f_i(X) z_i, \tag{6}$$

and consider the following optimization problem:

$$\max_{X \in \mathcal{I}} f^z(X). \tag{7}$$

If the optimal solution  $X^* \in \mathcal{I}$  has the objective value larger than 1 for the problem (7) with  $z = \hat{z}$ , then  $(f_1(X^*), \dots, f_n(X^*))$  is a desired vector as  $d$ . This is because  $\sum_{i \in [n]} f_i(X^*) \hat{z}_i > 1 \geq \sum_{i \in [n]} f_i(X^*) z_i$  for all feasible solutions  $z \in \mathbb{R}^n$ . Note that if  $f_i$ 's are additive or submodular, then the objective function  $f^z$  is also additive or submodular for any  $z \in \mathbb{R}_+^n$ , respectively.

The solvability of the separation problem is dependent on the class of the objective functions  $f_i$ 's and the constraint  $\mathcal{I}$ . It is generally difficult to solve the separation problem in polynomial time. As a remedy, we employ the technique of Jansen [13] that uses an approximate separation algorithm to solve the LP approximately. For a polyhedron  $P \subseteq \mathbb{R}^n$  and a positive  $\alpha \leq 1$ , the  $\alpha$ -approximate separation algorithm for  $P$  receives a vector  $y$  and either asserts  $\alpha y \in P$  or finds a vector  $d$  such that  $d^\top x > d^\top y$  for all  $x \in P$ . Jansen [13] focused on the LP of the form:

$$\begin{aligned} \max \quad & c^\top y \\ \text{s.t.} \quad & a_j^\top y \leq b_j \quad (\forall j \in [m]), \\ & y \in B, \end{aligned} \tag{8}$$

where  $a_j$  is an  $n$ -dimensional rational vector and  $b_j$  is a positive rational number for all  $j \in [m]$ , and  $B$  is a polyhedron defined by some inequalities such that the separation problem for  $B$  is solvable in polynomial time. Let  $K$  be the feasible region of (8). Then, Jansen [13] showed the following result. Suppose that we know an integer  $\psi$  for



which there is a system of inequalities with rational coefficients that has a solution set  $K$ , and such that the encoding length of each inequality is at most  $\psi$ . Also, suppose that we have an  $\alpha$ -approximation algorithm APP for finding  $\ell \in [m]$  maximizing  $a_j^\top y/b_j$ , where  $y \in B$  is fixed, in polynomial time with respect to  $n$  and  $\psi$ . Then, there is a polynomial-time  $\alpha$ -approximate separation algorithm for the polyhedron  $K$  with respect to  $n$ , and this yields an  $\alpha$ -approximation algorithm for (8) in polynomial time with respect to  $n$  and  $\psi$ . Moreover, there is an  $\alpha$ -approximation algorithm for the dual of (8) that runs in polynomial time in  $n$  and  $\psi$ .

We utilize the result of Jansen [13] as follows. We set  $c$  to be the all-one vector and  $B = \{z \in \mathbb{R}^n : z_k \geq 0 (\forall k \in [n])\}$ . Let  $\mu$  be the maximum encoding length of  $f_k(X)$  among  $k \in [n]$  and  $X \in \mathcal{I}$ . We set  $\psi = O(n \log \mu)$ . Indeed, each constraint in (5) has at most  $n$  coefficients of size bounded by  $\log \mu$ . In addition, suppose that we have an  $\alpha$ -approximation algorithm APP for the problem (7) in polynomial time with respect to  $n$  and the time to evaluate the value of  $f_i$  ( $i \in [n]$ ). According to the paper [13], an  $\alpha$ -approximate separation algorithm can be constructed in the same way as when the problem (7) can be solved exactly. We summarize this in Algorithm 1. When a violated constraint for a vector  $\hat{z}$  is not found, it returns that  $\alpha\hat{z}$  is feasible. This is correct because, when APP finds a solution  $\tilde{X}$ , we have  $1 \geq f^{\hat{z}}(\tilde{X}) \geq \alpha f^{\hat{z}}(X) = \alpha \cdot \sum_{i \in [n]} f_i(X)\hat{z}_i$  for all  $X \in \mathcal{I}$  by the construction. We can see that Algorithm 1 returns an answer in polynomial time in  $n$  and the running time of APP.

It is worth mentioning that this method is applicable thanks to the variable transformation from problem (3)–(4). In fact, the dual of LP (3) is not in the form of (8).

---

**Algorithm 1**  $\alpha$ -approximate separation algorithm for a vector  $z$

---

```

1: if  $z_i < 0$  for some  $i \in [n]$  then
2:   return “ $z_i \geq 0$  is violated”;
3: end if
4:  $\tilde{X} \leftarrow$  the  $\alpha$ -approximate solution returned by APP for  $\max_{X \in \mathcal{I}} f^z(X)$ ;
5: if  $f^z(\tilde{X}) > 1$  then
6:   return “ $\sum_{i \in [n]} f_i(X)z_i \leq 1$  is violated”;
7: else
8:   return “ $\alpha z$  is feasible”;
9: end if

```

---

Then, we solve the problem (4) as follows. First, solve the dual LP (5) using the ellipsoid algorithm with Algorithm 1. Let  $\mathcal{S}$  be the family of  $X \in \mathcal{I}$  corresponding to the violated constraints in (5) returned at Line 6 of Algorithm 1. Let  $\tilde{z}$  be the obtained feasible solution to the LP (5). Next, construct a restriction of the primal LP (4) by setting  $q_X = 0$  for all  $X \notin \mathcal{S}$ , i.e.,

$$\begin{aligned}
 \min \quad & \sum_{X \in \mathcal{S}} q_X \\
 \text{s.t.} \quad & \sum_{X \in \mathcal{S}} f_i(X)q_X \geq 1 \quad (\forall i \in [n]), \\
 & q_X \geq 0 \quad (\forall X \in \mathcal{S}).
 \end{aligned} \tag{9}$$

Then, solve LP (9) with a standard LP algorithm (such as the algorithm by Khachiyan [22]). We summarize this in Algorithm 2.

