Pesquisa Operacional (2024) 44: e281853 p.1-32
doi: 10.1590/0101-7438.2023.043.00281853
© 2024 Brazilian Operations Research Society
Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
ARTICLES

BIOBJECTIVE INTEGER STOCHASTIC OPTIMIZATION OVER THE INTEGER STOCHASTIC EFFICIENT SET

Ilias Badaoui[1*], Mustapha Moulaï[2],
Yacine Chaiblaine[3] and Djamal Chaabane[4]

Received December 29, 23 / Accepted June 11, 24

ABSTRACT. In various Multi-objective Programming (MOP) problems, decision-makers are often faced with a large set of efficient solutions, presenting a challenge in discerning and selecting the best solutions within this broad set. To overcome this, decision-makers can tune and optimize their preference function over this efficient set to identify the optimal solutions that align with their preferences. Most existing methods address such problems in the deterministic case, while we aim to tackle a new challenge by dealing with these problems in a stochastic environment, which increases their complexity. Indeed, we focus in this study on proposing an exact method to optimize two preference functions over the efficient set of a Multi-objective Stochastic Integer Linear Programming (MOSILP) problem in order to find the best compromise solutions without enumerating all elements of this efficient set. The proposed method is based on the combination of two techniques: the first one is called the L-shaped method, while the second one is an adaptation of the branch-and-bound strategy by reinforcing it with efficient cuts and tests, which allows the method to remove a large number of inefficient solutions within the search tree. Moreover, the method exhibits adaptability in addressing the general case involving multiple preference functions. A didactic example is presented for illustration, followed by a computational study evaluating the method's efficacy and performance by solving randomly generated instances.

Keywords: multi-objective programming, bi-objective programming, stochastic programming, optimization over an efficient set, two-stage stochastic programming, branch-and-cut.

*Corresponding author

[1]USTHB, LaROMaD, Department of Operational Research, Bp 32 El Alia, 16111 Algiers, Algeria – E-mail: iliesbadi@hotmail.fr – https://orcid.org/0000-0003-0189-327X

[2]USTHB, LaROMaD, Department of Operational Research, Bp 32 El Alia, 16111 Algiers, Algeria – E-mail: mmoulai@usthb.dz – https://orcid.org/0000-0002-2013-356X

[3]USTHB, LaROMaD, Department of Operational Research, Bp 32 El Alia, 16111 Algiers, Algeria – E-mail: ychaiblaine@usthb.dz – https://orcid.org/0000-0002-8935-1020

[4]USTHB, AMCD&RO, Department of Operational Research, Bp 32 El Alia, 16111 Algiers, Algeria – E-mail: chaabane_dj@yahoo.fr – https://orcid.org/0000-0002-7722-311X

INTRODUCTION

Multi-objective Programming (MOP) plays an important role in dealing with real-world problems especially when multiple decision-makers or stakeholders are involved in the same problem, making it difficult to decide which single goal to achieve since there may be conflicting objectives that need to be balanced. Imagine a situation where one group wants to maximize profits, while another wants to minimize costs, which makes balancing these conflicting objectives quite challenging. Furthermore, MOP can handle effectively both deterministic environments which refer to situations where all data are known with certainty, such as the works of Obal et al. (2013); Shidpour et al. (2013); Cao et al. (2018); Ren et al. (2021); Zhang et al. (2021); Halffmann et al. (2022); Motahari et al. (2023), and stochastic cases, which present intricate challenges due to the involvement of uncertain or random parameters, including works of Goicoechea et al. (1976); Urli & Nadeau (1990); Abdelaziz & Masri (2010); Ben Abdelaziz & Masmoudi (2012); Bozorgi-Amiri et al. (2013); Ramezani et al. (2013); Moayedi & Sadeghian (2023).

The biggest challenge in the MOP realm is the necessity of balancing the multiple conflicting objectives, this challenge can be tackled by identifying a set of solutions known as an efficient solution set. The size of this efficient set is influenced by both the complexity of the problem and the computing resources available, and as the problem size grows, the efficient set has the potential to expand significantly. Over the last few years, numerous authors have addressed this challenge of identifying the efficient set in deterministic cases, for example, Ecker & Kouada (1978); Sylva & Crema (2004); Jahanshahloo et al. (2004); Özlen & Azizoğlu (2009); Lokman & Köksalan (2013); Kirlik & Sayın (2014); Rasmi & Türkay (2019); Tamby & Vanderpooten (2021). In contrast, for stochastic cases, the presence of these stochastic parameters heightens the complexity of the problem. It is noteworthy that a limited number of studies have addressed this specific challenge, such as Abbas & Bellahcene (2006); Amrouche & Moulaï (2012).

In numerous real-world problems, decision-makers often find themselves in front of a multitude of efficient solutions, which places them in difficult situations where they must carefully evaluate and select the best solution among all these efficient solutions. One effective approach to address this problem is to employ optimization techniques that allow decision-makers to fine-tune their preference function, and optimize it over the set of efficient solutions to ultimately obtain the optimal solution that aligns with their preferences. It is worth noting that the classic approach of enumerating all the efficient solutions is not recommended, due to its impracticality and the significant computational complexity it entails.

Optimization over the efficient set is a well-established and dynamic research field that is constantly evolving. It was first considered by Philip (1972), who made a significant contribution to this field. Because of its substantial implications for various research endeavors, this field has garnered the attention of numerous scholars from its inception. Indeed, it has been studied in several notable works such as Benson (1984); Ecker & Song (1994). Subsequently, based on the previous works, Abbas & Chaabane (2006) developed the first method for optimizing a linear function over the efficient set without finding all non-dominated points. Following that, Jorge

(2009) proposed an exact algorithm to optimize a linear function over the integer efficient set of a Multi-objective Integer Linear Programming (MOILP) problem. Later, Ouaïl et al. (2017) introduced a branch and bound-based method to optimize a linear function over the efficient set of a MOILP problem. After that, Boland et al. (2017) described a new algorithm to solve the same problem by modifying the algorithm of Jorge (2009). In a different direction, Sierra Altamiranda & Charkhgard (2019) developed the first criterion space search algorithm for optimizing a linear function over the set of efficient solutions of Bi-objective Mixed Integer Linear Programs (BOMILP). Later on, Lokman (2021), based on the work of Jorge (2009) and Boland et al. (2017) developed two algorithms to optimize a linear function over the nondominated set of Multi-objective Integer Programming (MOIP) problem. Most recently, Belkhiri et al. (2022) proposed a new methodology to search for an efficient extreme point that optimizes a linear function over the set of efficient extreme points of a convex polyhedron. Furthermore, this area of research has extended further than linear cases, with many authors looking at non-linear scenarios, such as the work of Zerdani & Moulai (2011), where they developed an algorithm that optimizes an arbitrary linear function over an integer efficient set of a Multi-objective Linear Fractional Programming (MOLFP) problem. Later on, Drici et al. (2018) proposed a new exact method to maximize a linear fractional function over the integer efficient set of a MOILP problem. After that, Moulaï & Drici (2018) presented a new exact method for solving the maximization of an indefinite quadratic utility function over the efficient set of a MOILP problem. Recently, Chaiblaine & Moulaï (2021) described an exact method to optimize a quadratic function over the efficient set of a Multi-objective Integer Linear Fractional Programming (MOILFP) problem. However, it's worth noting that in stochastic cases, this specific problem has attracted less attention compared to its deterministic counterparts. To the best of our knowledge, the only work that addresses this problem is by Chaabane & Mebrek (2014), where their research is primarily centered on optimizing a linear function over the efficient set of Multi-objective Stochastic Integer Linear Programming (MOSILP) problem.

In this research endeavor, our focus is directed toward a new challenge known as bi-objective optimization over an efficient set. Essentially, this challenge can arise when two decision-makers or more collaborate on the same MOILP problem, with each of them possessing their unique preference objectives and visions. This situation presents a complex task of optimizing their individual preference objectives over the efficient solutions set of this problem in order to ultimately identify the best compromise solutions that satisfy the preferences of each decision-maker. In addition, this challenge also occurs when it comes to identifying the intersection between two efficient solution sets arising from two distinct MOILP and Bi-objective Integer Linear Programming (BOILP) problems to determine the common optimal solutions shared between these two sets.

Nowadays, BOILP has become a sophisticated and well-established field in the operations research domain, marked by significant contributions from various authors, including Soylu (2015); Boland et al. (2015); Adelgren & Gupte (2022); Bongo & Sy (2023). Despite the important roles played by both BOILP and optimization over an efficient set in various fields, the intersection of these two areas remains comparatively less explored and developed. Notably, few references delve into this particular area, such as the work of Chaiblaine et al. (2020), where the authors introduced an exact method for optimizing two fractional linear functions over the efficient set of a MOILFP problem. After that, Cherfaoui & Moulaï (2021) presented an exact method for optimizing two preference functions over the efficient set of a MOILP problem. This significant gap between BOILP and optimization over the efficient set offers an interesting direction for further exploration and progress in the research operations domain.

The main objective of our contribution is to address this new challenge, specifically tackling the resolution of the problem mentioned above within a stochastic environment. To the best of our knowledge, no prior study has addressed this specific problem thus far. The only study that tackled a similar problem is the previously mentioned work by Djamal and Fatma (2014), It is worth noting that their work focused on optimizing a single linear function over an efficient set of a MOSILP problem. In contrast, our research takes a different approach by concentrating on the optimization of two preference functions over the efficient set. To be precise, this paper introduces an exact method to optimize two preference functions over an efficient set of a MOSILP problem. The proposed method is a combination of two approaches: the first one is known as the L-shaped method which is an iterative algorithm used to solve the two-stage stochastic linear programming problems, such as Li & Grossmann (2018); Ušpurienė et al. (2018); Dos Santos & Oliveira (2019); Torres et al. (2022). In this context, first-stage decisions precede the realization of uncertainty, while second-stage decisions occur after the realization of uncertain parameters. One of the most significant roles of the L-shaped method is its efficiency in handling large-scale problems with numerous scenarios. Instead of solving the entire problem at once, it decomposes the original stochastic problem into two manageable levels: a master deterministic problem and subproblems, where the master problem is solved in the first level and provides an initial solution. After that, the subproblems are solved iteratively, incorporating information from realized scenarios to improve the solution. For more details see Kall (1976); Van Slyke & Wets (1969); Kall et al. (1994); Birge & Louveaux (2011). The second approach involves adapting the branch-and-bound procedure to suit our needs, this adaptation is enhanced by the implementation of efficient cuts and tests, enabling the removal of a substantial number of inefficient solutions within the search tree.

In summary, the rest of the paper is structured as follows, the first section (cf., Section 1) provides an overview of the problem under investigation and presents its definition within the general context, as well as explains the transformation of the stochastic problem to the equivalent deterministic problem. Section 2 presents some basic results concerning the L-shaped method. In addition, the section offers some definitions and results related to multi-objective integer linear programming. Section 3 explains the methodology used in the resolution process and introduces the technical presentation of the algorithm followed by a discussion of some theoretical results, as well as how to generalize the proposed method in the multi-decision case. In section 4, a didactic example is given to illustrate the different steps of the method, followed by the outline of the search tree. In section 5, a preliminary experimental study is provided to show the performance of the method by testing it on randomly generated instances. Finally, the paper is terminated with a conclusion that summarizes the work presented in this paper.

1 PRELIMINARIES

In this section, we establish some notations and definitions essential for comprehending the content presented in this article. First, we outline the MOSILP problem within the general context of Urli & Nadeau (1990); Adeyefa et al. (2011); Amrouche & Moulaï (2012), as well as the definition of our primary problem which is called ”Bi-objective Integer Stochastic Optimization over the Integer Stochastic Efficient Set”. Following this, we elucidate the methodology for transforming the stochastic problem into an equivalent deterministic counterpart.

1.1 Problem formulation

The multi-objective stochastic integer linear programming (MOSILP) problem with random variable coefficients in objective functions and/or some constraints is formulated as follows:

$$(MOSILP)\begin{cases}\min\ Z_i = C_i(\xi)x, \ \ i\in\{1,2,\dots,K\},\\ s.t.,\\ \quad A\,x = b,\\ \quad T(\xi)\,x = h(\xi),\\ \quad x\in\mathbb{N},\end{cases} \tag{1}$$

where $K \geq 2$ represents the number of objectives, the vector b of dimension $(m\times 1)$ and the real matrix A of dimension $(m\times n)$ are deterministic, whereas the decision vector variables x belong to $\mathbb{R}^n$. Besides, ξ represents the possible realization. $C_i(\xi), T(\xi)$ are random matrices of dimensions $(1\times n)$, $(m_1\times n)$ respectively, and $h(\xi)$ is a random vector of dimension $(m_1\times 1)$, with a known joint probability distribution which is not influenced by the choice of the decision x and defined on a probability space (Ω,Ξ,P).

In this article, our primary objective is to solve the problem involving the optimization of two preference functions over the efficient set of the MOSILP problem (1). This can be formulated as follows:

$$(P)\begin{cases}\min \quad \phi_1 = d_1(\xi)x,\\ \min \quad \phi_2 = d_2(\xi)x,\\ s.t.,\\ \qquad x\in\mathscr{X}_E,\end{cases} \tag{2}$$

where $d_1(\xi)$ and $d_2(\xi)$ are two random vectors in $\mathbb{R}^n$ that represent the preference functions of decision-makers. $\mathscr{X}_E$ denotes the efficient solution set of the MOSILP problem (1).