---

**Algorithm 2** Approximate algorithm for (4)

---

- 1: Solve the dual LP (5) using the ellipsoid algorithm with Algorithm 1.
  - 2: Let  $\mathcal{S} \subseteq \mathcal{I}$  be the family of the sets corresponding to the violated constraints in (5) returned at Line 6 of Algorithm 1;
  - 3: Let  $\tilde{z}$  be the obtained solution;
  - 4: Solve the restricted LP (9) with a standard LP algorithm;
  - 5: **return** an optimal solution  $(\tilde{q}_X)_{X \in \mathcal{S}}$  of LP (9);
- 

We remark that  $\mathcal{S}$  has polynomial size. The size of  $\mathcal{S}$  increases by at most one per one call for Algorithm 1. Moreover, by construction of the ellipsoid algorithm, the number of the calls for Algorithm 1 is bounded by a polynomial (in  $n$ ,  $\log \mu$ , and time to evaluate  $f_i$ 's).

Thus, we can rewrite the result of Jansen [13] as follows.

**Theorem 2** (Jansen [13]) *Suppose that there is an  $\alpha$ -approximation algorithm APP for (7). Then, Algorithm 1 is a polynomial-time  $\alpha$ -approximate separation algorithm for (5), and Algorithm 2 is an  $\alpha$ -approximation algorithm for (3) that runs in polynomial time in  $n$ ,  $\log \mu$  and the running time of APP.*

From the output  $(\tilde{q}_X)_{X \in \mathcal{S}}$  of Algorithm 2, we construct a solution  $((\tilde{p}_X)_{X \in \mathcal{S}}, \tilde{t})$  of LP (3) as  $\tilde{t} = 1 / \sum_{X \in \mathcal{S}} \tilde{q}_X$  and  $\tilde{p}_X = \tilde{t} \cdot \tilde{q}_X$  for  $X \in \mathcal{S}$ . We provide a theoretical guarantee on this solution.

**Theorem 3** *With an  $\alpha$ -approximation algorithm APP for the problem (7), we can obtain an  $\alpha$ -approximation algorithm for the robust optimization problem (1) in polynomial time with respect to  $n$ ,  $\log \mu$ , and the time to evaluate the value of  $f_i$  ( $i \in [n]$ ).*

**Proof** We see from Theorem 2 that Algorithm 2 terminates in polynomial time. The construction of  $((\tilde{p}_X)_{X \in \mathcal{S}}, \tilde{t})$  can be done also in polynomial time. Since  $(\tilde{q}_X)_{X \in \mathcal{S}}$  is feasible for (3),  $((\tilde{p}_X)_{X \in \mathcal{S}}, \tilde{t})$  is feasible for (3). Let OPT be the optimal value of (3). By construction, the optimal value of (3) is equal to  $1/\text{OPT}$ . By Theorem 2, we have  $\sum_{X \in \mathcal{S}} \tilde{q}_X \leq (1/\alpha) \cdot (1/\text{OPT})$ . Thus, it holds that  $\alpha \cdot \text{OPT} \leq 1 / \sum_{X \in \mathcal{S}} \tilde{q}_X = \tilde{t}$ . Therefore,  $((\tilde{p}_X)_{X \in \mathcal{S}}, \tilde{t})$  is an  $\alpha$ -approximate solution of LP (3). □

## 4 Application of Our Framework

In this section, we apply Theorem 3 for concrete problems.

### 4.1 Robust Additive Maximization

We consider the case where  $f_1, \dots, f_n$  are additive. We call such a problem *the additive robust optimization problem*. In this case, the function  $f^z$  defined as (6) is additive for

any  $z \in \mathbb{R}_+^n$ . It is well-known that the problem (7) is solvable in polynomial-time if the constraint  $\mathcal{I}$  is a matroid intersection (see, e.g., the book of Schrijver [31]). The maximum encoding length  $\mu$  of the function values can be obtained by evaluating the encoding length of  $f_k(\{e\})$  for each  $k \in [n]$  and  $e \in E$ . Hence, we obtain the following theorem.

**Theorem 4** *There exists a polynomial-time algorithm for the additive robust optimization problem if the constraint  $\mathcal{I}$  is defined from a matroid intersection.*

In the max–min fair randomized allocation problem, the constraint  $\mathcal{I} = \{X \subseteq [n] \times E : |X \cap \{(k, e) : e \in E\}| \leq 1 \ (\forall k \in [n])\}$  is a matroid. Hence, we obtain the following corollary from Theorem 4.

**Corollary 5** *There exists a polynomial-time algorithm for the max–min fair randomized allocation problem when every agent has an additive utility function.*

Let us consider a general case in which  $\mathcal{I}$  is a  $\mu$ -matroid intersection. It is NP-hard to maximize an additive function subject to a  $\mu$ -matroid intersection constraint if  $\mu \geq 3$  [10]. Hence, it is also NP-hard to solve the additive robust optimization subject to a  $\mu$ -matroid intersection constraint if  $\mu \geq 3$ . Nevertheless, by utilizing  $1/(\mu - 1 + \epsilon)$ -approximation algorithm [27] ( $\epsilon > 0$ ), we obtain the following theorem.