The main difficulty in solving problem (2) is the fact that it considers several stochastic objective functions simultaneously. Before starting to solve problem (2), we pass through an essential step which is the transformation of a stochastic problem into an equivalent deterministic problem. The most usual approach that has used this strategy is the work of Caballero et al. (2001, 2004). This transformation is necessary because stochastic problems contain uncertain parameters, which cannot be directly optimized. We will discuss this transformation in detail in the following subsection.

1.2 Equivalent deterministic problem

Let us commence by considering a probability space (Ω, Ξ, P) that encompasses random variables controlled by a finite discrete probability distribution $(\xi^r, P^r), r \in \{1, 2, \dots, R\}$. This distribution comprises a limited number of realizations, denoted by R, with each realization corresponding to a distinct scenario.

For each realization ξ^r of ξ, we associate a criterion $Z_i^r = C_i(\xi^r)x$, where $C_i(\xi^r)x$ represents the objective function for the ith objective under the rth scenario. Furthermore, we introduce a matrix $T(\xi^r)$ and a vector $h(\xi^r)$ to incorporate the diverse scenarios that impact the K objectives and stochastic constraints. By employing this approach, we can effectively integrate the uncertainty within the problem and reformulate problem (1) as follows:

$$(MOSILP_R) \begin{cases} \min\ Z_i^r = C_i(\xi^r)x,\ i \in \{1,2,\dots,K\},\ r \in \{1,2,\dots,R\}, \\ s.t., \\ \quad A\, x = b, \\ \quad T(\xi^r)\, x = h(\xi^r),\ r \in \{1,2,\dots,R\}, \\ \quad x \in \mathbb{N}. \end{cases} \tag{3}$$

This paper adopts the concept of recourse from single-criterion stochastic programming, as previously explored by Van Slyke & Wets (1969); Teghem (1983, 1990); Higle & Sen (1991), using a deterministic recourse matrix W. To effectively address constraint violations, penalties denoted as $q^r = q(\xi^r)$ are assigned to each realization Z_i^r, where $r \in \{1, 2, \dots, R\}$. Moreover, to integrate these penalties into the problem formulation, a recourse function $\mathbb{Q}(x, \xi^r)$ is introduced, as defined in (4). This function encapsulates the penalties associated with constraint violations and is incorporated into each criterion Z_i^r.

$$\mathbb{Q}(x, \xi^r) = \min_y \{(q^r)^\intercal y | Wy = h(\xi^r) - T(\xi^r)x, y \geq 0\}. \tag{4}$$

The deterministic equivalent of the problem presented in (3) takes the form of a multi-objective integer linear program (MOILP). This program can be defined as follows:

$$(MOILP_D) \begin{cases} \min\ \ \widetilde{Z}_i + Q(x),\ \ i \in \{1,2,\dots,K\}, \\ s.t., \\ \quad Ax = b, \\ \quad x \in \mathbb{N}, \end{cases} \tag{5}$$

where

$$\begin{cases} \widetilde{Z}_i = \mathbb{E}[Z_i^r] = \sum_{r=1}^{R} P^r Z_i^r = \sum_{r=1}^{R} P^r C_i(\xi^r)x = \widetilde{C}_i x, \ i \in \{1,2,\ldots,K\}, \\ Q(x) = \mathbb{E}[Q(x,\xi)] = \sum_{r=1}^{R} P^r (q^r)^\intercal y^r. \end{cases}$$

Similarly, when considering preference objectives, we express $\phi_i^r = d_i(\xi^r)x$ for $i \in \{1,2\}$, where $d_i(\xi^r)x$ represents the preference objective for the ith criterion in the rth scenario. In this context, the deterministic equivalent of our primary problem is defined as follows:

$$(P_D) \begin{cases} \min \quad \widetilde{\phi}_1 + Q(x), \\ \min \quad \widetilde{\phi}_2 + Q(x), \\ s.t., \\ \quad x \in \mathscr{X}_{ED}, \end{cases} \tag{6}$$

where

$$\begin{cases} \widetilde{\phi}_1 = \mathbb{E}[\phi_1^r] = \mathbb{E}[d_1(\xi)x] = \sum_{r=1}^{R} P^r d_1(\xi^r)x = \widetilde{d}_1 x, \\ \widetilde{\phi}_2 = \mathbb{E}[\phi_2^r] = \mathbb{E}[d_2(\xi)x] = \sum_{r=1}^{R} P^r d_2(\xi^r)x = \widetilde{d}_2 x, \\ Q(x) = \mathbb{E}[Q(x,\xi)] = \sum_{r=1}^{R} P^r (q^r)^\intercal y^r. \end{cases}$$

Here, $\mathscr{X}_{ED}$ signifies the set of efficient solutions for the deterministic equivalent problem (5). Furthermore, the relaxed problem of our main problem can be defined as follows:

$$(P_R) \begin{cases} \min \quad \widetilde{\phi}_1 = \widetilde{d}_1 x + \mathbb{E}[Q(x,\xi)], \\ \min \quad \widetilde{\phi}_2 = \widetilde{d}_2 x + \mathbb{E}[Q(x,\xi)], \\ s.t, \\ \quad x \in \mathscr{S}, \end{cases} \tag{7}$$

where $\mathscr{S} = \{x \in \mathbb{N}^n | Ax = b\}$.

2 THE MAIN RESULTS OF THE COMBINED TECHNIQUE

This section introduces the various methods and strategies incorporated into the proposed algorithm. It begins by providing a brief detailed description of the L-shaped method, as expounded in Kall (1976); Van Slyke & Wets (1969); Kall et al. (1994); Birge & Louveaux (2011). Following this, the section delves into the adaptations adopted into our algorithm by elucidating the efficiency definition within a broader context - Ecker & Kouada (1978), followed by a detailed explanation of the efficient cutting strategy - Ouaïl et al. (2017).

2.1 The principle of the L-shaped method

The L-shaped method stands as an optimization algorithm developed to tackle two-stage stochastic linear programming problems. These problems involve decision-making amid uncertainty, where the decision-maker navigates choices based on both known and uncertain parameters. The fundamental concept behind the "L-shaped method" revolves around a two-stage resolution process. In the initial stage, decisions are formulated relying on known parameters, while the second stage entails decision-making grounded in uncertain parameters. The method iteratively tackles the second-stage problem, referred to as feasibility and optimality tests, while maintaining the first-stage decision as a constant until an optimal solution is found, see Van Slyke & Wets (1969).

It's crucial to discern between two methodologies, both labeled as the "L-shaped method." The more widely recognized version, employed in this paper, pertains to a stochastic programming approach. In contrast, Boland et al. (2016) introduced their "L-shaped method" tailored for solving deterministic multi-objective problems through a search for non-dominated solutions. In exploring the complexities of decision-making in a stochastic environment, we focus on two essential evaluations that are part of the L-method: the feasibility test and the optimality test. In the following subsections, we meticulously discuss the different methodologies used to evaluate the feasibility and optimality of solutions within the L-shaped method framework across the various scenarios.

2.1.1 Feasibility Test

Suppose that the recourse matrix W is fixed, and let x^l be a feasible solution to problem (7). To check for feasibility of the second-stage problems, the L-shaped algorithm endeavors to identify a directional vector σ by solving the following program (8) - Farkas (1902):

$$(FT)\begin{cases}\max \quad \sigma^{\intercal}[(h(\xi^r)-T(\xi^r)x^l)],\\ s.t.,\\ \qquad \sigma^{\intercal}W \leq 0,\\ \qquad \|\sigma\|_1 \leq 1.\end{cases} \tag{8}$$

The last constraint is added to bound σ; otherwise, the maximum value will be set up to $+\infty$ which is not interesting. In addition, if, for some ξ^r, $\sigma_r^{\intercal}[(h(\xi^r)-T(\xi^r)x^l)] > 0,\ r \in \{1,2,\dots,R\}$, where $\sigma_r, r \in \{1,2,\dots,R\}$ is the optimal solution of program (8), then we have found a ξ^r for which x^l does not yield a feasible second-stage problem. In this case, the feasibility cut (9) is added to problem (7).

$$\sigma^{\intercal}[(h(\xi^r)-T(\xi^r)x] \leq 0. \tag{9}$$

It is worth noting that the program (8) is solved iteratively during the algorithm's process, helping to obtain a feasible solution for the second-stage problem.

2.1.2 Optimality Test

Supposing that all feasible cuts are there, we can reformulate problem (7) by introducing a new variable θ

$$(P_R^l)\begin{cases} min\ \ \widetilde{\phi}_i = \widetilde{d}_i x + \theta,\ i \in \{1,2\}, \\ s.t., \\ \quad x\ \in \mathscr{S}_l, \\ \quad \theta \geq Q(x), \end{cases} \tag{10}$$

where $S_l = \{x \in \mathscr{S},\ \sigma_r^\intercal T(\xi^r)x \geq \sigma_r^\intercal h(\xi^r),\ r \in \{1,2,\ldots,R\}\}$ is a non-empty compact polyhedron in $\mathbb{R}^n$ and $\theta \geq Q(x)$ represents the optimality cut. Of course, computationally we can not use $\theta \geq Q(x)$ as a constraint since $Q(x)$ is defined implicitly by a large number of second-stage problems.
To address this, we solve problem (10) by removing this constraint to find a feasible solution (x^l, θ^l). The process begins by initializing $\theta = -\infty$.

To verify the optimality of x^l, we solve first the dual problem of problem (4) as outlined below:

$$(OT)\begin{cases} \max \quad \pi^\intercal[(h(\xi^r) - T(\xi^r)x^l)], \\ s.t., \\ \qquad \pi^\intercal W \leq (q^r)^\intercal. \end{cases} \tag{11}$$

In this situation, the optimal solutions $(\pi_r,\ r \in \{1,2,\ldots,R\})$ of problems (11) are used to calculate the expected recourse function value $Q(x^l)$ given by:

$$Q(x^l) = \sum_{r=1}^{R} P^r Q(x^l, \xi^r) = \sum_{r=1}^{R} P^r (\pi_r)^\intercal [h(\xi^r) - T(\xi^r)x^l].$$

If $\theta^l \geq Q(x^l)$, then x^l is optimal for problem (10), otherwise, the optimality cut (12) is incorporated into problem (10) which is being optimised again.

$$\theta \geq \sum_{r=1}^{R} P^r (\pi_r)^\intercal [h(\xi^r) - T(\xi^r)x]. \tag{12}$$

In the following subsections, we will discuss in detail the strategies adopted and explain how they integrate seamlessly into the proposed algorithm to provide an overview of the proposed method. We begin by describing the famous efficiency test cited in Ecker & Kouada (1978). Then we introduce the cutting technique used to reduce the search domain, known as ”efficient cut”, see Ouaïl et al. (2017).

2.2 Efficiency Test

In the realm of multi-objective programming (MOP), the existence of conflicting objectives leads to the generation of multiple efficient solutions. Additionally, efficiency can be defined in this context as follows:

Definition 1. *A point $\bar{x} \in \mathscr{S}$ is an efficient solution for problem (5) if, and only if, there is no $x \in \mathscr{S}$, such that, $\widetilde{Z}_i(x) \leq \widetilde{Z}_i(\bar{x})$, for all $i \in \{1,2,\ldots,K\}$ and $\widetilde{Z}_i(x) < \widetilde{Z}_i(\bar{x})$, for at least one $i \in \{1,2,\ldots,K\}$. Otherwise, x- is not efficient and the corresponding vector $(\widetilde{Z}_1(\bar{x}),\widetilde{Z}_2(\bar{x}),\ldots,\widetilde{Z}_p(\bar{x}))$ is said to be dominated.*

The following theorem (1) provides another characterization of an efficient solution, which is used as a test procedure in our study. Moreover, let x^l be the integer solution obtained following the branching process reinforced by the feasibility and optimality test at each step.

Theorem 1. *$x^l \in \mathscr{X}_{ED}$ if, and only if, the optimal value of the objective function $\Psi(\psi,x)$ is null in the following integer linear programming problem:*

$$(EK(x^l))\begin{cases} max\ \Psi = \sum_{i=1}^{K} \psi_i, \\ \\ s.t., \quad \begin{cases} \widetilde{C}_i x + \psi_i = \widetilde{C}_i x^l,\ i \in \{1,2,\ldots,K\}, \\ x \in \mathscr{S}, \\ \psi_i \geq 0, \quad i \in \{1,2,\ldots,K\}. \end{cases} \end{cases} \tag{13}$$

The efficiency of the solution x^l is verified through the resolution of problem (13). In particular, x^l is efficient if and only if the optimal value of the objective function Ψ equals zero in problem (13). In cases where the optimal value is not zero, the optimal solution of problem (13) is considered an efficient solution to problem (5). It is noteworthy that Theorem 1 is specifically applicable to the linear case and was originally proposed by Ecker & Kouada (1978). For the general case, we mention the scalarization introduced by Benson (1984).