**Theorem 6** *For any fixed  $\epsilon > 0$ , there exists a  $1/(\mu - 1 + \epsilon)$ -approximation algorithm for the additive robust optimization problem if the constraint  $\mathcal{I}$  is a  $\mu$ -matroid intersection.*

When  $\mathcal{I}$  is a knapsack constraint, the additive robust optimization problem is NP-hard. However, an FPTAS is known for the knapsack problem.

**Theorem 7** *There exists an FPTAS for the additive robust optimization problem if the constraint  $\mathcal{I}$  is defined from a knapsack constraint.*

Furthermore, our method can be applied to the robust shortest  $s$ – $t$  path problem. In the problem, we are given a directed graph  $G = (V, E)$ , a source  $s \in V$ , a destination  $t \in V$ , and lengths  $\ell_k : E \rightarrow \mathbb{R}_{++}$  ( $k \in [n]$ ), where  $\mathbb{R}_{++}$  is the set of positive reals. Let  $\mathcal{I} \subseteq 2^E$  be the set of  $s$ – $t$  paths and  $f_k(X) = -\sum_{e \in X} \ell_k(e)$  for  $k \in [n]$ . Then, the robust shortest path problem is to find a probability distribution over  $s$ – $t$  paths  $p \in \Delta(\mathcal{I})$  that maximizes  $\min_{k \in [n]} \sum_{X \in \mathcal{I}} p_X f_k(X)$ . As the shortest  $s$ – $t$  path problem is solvable in polynomial time, the following theorem holds.

**Theorem 8** *There exists a polynomial-time algorithm for the robust shortest  $s$ – $t$  path problem.*

## 4.2 Robust Monotone Submodular Maximization

We consider the case where  $f_1, \dots, f_n$  are monotone submodular. We call such a problem *the monotone submodular robust optimization problem*. For any  $z \in \mathbb{R}_+^n$ , the function  $f^z$  defined by (6) is monotone submodular since a nonnegative linear

combination of monotone submodular functions is a monotone submodular function. Thus, (7) is an instance of the monotone submodular function maximization problem. We assume that the maximum encoding length  $\mu$  of the function values is known. There exists  $(1 - 1/e)$ -approximation algorithms for this problem under a knapsack constraint [32] or under a matroid constraint [3, 6]. When  $\mathcal{I}$  is defined from a knapsack constraint or a matroid, Theorem 3 implies that Algorithm 2 finds a  $(1 - 1/e)$ -approximate solution to (1) by using the existing approximation algorithms as APP in Algorithm 1.

**Theorem 9** *There exists a  $(1 - 1/e)$ -approximation algorithm for the monotone submodular robust optimization problem if the constraint  $\mathcal{I}$  is defined from a knapsack constraint or a matroid.*

Recall that the max–min fair randomized allocation problem is to solve  $\max_{p \in \Delta(\mathcal{I})} \min_{k \in [n]} \sum_{X \in \mathcal{I}} p_X \cdot f_k(X)$  where  $\mathcal{I} = \{X \subseteq [n] \times E : |X \cap \{(k, e) : e \in E\}| \leq 1 (\forall k \in [n])\}$  and  $f_k(X) = u_k(\{e \in E : (k, e) \in X\})$ . Here,  $\mathcal{I}$  is a (partition) matroid. Additionally,  $f_k$  is a monotone submodular function if the utility function  $u_k$  is a monotone submodular function. Hence, from Theorem 9, we obtain the following corollary.

**Corollary 10** *There exists a  $(1 - 1/e)$ -approximation algorithm for the max–min fair randomized allocation problem when every agent has a monotone submodular utility function.*

In addition, there exists a  $1/(\mu + \epsilon)$ -approximation algorithm for the monotone submodular function maximization problem under a  $\mu$ -matroid intersection constraint [27]. Hence, the following theorem holds by Theorem 3.

**Theorem 11** *For any fixed  $\epsilon > 0$ , there exists a  $1/(\mu + \epsilon)$ -approximation algorithm for the monotone submodular robust optimization problem if the constraint  $\mathcal{I}$  is defined from a  $\mu$ -matroid intersection.*

### 4.3 MCRP with a Knapsack Constraint

Now, we apply Theorem 3 to MCRP with a knapsack constraint. In MCRP, suppose that we are given a set of items  $E = \{1, 2, \dots, n\}$  with size  $s : E \rightarrow \mathbb{R}_{++}$ , and value  $v : E \rightarrow \mathbb{R}_{++}$ , and a knapsack capacity  $c$ . Without loss of generality, we assume that  $s(e) \leq c$  for every  $e \in E$ . The set of feasible solutions is  $\mathcal{I} = \{X \subseteq E : s(X) \leq c\}$ . Let us denote

$$v_{\leq k}(X) = \max \left\{ \sum_{e \in X'} v(e) : |X'| \leq k, X' \subseteq X \right\}$$

and let  $X_k^* \in \arg \max_{X \in \mathcal{I}} v_{\leq k}(X)$  for  $k \in [n]$ . The *cardinality robustness* of a solution  $X \in \mathcal{I}$  is defined as

$$\min_{k \in [n]} \frac{v_{\leq k}(X)}{v_{\leq k}(X_k^*)}.$$

MCRP with a knapsack constraint is to find a randomized solution  $p \in \Delta(\mathcal{I})$  with the maximum cardinality robustness under the knapsack constraint. We show that this problem is NP-hard but admits an FPTAS; the proof of the hardness is deferred to Appendix A. We note that Kakimura et al. [15] proved that its deterministic version is NP-hard but also admits an FPTAS. However, their results and ours do not directly imply each other. Because the optimal values in the deterministic and randomized setting may be significantly different, even an optimal deterministic solution may not be a good approximate randomized solution. Furthermore, converting an optimal randomized solution into a deterministic solution with a good approximation ratio is a challenging task.

We can see MCRP as the robust optimization problem (1) with constraint  $\mathcal{I}$  and objective functions

$$f_k(X) = \frac{v_{\leq k}(X)}{v_{\leq k}(X_k^*)} \quad (k \in [n]).$$

We note that  $\max_{X \in \mathcal{I}} f_k(X) = f_k(X_k^*) = 1$  for each  $k \in [n]$ .

To construct an FPTAS for the problem (7), we need to evaluate  $f_k(X)$ , and we do this by using the following result.

**Lemma 12** Caprara et al. [4] *There exists an FPTAS to compute the value of  $v_{\leq k}(X_k^*)$  for each  $k \in [n]$ .*

Then we construct our FPTAS for (7) based on the dynamic programming.

**Lemma 13** *For a given  $z \in \mathbb{R}_+^n$ , there exists an FPTAS to solve the problem  $\max_{X \in \mathcal{I}} f^z(X)$ .*

**Proof** Without loss of generality, we may assume  $z \neq 0$ . Let  $\epsilon$  be a positive real and let  $v$  be the optimal value, i.e.,  $v = \max_{X \in \mathcal{I}} f^z(X) = \max_{X \in \mathcal{I}} \sum_{k \in [n]} \frac{z_k \cdot v_{\leq k}(X)}{v_{\leq k}(X_k^*)}$ . For each  $k \in [n]$ , let  $v_k^*$  be a value such that  $v_{\leq k}(X_k^*) \leq v_k^* \leq v_{\leq k}(X_k^*)/(1 - \epsilon)$ . We can compute such values in polynomial time with respect to  $n$  and  $1/\epsilon$  by Lemma 12. For each  $X \in \mathcal{I}$ , let  $g(X) = \sum_{k \in [n]} \frac{z_k \cdot v_{\leq k}(X)}{v_k^*}$ . Then by the definition of  $v_k^*$ , we have

$$(1 - \epsilon)v \leq \max_{X \in \mathcal{I}} g(X) \leq v.$$

Hence, it suffices to solve  $\max_{X \in \mathcal{I}} g(X)$  to know an approximate solution to problem (7).

A simple dynamic programming based algorithm for maximizing  $g$  runs in time depending on the value of  $g$ . By appropriately scaling the value of  $g$  according to  $\epsilon$ , we will obtain a solution whose objective value is at least  $(1 - 2\epsilon)v$  in polynomial time with respect to both  $n$  and  $1/\epsilon$ . Let  $\kappa = \lceil \frac{n^2}{\epsilon} \rceil$ . For each  $X = \{e_1, \dots, e_\ell\}$  with  $v(e_1) \geq \dots \geq v(e_\ell)$ , we define

$$\bar{g}(X) = \sum_{k=1}^{\ell} \left[ \left( \sum_{i=k}^n \frac{z_i}{v_i^*} \right) v(e_k) \cdot \frac{\kappa}{\sum_{i \in [n]} z_i} \right] \cdot \frac{\sum_{i \in [n]} z_i}{\kappa}.$$

We note that

$$g(X) = \sum_{k=1}^n \frac{z_k \cdot v_{\leq k}(X)}{v_k^*} = \sum_{k=1}^n \frac{z_k}{v_k^*} \sum_{i=1}^{\min\{k, \ell\}} v(e_i) = \sum_{k=1}^{\ell} \left( \sum_{i=k}^n \frac{z_i}{v_i^*} \right) v(e_k).$$

Thus, we have  $\bar{g}(X) \leq g(X) \leq \bar{g}(X) + n \cdot (\sum_{i \in [n]} z_i / \kappa)$  for all  $X$ .

In addition, we have

$$v = \max_{X \in \mathcal{I}} f^z(X) \geq \max_{X \in \{X_1^*, \dots, X_n^*\}} \sum_{k \in [n]} \frac{z_k \cdot v_{\leq k}(X)}{v_{\leq k}(X_k^*)} \geq \max_{k \in [n]} z_k \geq \frac{\sum_{k \in [n]} z_k}{n}.$$

Hence, we obtain

$$\begin{aligned} \max_{X \in \mathcal{I}} \bar{g}(X) &\geq \max_{X \in \mathcal{I}} g(X) - \frac{n \sum_{k \in [n]} z_k}{\kappa} \geq \max_{X \in \mathcal{I}} g(X) - \epsilon \cdot \frac{\sum_{k \in [n]} z_k}{n} \\ &\geq (1 - \epsilon)v - \epsilon \cdot v = (1 - 2\epsilon)v. \end{aligned}$$

This implies that there exists an FPTAS to solve  $\max_{X \in \mathcal{I}} f^z(X)$  if we can compute  $\max_{X \in \mathcal{I}} \bar{g}(X)$ , or equivalently  $\max_{X \in \mathcal{I}} (\kappa / \sum_{k \in [n]} z_k) \cdot \bar{g}(X)$  in polynomial time in  $n$  and  $1/\epsilon$ .