2.3 Efficient Cut

In each iteration l of the optimization process, the efficient cut is utilized to eliminate all inefficient solutions from the feasible region and remove areas that do not improve the value of the utility function.

Assuming that x^l is the l^{th} generated optimal solution of problem (10), the following definitions and notations are used:

- $\mathscr{B}_l$ is the indexes set of the basic variables of x^l.
- $\mathscr{N}_l$ is the indexes set of the non-basic variables of x^l.

The decrease direction of each criterion $\widetilde{Z}_i, i \in \{1,2,\ldots,K\}$ of problem (5) is determined by using their reduced gradient vectors. On the other hand, the method uses this information to build an efficient cut to remove integer solutions that are not efficient for problem (5) and determine an efficient new integer solution. To do so, we define a new set of all decreasing directions of the criteria as follows:

- $\mathscr{H}_l = \left\{ j \in \mathscr{N}_l | \exists i \in \{1,2,\ldots,K\},\ \text{with}\ \bar{C}_i^j < 0 \right\} \bigcup \left\{ j \in \mathscr{N}_l | \bar{C}_i^j = 0,\ \forall i \in \{1,2,\ldots,K\} \right\}.$

Here, $\bar{C}_i^j$ is the j^{th} component of reduced gradient vectors of the objective function $\widetilde{Z}_i$.

Once this set has been identified, we proceed to formulate the efficient cut (14) designed to eliminate all inefficient solutions, see Ouaïl et al. (2017). This cut plays an essential role in the optimization process, enabling us to reduce the search space, improve the overall efficiency of the optimization process, and ultimately, find optimal solutions.

$$\sum_{j \in \mathscr{H}_l} x_j \geq 1. \tag{14}$$

3 METHODOLOGY, ALGORITHM AND THEORETICAL RESULTS

In the following subsection, we detail each step of our methodology that is designed to solve the problem described in (6). Following this, we describe the algorithm, giving a comprehensive overview of the essential steps in our proposed method. After that, we turn to a discussion of various theoretical results. Finally, we demonstrate how to extend the method to general multi-decision cases.

3.1 Methodology description

The proposed algorithm aims to generate a subset $\mathscr{X}_B \subset \mathscr{X}_E$, where $\mathscr{X}_E$ is the efficient set of the deterministic equivalent problem (5). The elements in $\mathscr{X}_B$ are selected to be also efficient in terms of both objective functions d_1 and d_2. Contrary to the classical approach, our algorithm avoids the exhaustive enumeration of all elements in $\mathscr{X}_E$ to generate $\mathscr{X}_B$. The classical method for solving problem (6) is to determine two sets $\mathscr{X}_E$ and $\mathscr{X}_C$, where $\mathscr{X}_C$ represents the efficient solutions in terms of d_1 and d_2. Then we have to select the elements that are in both sets $\mathscr{X}_E$ and $\mathscr{X}_C$ simultaneously, which is very computationally expensive. As mentioned previously, the proposed algorithm employs two distinct techniques to address the problem. These include the well-known L-shaped method and an adaptation of the branch-and-bound procedure, reinforced by efficient cuts and tests.

To start the process, we initialize the two variables, θ and l, to $-\infty$ and 0, respectively. Furthermore, at each node l, the algorithm resolves the linear program (15) using the simplex or dual simplex method.

$$(P^l) \begin{cases} \min \quad \widetilde{\phi}_1 = \widetilde{d}_1 x, \\ s.t., \\ \quad x \in \mathscr{S}_l = \{x \in \mathbb{N}^n | Ax = b\}. \end{cases} \tag{15}$$

Once the linear program (15) has been solved, the algorithm checks for the existence of a solution. In case the program has no solution, the node is fathomed. Otherwise, there are two possible situations:

1. The optimal solution x^l is not an integer. in this situation, the algorithm follows a basic branching process according to the following steps: Identify a component x_j of x^l such

that $x_j = \alpha_j$, where α_j represents a fractional value. After that, the node l of the tree is separated into two nodes which are imposed by the following two additional constraints:

$$\begin{cases} x_j \leq \lfloor \alpha_j \rfloor, \\ x_j \geq \lfloor \alpha_j \rfloor + 1, \end{cases} \tag{16}$$

where, $\lfloor \alpha_j \rfloor$ indicates the greatest integer less than α_j. The linear program obtained must be solved until an integer feasible solution is found (if it exists).

2. The solution x^l is an integer. Here, the process passes through two tests: the feasibility and optimality tests, which are detailed in subsections (2.1.1) and (2.1.2), respectively.

As soon as an optimal integer solution x^l is found after the feasibility and optimality tests, then two efficiency tests are performed by solving the programs 17 and 18 (see Subsection 2.2).

1. In the first step, we test the efficiency of the solution x^l in terms of $\widetilde{d_1}$ and $\widetilde{d_2}$ by solving the following first program:

$$(EK^1(x^l)) \begin{cases} \max \sum_{i=1}^{2} v_i, \\ s.t., \\ \widetilde{d_i}x - v_i = \widetilde{d_i}x^l,\ i \in \{1,2\}, \\ x \in \mathscr{S}_l, \\ v_i \geq 0,\ i \in \{1,2\}. \end{cases} \tag{17}$$

As a well-known result, x^l is efficient if and only if $EK^1(x^l)$ has a maximum value of zero. See, Ecker & Kouada (1978).

2. The process continues by passing the second efficiency test for the deterministic equivalent problem (5).

$$(EK^2(x^l)) \begin{cases} \max \sum_{i=1}^{K} w_i, \\ s.t., \\ \widetilde{C_i}x - w_i = \widetilde{C_i}x^l,\ i \in \{1,2,\ldots,K\}, \\ x \in \mathscr{S}_l, \\ w_i \geq 0,\ i \in \{1,2,\ldots,K\}. \end{cases} \tag{18}$$

Based on the same previous concept, the solution x^l is efficient for the deterministic equivalent problem (5) if and only if $EK^2(x^l)$ has a maximum value of zero.

As a result, if both $EK^1(x^l)$ and $EK^2(x^l)$ have zero maximum value, then x^l belongs to $\mathscr{X}_B$. The process continues to search for other efficient solutions in other nodes (if they exist) by applying efficient cuts to the related program (see, subsection 2.3). Initially, two sets $\mathscr{H}_l$ and $\mathscr{H}_l'$ are constructed. After that, the efficient cuts (19) and (20) are constructed and incorporated into the successor nodes of l.

$$\mathscr{H}_l = \left\{ j \in \mathscr{N}_l | \exists i \in \{1,2,\ldots,K\},\ \bar{C}_i^j < 0 \right\} \cup \left\{ j \in \mathscr{N}_l |\ \bar{C}_i^j = 0,\ \forall i \in \{1,2,\ldots,K\} \right\},$$

$$\sum_{j \in \mathscr{H}_l} x_j \geq 1, \tag{19}$$

And

$$\mathscr{H}_l^{'} = \left\{ j \in \mathscr{N}_l | \, \bar{d}_2^j \, < \, 0 \right\} \cup \left\{ j \in \mathscr{N}_l | \, \bar{d}_1^j = 0 \text{ and } \bar{d}_2^j = 0 \right\},$$

$$\sum_{j \in \mathscr{H}_l^{'}} x_j \geq 1. \tag{20}$$

Hence, the domain of the successor node $l+1$ is determined by applying the efficient cuts (19) and (20) to $\mathscr{S}_l$.

$$\mathscr{S}_{l+1} = \mathscr{S}_{l+1}^1 \cap \mathscr{S}_{l+1}^2, \tag{21}$$

where

$$\begin{cases} \mathscr{S}_{l+1}^1 = \left\{ x \in \mathscr{S}_l | \, \sum_{j \in \mathscr{H}_l} x_j \geq 1 \right\}, \\ \\ \mathscr{S}_{l+1}^2 = \left\{ x \in \mathscr{S}_l | \, \sum_{j \in \mathscr{H}_l^{'}} x_j \geq 1 \right\}. \end{cases}$$

The method ends when all the created nodes are fathomed or when one of the sets ($\mathscr{H}_l$ or $\mathscr{H}_l^{'}$) is empty, which means that $\mathscr{H}_l = \emptyset$ or $\mathscr{H}_l^{'} = \emptyset$.

3.2 Algorithm

Algorithm 1 resumes in detail the main steps of the proposed method, these steps are treated according to the backtracking principle.

3.3 Theoretical Results

The following theoretical tools show that the algorithm yields the set of solutions for problem (6) in a finite number of iterations.

Theorem 2. *Assume that $\mathscr{H}_l \neq \emptyset$ and $\mathscr{H}_l^{'} \neq \emptyset$ at the current integer solution x^l. If $x \neq x^l$ is an efficient solution of problem* (6) *in domain $\mathscr{S}_l$, then, $x \in \mathscr{S}_{l+1}$ ($l+1$ is the successor of l).*

Proof. Let $x \neq x^l$ be an integer solution in domain $\mathscr{S}_l$, such that, $x \notin \mathscr{S}_{l+1}$. Two situations can occur in this case:

1. $x \notin \mathscr{S}_{l+1}^1$, implies that $x \in \left\{ x \in \mathscr{S}_l | \, \sum_{j \in \mathscr{N}_l \setminus \mathscr{H}_l} x_j \geq 1 \right\}$. The components of vector x here satisfy the following inequalities:

$$\begin{cases} \sum\limits_{j \in \mathscr{H}_l} x_j < 1, \\ \sum\limits_{j \in \mathscr{N}_l \setminus \mathscr{H}_l} x_j \geq 1. \end{cases}$$

Which means that $x_j = 0$, for all $j \in \mathscr{H}_l$ and $x_j \geq 1$, for at least one index $j \in \mathscr{N}_l \setminus \mathscr{H}_l$.

Algorithm 1 A Branch and Cut Strategies based Method

```
Data: $\mathscr{X}_B = \emptyset$, $\theta = -\infty$, $l = 0$.
Result: $\mathscr{X}_B$: The solutions set of problem (6).
while As long as a unfathomed node l exists in the tree do
    Solve problem (15) using simplex or dual simplex method;
    if Problem (15) has a feasible solution $x^l$ then
        if $x^l$ is integer then
            Feasibility test.
              Solve program (8);
              if $\sigma^\top[h(\xi^r) - T(\xi^r)x^l] > 0$ then
                  Add the feasibility cut (9) to the successors of $l$;
            else
                Optimality test.
                  Solve program (11), calculate Q(x);
                  if $Q(x^l) > \theta^l$ then
                      Add the optimality cut (12) to the successors of $l$;
                else
                    Efficiency test.
                      Solve program (17);
                      $v$ is the optimal solution criteria;
                      if $v = 0$ then
                          Solve program (18);
                            $w$ is the optimal solution criteria;
                            if $w = 0$ then
                                $\mathscr{X}_B = \mathscr{X}_B \cup \{x^l\}$;
                          end
                    end
                    Update set $\mathscr{S}$.
                      Construct the sets $\mathscr{H}_l$ and $\mathscr{H}_l^{'}$;
                      if $\mathscr{H}_l$ or $\mathscr{H}_l^{'} = \emptyset$ then
                          Fathom the node $l$;
                    else
                        Add the efficient cuts (19) and (20) to the successors of $l$;
                    end
                end
            end
        else
            Branching process.
              Choose an index j such that $\alpha_j$ is the most fractional number, then:
              Split the program $P^l$ into two sub programs, by adding respectively
              the constraints $x_j \leq \lfloor \alpha_j \rfloor$ and $x_j \geq \lfloor \alpha_j \rfloor + 1$, to obtain $(P^{l_1})$ and $(P^{l_2})$
              $(l_1 \neq l_2$ and $l_1 \geq l+1,\ l_2 \geq l+1)$;
        end
    else
        The corresponding node $l$ is fathomed;
    end
end
```

On the other hand, for all criterion $i \in \{1,2,\dots,K\}$, the following equality is supported using the simplex table:

$$\widetilde{Z}_i(x) = \widetilde{C}_i x = \sum_{j\in\mathscr{B}_l} \overline{C}_i^j x_j + \sum_{j\in\mathscr{N}_l} \overline{C}_i^j x_j, \text{ where } \sum_{j\in\mathscr{B}_l} \overline{C}_i^j x_j = \widetilde{Z}_i(x^l).$$

Hence, we can write as follows:

$$\begin{aligned}\widetilde{Z}_i(x) - \widetilde{Z}_i(x^l) &= \sum_{j\in\mathscr{N}_l} \overline{C}_i^j x_j,\\ &= \sum_{j\in\mathscr{H}_l} \overline{C}_i^j x_j + \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l} \overline{C}_i^j x_j,\\ &= \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l} \overline{C}_i^j x_j.\end{aligned}$$

Thus, this implies that $\widetilde{Z}_i(x) \le \widetilde{Z}_i(x^l)$, for all criterion $i \in \{1,2,\dots,K\}$. As well as $\widetilde{Z}_i(x) < \widetilde{Z}_i(x^l)$, for at least one criterion of $\overline{C}_i^j \le 0$, for all $j \in \mathscr{N}_l \setminus \mathscr{H}_l$.