Here, we assume that  $E = \{1, 2, \dots, n\}$  and  $v(1) \geq v(2) \geq \dots \geq v(n)$ . Let  $\tau(\zeta, \xi, \phi) = \min\{s(X) : X \subseteq \{1, \dots, \zeta\}, |X| = \xi, (\kappa / \sum_{k \in [n]} z_k) \cdot \bar{g}(X) = \phi\}$ . Then we can compute the value of  $\tau(\zeta, \xi, \phi)$  by the following equation:

$$\tau(\zeta, \xi, \phi) = \min \left\{ \tau(\zeta - 1, \xi, \phi), s(\zeta) + \tau(\zeta - 1, \xi - 1, \phi - \lfloor (\sum_{i=\zeta}^n \frac{z_i}{v_i^*}) \frac{v(\zeta) \cdot \kappa}{\sum_{i \in [n]} z_i} \rfloor) \right\}.$$

We see that  $\max_{X \in \mathcal{I}} \bar{g}(X) = \max\{\phi : \tau(n, \xi, \phi) \leq c, 0 \leq \xi \leq n\}$ , and an optimal solution is obtained straightforwardly.

It remains to discuss the running time. For all  $X$ , the value  $(\kappa / \sum_{k \in [n]} z_k) \cdot \bar{g}(X)$  is an integer in  $[0, \kappa]$  because  $\bar{g}(X) \leq \sum_{k \in [n]} \frac{z_k \cdot v_{\leq k}(X)}{v_{\leq k}(X_k^*)} \leq \sum_{k \in [n]} z_k$ . Hence, there exist  $\kappa + 1$  possibilities of  $\phi$ . Therefore, we can compute  $\max_{X \in \mathcal{I}} \bar{g}(X)$  in  $O(n^2 \kappa) = O(n^4 / \epsilon)$  time.  $\square$

Note that the maximum encoding length  $\mu$  of the function values  $f_k(X)$  can be obtained by summing up the encoding length of  $v(e)$  for each  $e \in E$ . Therefore, we obtain the following theorem by combining Theorem 3 and Lemma 13.

**Theorem 14** *There exists an FPTAS to solve MCRP with a knapsack constraint.*

**Acknowledgements** The first author is supported by JSPS KAKENHI Grant Nos. JP16K16005, JP20K19739, JST ACT-I Grant No. JPMJPR17U7, JST PRESTO Grant No. JPMJPR2122, and Value Exchange Engineering, a joint research project between R4D, Mercari, Inc. and the RIISE. The second author is supported by JST ERATO Grant No. JPMJER1201, Japan, and JSPS KAKENHI Grant Nos. JP17K12646, JP21K17708, and JP21H03397, Japan.

**Author Contributions** All authors wrote and reviewed the manuscript.

**Funding** Open access funding provided by The University of Tokyo.

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## Appendix A: NP-Hardness of MCRP for the Knapsack Problem

In this section, we show that MCRP is NP-hard. We give a reduction from the partition problem with restriction that two partitioned subsets are restricted to have equal cardinality, which is an NP-complete problem [10]. Given an even number of positive integers  $a_1, a_2, \dots, a_{2n}$ , the problem is to find a subset  $I \subseteq [2n]$  such that  $|I| = n$  and  $\sum_{i \in I} a_i = \sum_{i \in [2n] \setminus I} a_i$ . Recall that  $[2n] = \{1, 2, \dots, 2n\}$ .

**Theorem 15** *It is NP-hard to find a solution  $p \in \Delta(\mathcal{I})$  with the maximum cardinality robustness for the knapsack problem.*

**Proof** Let  $(a_1, a_2, \dots, a_{2n})$  be an instance of the partition problem. Without loss of generality, we assume that  $a_1 \geq a_2 \geq \dots \geq a_{2n} (\geq 1)$  and  $n \geq 4$ . Define  $A = \sum_{i=1}^{2n} a_i/2$ . We construct the following instance of MCRP for the knapsack problem:

- $E = \{0, 1, \dots, 2n\}$ ,
- $s(0) = A + 2n^2a_1, v(0) = 2(2n + 1)a_1$ ,
- $s(i) = v(i) = a_i + 2na_1$  ( $i = 1, \dots, 2n$ ),
- $c = \sum_{i=1}^{2n} s(i) = 2A + 4n^2a_1$ .

Note that  $c/2 = s(0) \geq s(1) \geq \dots \geq s(2n)$  and  $v(0) = 2v(1) > v(1) \geq \dots \geq v(2n)$ . We denote by  $\mathcal{I}$  the set of knapsack solutions.

Let

$$\alpha = \frac{1}{4} \cdot \frac{3c - 2v(0)}{c - v(0)} = \frac{1}{4} \cdot \frac{3(A + 2n^2a_1) - 2(2n + 1)a_1}{A + 2n^2a_1 - (2n + 1)a_1} \in [0, 1].$$

We claim that this instance has a randomized  $\alpha$ -robust solution  $p \in \Delta(\mathcal{I})$  if and only if the partition problem instance has a solution. Recall that  $p \in \Delta(\mathcal{I})$  is called  $\alpha$ -robust if  $\sum_{X \in \mathcal{I}} p_X \cdot v_{\leq k}(X) \geq \alpha \cdot \max_{Y \in \mathcal{I}} v_{\leq k}(Y)$  for any  $k \in [E]$ . For  $p \in \Delta(\mathcal{I})$ , we denote  $v_{\leq k}(p) = \sum_{X \in \mathcal{I}} p_X \cdot v_{\leq k}(X)$ .