We conclude then that the solution x is not efficient for the deterministic equivalent problem 5. In other words, this means that $x \notin \mathscr{X}_E$, and $x \notin \mathscr{X}_B$.

2. $x \notin \mathscr{S}_{l+1}^2$ implies that $x \in \left\{x \in \mathscr{S}_l | \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l'} x_j \ge 1\right\}$. The components of vector x in this situation satisfy the following inequalities:

$$\begin{cases} \sum_{j\in\mathscr{H}_l'} x_j < 1,\\ \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l'} x_j \ge 1.\end{cases}$$

Which means that $x_j = 0$, for all $j \in \mathscr{H}_l'$ and $x_j \ge 1$, for at least one index $j \in \mathscr{N}_l \setminus \mathscr{H}_l'$. Furthermore, by using the simplex table in x^l, the following equality is satisfied:

$$\widetilde{d}_2 x = \sum_{j\in\mathscr{B}_l} \overline{d}_2^j x_j + \sum_{j\in\mathscr{N}_l} \overline{d}_2^j x_j \text{ where } \sum_{j\in\mathscr{B}_l} \overline{d}_2^j x_j = \widetilde{d}_2 x^l,$$

Here, we can write as follows:

$$\begin{aligned}\widetilde{d}_2 x - \widetilde{d}_2 x^l &= \sum_{j\in\mathscr{H}_l'} \overline{d}_2^j x_j + \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l'} \overline{d}_2^j x_j,\\ &= \sum_{j\in\mathscr{N}_l\setminus\mathscr{H}_l'} \overline{d}_2^j x_j.\end{aligned}$$

Thus, $\widetilde{d}_2 x \le \widetilde{d}_2 x^l$, with $\widetilde{d}_2 x < \widetilde{d}_2 x^l$ and $\widetilde{d}_1 x \le \widetilde{d}_1 x^l$, since x^l is the optimum of the current problem. This means that $x \notin \mathscr{X}_B$.

Since $x \notin \mathscr{X}_B$ in both situations. As a result, x is not an efficient solution of problem (6). □

Proposition 1. *Suppose that $\mathscr{H}_l = \emptyset$ or $\mathscr{H}'_l = \emptyset$ at the current integer solution x^l. Then there is no solution in the remaining domain that is not dominated by x^l.*

Proof.

- Suppose that $\mathscr{H}_l = \emptyset$, which means that we have $\bar{C}_i^j \leq 0$, $\forall j \in \mathscr{N}_l$, $\forall i \in \{1,2,\ldots,K\}$. As well as, $\forall j \in \mathscr{N}_l, \exists i_0 \in \{1,2,\ldots,K\}$, such that $\bar{C}_{i_0}^j < 0$. This implies that x^l dominates all points x, $x \neq x^l$ of domain $\mathscr{S}_l$.
- Now assume that $\mathscr{H}_l^{'} = \emptyset$, and this means that $\forall j \in \mathscr{N}_l$, $\bar{d}_2^j < 0$ or $\bar{d}_2^j = 0$, and $\bar{d}_1^j < 0$, leading to $\bar{d}_1^j < 0$, $\forall j \in \mathscr{N}_l$. Since it is an optimal solution for (P^l). Thus, x^l becomes the most preferred solution in the domain $\mathscr{S}_l$. □

Theorem 3. *Algorithm 1 terminates in a finite number of iterations and the set $\mathscr{X}_B$ contains all the solutions of problem* (6).

Proof. Let $\mathscr{S}$ be the set of feasible integer solutions of the deterministic equivalent problem 5 a bounded set. Moreover, the efficient sets $\mathscr{X}_B$ and $\mathscr{X}_E$ contain a finite number of integer solutions which means that the cardinality of these sets is a finite number. For each step l of the algorithm 1, when an integer solution is found, there are only a finite number of cuts that eliminate x^l and all dominated solutions from the search tree (see Proposition 1). Similarly, the algorithm used a limited number of feasibility and optimality cuts. which means that the search tree would have a finite number of branches and that the algorithm terminates in a finite number of steps.

For each step l of Algorithm 1, when an integer solution x^l is found, the cuts eliminate x^l and all dominated solutions from the search tree (see Proposition 1). In addition, for $\mathscr{X}_B$ to contain all the solutions of problem (6), the fathoming rules are used without loss of any elements in $\mathscr{X}_B$. The first rule is when the set $\mathscr{H}_l$ or $\mathscr{H}_l^{'}$ is empty. In this case, the current node can be fathomed since the rest of the domain contains only dominated solutions either in terms of the deterministic equivalent problem or in terms of the two preference functions. The second rule is the trivial case when the reduced domain becomes infeasible, whether it is because of previous cuts or the branching. □

3.4 Generalization of the method

The same algorithm can be used to solve the problem where there are p decision-makers, the general problem can be formulated as follows:

$$(P)\begin{cases} \min & \phi_1 = d_1(\xi)x, \\ \min & \phi_2 = d_2(\xi)x, \\ \vdots & \\ \min & \phi_p = d_p(\xi)x, \\ s.t., & \\ & x \in \mathscr{X}_E. \end{cases} \tag{22}$$

To solve this problem, we use our algorithm and modify the efficiency test $EK(x^l)$ as follows:

$$(EK(x^l))\begin{cases} \max \sum_{i=1}^{p} v_i, \\ s.t., \\ \widetilde{d}_i x - v_i = \widetilde{d}_i x^l,\ i \in \{1,2,\dots,p\}, \\ x \in \mathscr{S}_l, \\ v_i \geq 0,\ i \in \{1,2,\dots,p\}. \end{cases} \tag{23}$$

Also, $\mathscr{H}_l^{'}$ is calculated as follows:

$$\mathscr{H}_l^{'} = \left\{ j \in \mathscr{N}_l | \exists i \in \{1,2,\dots,p\}, \bar{d}_i^j < 0 \right\} \cup \left\{ j \in \mathscr{N}_l |\ \bar{d}_i^j = 0,\ \forall i \in \{1,2,\dots,p\} \right\}.$$

The rest of the algorithm remains the same. The algorithm returns the set of efficient solutions $\mathscr{X}_B = \mathscr{X}_E \cap \mathscr{X}_C$.