Suppose that  $I \subseteq [2n]$  is a solution to the partition problem instance, i.e.,  $|I| = n$  and  $\sum_{i \in I} a_i = \sum_{i \in [2n] \setminus I} a_i (= A)$ . Let

$$r = \frac{1}{2} \cdot \frac{c - 2v(0)}{c - v(0)} = \frac{1}{2} \cdot \frac{A + 2n^2a_1 - 2(2n + 1)a_1}{A + 2n^2a_1 - (2n + 1)a_1}.$$

We note that  $r = 2(1 - \alpha)$ . We define a randomized solution  $p \in \Delta(\mathcal{I})$  by  $p_Y = r$  if  $Y = [2n]$ ,  $p_Y = 1 - r$  if  $Y = \{0\} \cup I$ , and  $p_Y = 0$  otherwise. We observe that  $[2n] \in \mathcal{I}$  by the definition of  $c$ , and that  $\{0\} \cup I \in \mathcal{I}$  because  $s(0) = c/2$  and  $\sum_{i \in I} s(i) = c/2$ . We claim that  $p$  is an  $\alpha$ -robust solution by showing that  $v_{\leq k}(p)/v_{\leq k}(X_k^*) \geq \alpha$  for all  $k = 1, \dots, 2n + 1$ . Recall that  $X_k^* \in \arg \max_{X \in \mathcal{I}} v_{\leq k}(X)$ .

First, take arbitrarily  $k \in \{1, \dots, n + 1\}$ . We have  $v_{\leq k}(X_k^*) \leq \sum_{i=0}^{k-1} v(i) \leq (k + 1)v(1)$ . Moreover,  $v_{\leq k}([2n]) = \sum_{i=1}^k v(i) \geq v(1) + (k - 1)2na_1$ , and  $v_{\leq k}(\{0\} \cup I) \geq v(0) + (k - 1)2na_1$ . Thus, we see that

$$\begin{aligned} \frac{v_{\leq k}(p)}{v_{\leq k}(X_k^*)} &\geq \frac{(r(2n + 1)a_1 + (1 - r) \cdot 2(2n + 1)a_1) + (k - 1) \cdot 2na_1}{(k + 1)(2n + 1)a_1} \\ &= \frac{2n}{2n + 1} + \left( \frac{2}{2n + 1} - r \right) \cdot \frac{1}{k + 1}. \end{aligned}$$

Here, because  $a_1 \leq A \leq 2na_1$  and  $n \geq 4$ , it follows that

$$\alpha = \frac{1}{4} \cdot \frac{3(A + 2n^2a_1) - 2(2n + 1)a_1}{A + 2n^2a_1 - (2n + 1)a_1} \leq \frac{1}{4} \cdot \frac{6n^2 - 4n + 1}{2n(n - 1)} \leq \frac{2n}{2n + 1}. \tag{10}$$

Then, we see that  $\frac{2}{2n+1} - r \leq 0$  since  $r = 2(1 - \alpha) \geq 2(1 - \frac{2n}{2n+1}) = \frac{2}{2n+1}$ . This implies that

$$\frac{v_{\leq k}(p)}{v_{\leq k}(X_k^*)} \geq \frac{2n}{2n + 1} + \left( \frac{2}{2n + 1} - r \right) \cdot \frac{1}{2} = 1 - \frac{r}{2} = \alpha.$$

Next, assume that  $k = n + 2$ . We claim that  $v_{\leq k}(X_k^*) \leq (n + 2) \cdot (2n + 1)a_1$ . To describe this, let  $X$  be any set in  $\mathcal{I}$  with  $0 \in X$ . Since  $s(0) = c/2$  and  $s(i) = v(i)$  for all  $i \geq 1$ , we have  $\sum_{i \in X} v(i) \leq v(0) + c/2 = v(0) + \sum_{i \in I} v(i) \leq v(0) + \sum_{i \in I} v(1) \leq (n + 2)v(1)$ . On the other hand, for any set  $X \in \mathcal{I}$  with  $0 \notin X$ , we have  $v_{\leq k}(X_k^*) \leq \sum_{i=1}^{n+2} v(i) \leq (n + 2)v(1)$ . Hence,  $v_{\leq k}(X_k^*) \leq (n + 2) \cdot (2n + 1)a_1$ . In addition, we observe that  $v_{\leq k}([2n]) = \sum_{k=1}^{n+2} v(i) \geq (n + 2) \cdot 2na_1$  and  $v_{\leq k}(\{0\} \cup I) \geq v(0) + n \cdot 2na_1 \geq (n + 2)2na_1$ . Thus,  $v_{\leq k}(p) \geq (n + 2) \cdot 2na_1$ . These facts together with (10) imply that

$$\frac{v_{\leq k}(p)}{v_{\leq k}(X_k^*)} \geq \frac{(n + 2)2na_1}{(n + 2)(2n + 1)a_1} = \frac{2n}{2n + 1} \geq \alpha.$$

Let us consider the case when  $k \in \{n + 3, \dots, 2n - 1\}$ . For any  $X \in \mathcal{I}$  with  $0 \in X$ , we observe that  $v_{\leq k}(X) \leq (n + 2)(2n + 1)a_1 \leq 2n(n + 3)a_1 \leq \sum_{i=1}^k v(i) = v_{\leq k}([2n])$ ,



where the first inequality holds by a similar argument to the one in the above case. Thus, we see that  $v_{\leq k}(X_k^*) = v_{\leq k}([2n])$ . Because  $v_{\leq k}([2n]) \leq v_{\leq k+1}([2n])$  and  $v_{\leq k}(\{0\} \cup I) = v_{\leq k+1}(\{0\} \cup I)$ , it holds that

$$\begin{aligned} \frac{v_{\leq k}(p)}{v_{\leq k}(X_k^*)} &= \frac{r \cdot v_{\leq k}([2n]) + (1 - r)v_{\leq k}(\{0\} \cup I)}{v_{\leq k}([2n])} \\ &= r + \frac{(1 - r)v_{\leq k}(\{0\} \cup I)}{v_{\leq k}([2n])} \\ &\geq r + \frac{(1 - r)v_{\leq k}(\{0\} \cup I)}{v_{\leq k+1}([2n])} \\ &= \frac{r \cdot v_{\leq k+1}([2n]) + (1 - r)v_{\leq k}(\{0\} \cup I)}{v_{\leq k+1}([2n])} \\ &= \frac{v_{\leq k+1}(p)}{v_{\leq k+1}(X_{k+1}^*)}. \end{aligned}$$

Hence, it remains to show that  $\frac{v_{\leq k}(p)}{v_{\leq k}(X_k^*)} \geq \alpha$  when  $k = 2n$  and  $k = 2n + 1$ . It is clear that  $v_{\leq 2n}(p) = v_{\leq 2n+1}(p)$  since  $v_{\leq 2n}([2n]) = v_{\leq 2n+1}([2n])$  and  $v_{\leq 2n}(\{0\} \cup I) = v_{\leq 2n+1}(\{0\} \cup I)$ . We have also  $v_{\leq 2n+1}(X_k^*) = v_{\leq 2n}(X_k^*) = \sum_{i=1}^{2n} v(i) = c$  because  $\{0, 1, \dots, 2n\} \notin \mathcal{I}$ . Thus, it follows that

$$\begin{aligned} \frac{v_{\leq 2n+1}(p)}{v_{\leq 2n+1}(X_k^*)} &= \frac{v_{\leq 2n}(p)}{v_{\leq 2n}(X_{2n}^*)} = \frac{rc + (1 - r)(v(0) + c/2)}{c} \\ &= \frac{1 + r}{2} + \frac{(1 - r)v(0)}{c} = 1 - \frac{r}{2} = \alpha, \end{aligned}$$

where the last second equation holds because  $r = (c - 2v(0))/2(c - v(0)) \Leftrightarrow (1 - r)v(0) = c(1/2 - r)$ . Therefore,  $p$  is  $\alpha$ -robust.

It remains to prove that if the partition problem instance has no solution, then there exists no  $\alpha$ -robust solution. Let  $p \in \Delta(\mathcal{I})$  be a solution and let  $r = \sum_{X:0 \notin X \in \mathcal{I}} pX$ . To show a contradiction, we assume that  $p$  is  $\alpha$ -robust. Then it must hold that

$$\frac{v_{\leq 1}(p)}{v_{\leq 1}(X_1^*)} = \frac{r \cdot v(1) + (1 - r) \cdot v(0)}{v(0)} = \frac{(2 - r)v(1)}{2v(1)} = \frac{2 - r}{2} \geq \alpha = \frac{1}{4} \cdot \frac{3c - 2v(0)}{c - v(0)}$$

and hence

$$r \leq \frac{1}{2} \cdot \frac{c - 2v(0)}{c - v(0)}.$$

This implies that  $r < 1$  and  $p_X > 0$  for some  $X \in \mathcal{I}$  with  $0 \in X$ .

On the other hand, we claim that  $v_{\leq 2n+1}(p)/v_{\leq 2n+1}(X_{2n+1}^*) < \alpha$ . For any  $X \in \mathcal{I}$  with  $0 \notin X$ , we observe that  $v_{\leq 2n+1}(X) \leq \sum_{i=1}^{2n} v(i) = c$ . Take an arbitrary set  $Y$  such that  $Y \cup \{0\} \in \mathcal{I}$ . It holds that  $v_{\leq 2n+1}(Y \cup \{0\}) = v(0) + \sum_{i \in Y} s(i) \leq v(0) + c/2$ .

We claim that  $\sum_{i \in Y} s(i) \neq c/2$ . If  $|Y| > n$ , then  $\sum_{i \in Y} s(i) \geq (n + 1) \cdot 2na_1 > na_1 + 2n^2a_1 \geq A + 2n^2a_1 = c/2$ . If  $|Y| < n$ , then  $\sum_{i \in Y} s(i) \leq (n - 1) \cdot (2n + 1)a_1 \leq -n - 1 + 2n^2a_1 < A + 2n^2a_1 = c/2$ . If  $|Y| = n$ , then  $\sum_{i \in Y} s(i) = \sum_{i \in Y} a_i + 2n^2a_1 \neq A + 2n^2a_1 (= c/2)$  since the partition problem instance has no solution. Thus we see that  $v_{\leq 2n+1}(p) < r \cdot c + (1 - r) \cdot (v(0) + c/2)$  by  $r < 1$ . Since  $v_{\leq 2n+1}(X_{2n+1}^*) = c$  and  $r \leq \frac{1}{2} \cdot \frac{c - 2v(0)}{c - v(0)}$ , we have

$$\begin{aligned} \frac{v_{\leq 2n+1}(p)}{v_{\leq 2n+1}(X_{2n+1}^*)} &< \frac{rc + (1 - r)(v(0) + c/2)}{c} \\ &= \frac{r(c - 2v(0))}{2c} + \frac{c + 2v(0)}{2c} \\ &\leq \frac{1}{4} \cdot \frac{(c - 2v(0))^2 + 2(c - v(0))(c + 2v(0))}{c(c - v(0))} \\ &= \frac{1}{4} \cdot \frac{3c - 2v(0)}{c - v(0)} = \alpha. \end{aligned}$$

This implies that  $p$  cannot be  $\alpha$ -robust.

Therefore, there exists a randomized  $\alpha$ -robust solution  $p \in \Delta(\mathcal{I})$  if and only if the partition problem instance has a solution. This completes the proof.  $\square$

## References

1. Aissi, H., Bazgan, C., Vanderpooten, D.: Min–max and min–max regret versions of combinatorial optimization problems: a survey. *EJOR* **197**(2), 427–438 (2009)
2. Bowles, S.: *Microeconomics: Behavior, Institutions, and Evolution*. Princeton University Press, Princeton (2009)
3. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint. In: *IPCO*, vol. 7, pp. 182–196. Springer (2007)
4. Caprara, A., Kellerer, H., Pferschy, U., Pisinger, D.: Approximation algorithms for knapsack problems with cardinality constraints. *EJOR* **123**(2), 333–345 (2000)
5. Chen, R.S., Lucier, B., Singer, Y., Syrgkanis, V.: Robust optimization for non-convex objectives. In: *Proceedings of the NIPS*, pp. 4708–4717 (2017)
6. Filmus, Y., Ward, J.: A tight combinatorial algorithm for submodular maximization subject to a matroid constraint. In: *Proceedings of the FOCS*, 659–668. IEEE (2012)
7. Freund, Y., Schapire, R.E.: Adaptive game playing using multiplicative weights. *GEB* **29**(1–2), 79–103 (1999)
8. Fujishige, S.: *Submodular Functions and Optimization*, vol. 58. Elsevier, Amsterdam (2005)
9. Fujita, R., Kobayashi, Y., Makino, K.: Robust Matchings and Matroid Intersections **27**, 1234–1256 (2013)
10. Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York (1979)
11. Grötschel, M., Lovász, L., Schrijver, A.: *Geometric Algorithms and Combinatorial Optimization*, vol. 2. Springer, Berlin (2012)
12. Hassin, R., Rubinfeld, S.: Robust matchings. *SIDMA* **15**(4), 530–537 (2002)
13. Jansen, K.: Approximate strong separation with application in fractional graph coloring and preemptive scheduling. *Theor. Comput. Sci.* **302**(1–3), 239–256 (2003)
14. Kakimura, N., Makino, K.: Robust independence systems. *SIDMA* **27**(3), 1257–1273 (2013)
15. Kakimura, N., Makino, K., Seimi, K.: Computing knapsack solutions with cardinality robustness. *Jpn. J. Ind. Appl. Math.* **29**(3), 469–483 (2012)

16. Kale, S.: Efficient algorithms using the multiplicative weights update method. Ph.D. thesis, Princeton University (2007)
17. Kasperski, A., Zieliński, P.: Robust Discrete Optimization Under Discrete and Interval Uncertainty: A Survey, pp. 113–143. Springer, Berlin (2016)
18. Kawase, Y., Sumita, H.: Randomized strategies for robust combinatorial optimization. In: Proceedings of the AAAI (2019)
19. Kawase, Y., Sumita, H.: On the max–min fair stochastic allocation of indivisible goods. In: Proceedings of the AAAI (2020)
20. Kawase, Y., Miyauchi, A., Sumita, H.: Stochastic solutions for dense subgraph discovery in multilayer networks. In: Proceedings of the WSDM, pp. 886–894 (2023)
21. Kellerer, H., Mansini, R., Speranza, M.G.: Two linear approximation algorithms for the subset-sum problem. *EJOR* **120**, 289–296 (2000)
22. Khachiyan, L.: Polynomial algorithms in linear programming. *USSR Comput. Math. Math. Phys.* **20**(1), 53–72 (1980)
23. Kobayashi, Y., Takazawa, K.: Randomized strategies for cardinality robustness in the knapsack problem. *Theor. Comput. Sci.* **699**, 53–62 (2016)
24. Krause, A., Golovin, D.: Submodular function maximization. In: Tractability: Practical Approaches to Hard Problems (to appear). Cambridge University Press, London (2014)
25. Krause, A., McMahan, H.B., Guestrin, C., Gupta, A.: Robust submodular observation selection. *J. Mach. Learn. Res.* **9**, 2761–2801 (2008)
26. Krause, A., Roper, A., Golovin, D.: Randomized sensing in adversarial environments. In: Proceedings of the IJCAI, vol. 22, pp. 2133–2139 (2011)
27. Lee, J., Sviridenko, M., Vondrák, J.: Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.* **35**(4), 795–806 (2010)
28. Matuschke, J., Skutella, M., Soto, J.A.: Robust randomized matchings. *Math. Oper. Res.* **43**(2), 675–692 (2018). <https://doi.org/10.1287/moor.2017.0878>
29. Nisan, N., Roughgarden, T., Tardos, É., Vazirani, V.V.: Algorithmic Game Theory. Cambridge University Press, London (2007)
30. Orlin, J.B., Schulz, A.S., Udwani, R.: Robust monotone submodular function maximization. In: Proceedings of the IPCO, pp. 312–324. Springer (2016)
31. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Algorithms and Combinatorics. Springer, Berlin (2003)
32. Sviridenko, M.: A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.* **32**(1), 41–43 (2004)
33. Tambe, M.: Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned. Cambridge University Press, London (2011)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.