4 DIDACTIC EXAMPLE

Consider the following multi-objective integer linear programming stochastic problem:

Scenario 1

$$MOSILP(\xi^1)\begin{cases} \min Z_1^1 = -9x_1 + 4x_2, \\ \min Z_2^1 = 3x_1 - 5x_2, \\ \min Z_3^1 = 5x_1 - 11x_2, \\ \text{s.t.}, \\ x_1 - 4x_2 \leq 3, \\ x_1 + 3x_2 \leq 18, \\ 2x_1 + x_2 \leq 10, \\ x_1 + 2x_2 \leq 3, \\ -3x_1 + x_2 \leq 5, \\ x_1,\ x_2 \in \mathbb{N}. \end{cases}$$

Scenario 2

$$MOSILP(\xi^2)\begin{cases} \min Z_1^2 = 3x_1 - 2x_2, \\ \min Z_2^2 = 6x_1 + x_2, \\ \min Z_3^2 = -7x_1 + 10x_2, \\ \text{s.t.}, \\ x_1 - 4x_2 \leq 3, \\ x_1 + 3x_2 \leq 18, \\ 2x_1 + x_2 \leq 10, \\ 5x_1 + x_2 \leq 4, \\ -3x_1 + 2x_2 \leq 1, \\ x_1,\ x_2 \in \mathbb{N}. \end{cases}$$

The recourse matrix and the deterministic constraints matrices are given for both scenarios by:

$$A = \begin{pmatrix} 1 & -4 \\ 1 & 3 \\ 2 & 1 \end{pmatrix},\ b = \begin{pmatrix} 3 \\ 18 \\ 10 \end{pmatrix}, W = \begin{pmatrix} -2 & -1 & 2 & 1 \\ 3 & 2 & -5 & -6 \end{pmatrix}.$$

The penalties of constraint violations and the probability distribution are given for both scenarios by:

$$q\left(\xi^1\right) = (1,3,6,2)^\intercal, \quad \mathbb{P}\left(\xi^1\right) = \frac{2}{3}, \quad q\left(\xi^2\right) = (5,3,2,1)^\intercal, \quad \mathbb{P}\left(\xi^2\right) = \frac{1}{3}.$$

The stochastic constraints matrices are given for both scenarios by:

$$T\left(\xi^1\right) = \begin{pmatrix} 1 & 2 \\ -3 & 1 \end{pmatrix}, T\left(\xi^2\right) = \begin{pmatrix} 5 & 1 \\ -3 & 2 \end{pmatrix}, h\left(\xi^1\right) = \begin{pmatrix} 3 \\ 5 \end{pmatrix}, h\left(\xi^2\right) = \begin{pmatrix} 4 \\ 1 \end{pmatrix}.$$

The deterministic multiple objective integer linear programming problem:

$$\widetilde{Z}_1 = \mathbb{E}[C_1(\xi)x] = \mathbb{P}(\xi^1)C_1(\xi^1)x + \mathbb{P}(\xi^2)C_1(\xi^2)x,$$
$$\widetilde{Z}_2 = \mathbb{E}[C_2(\xi)x] = \mathbb{P}(\xi^1)C_2(\xi^1)x + \mathbb{P}(\xi^2)C_2(\xi^2)x,$$
$$\widetilde{Z}_3 = \mathbb{E}[C_3(\xi)x] = \mathbb{P}(\xi^1)C_3(\xi^1)x + \mathbb{P}(\xi^2)C_3(\xi^2)x.$$

$$(MOSILP_D)\begin{cases} \min\ \widetilde{Z}_1 = -5x_1 + 2x_2, \\ \min\ \widetilde{Z}_2 = 4x_1 - 3x_2, \\ \min\ \widetilde{Z}_3 = x_1 - 4x_2, \\ \text{s.t.,} \\ x_1 - 4x_2 \le 3, \\ x_1 + 3x_2 \le 18, \\ 2x_1 + x_2 \le 10, \\ x_1, x_2 \in \mathbb{N}. \end{cases}$$

Consider now two decision makers with the following preference functions for each scenario.

Scenario 1

$$P(\xi^1)\begin{cases} \min \phi_1^1 = -2x_1 + 4x_2, \\ \min \phi_2^1 = 2x_1 - 1x_2, \\ \text{s.t.,} \\ x_1,\ x_2 \in \mathscr{X}_{ED}. \end{cases}$$

Scenario 2

$$P(\xi^2)\begin{cases} \min \phi_1^2 = x_1 + 4x_2, \\ \min \phi_2^2 = 2x_1 + 5x_2, \\ \text{s.t.,} \\ x_1,\ x_2 \in \mathscr{X}_{ED}. \end{cases}$$

The main deterministic relaxed problem is defined by:

$$\widetilde{\phi}_1 = \mathbb{E}[d_1(\xi)x] = \mathbb{P}\left(\xi^1\right)d_1\left(\xi^1\right)x + \mathbb{P}\left(\xi^2\right)d_1\left(\xi^2\right)x,$$
$$\widetilde{\phi}_2 = \mathbb{E}[d_2(\xi)x] = \mathbb{P}\left(\xi^1\right)d_2\left(\xi^1\right)x + \mathbb{P}\left(\xi^2\right)d_2\left(\xi^2\right)x.$$

$$(P_{DR})\begin{cases} \min\ \widetilde{\phi}_1 = -x_1 + 4x_2, \\ \min\ \widetilde{\phi}_2 = 2x_1 + x_2, \\ \text{s.t.,} \\ x_1, x_2 \in \mathscr{S}, \end{cases}$$

where $\mathscr{S} = \{x_1 - 4x_2 \le 3,\ x_1 + 3x_2 \le 18,\ 2x_1 + x_2 \le 10 \mid x_1, x_2 \in \mathbb{R}^n\}$.

Initialization: $\mathscr{X}_B = \emptyset,\ \theta = -\infty,\ \mathscr{S}_0 = \mathscr{S}$.

Step 1: We start by solving the first following relaxed problem (P^0).

$$(P^0)\begin{cases} \min\ \phi = -x_1 + 4x_2, \\ \text{s.t.,} \\ x \in \mathscr{S}_0. \end{cases}$$

The first optimal solution is $x^0 = (\frac{43}{9}, \frac{4}{9})$ which is not an integer. The algorithm starts the branching process.

Step 2: The branching process. The search nodes obtained after branching on x^0 are:

Node 1 $\mathscr{S}_1 = \{\, x \in \mathscr{S}_0\ :\ x_1 \le 4\}$.

Node 2 $\mathscr{S}_2 = \{x \in \mathscr{S}_0 \ : \ x_1 \geq 5\}$. No solution exists in this node. Thus, this node is **fathomed**.

Now, we continue the process by solving the new problem (P^1) at node 1.

$$(P^1)\begin{cases} \min\ \phi = -x_1 + 4x_2, \\ \text{s.t.,} \\ x \in \mathscr{S}_1. \end{cases}$$

The optimal solution is $x^1 = (4, \frac{1}{4})$. The algorithm will again perform branching process.

Step 2: The branching process. The obtained search nodes after branching on x^1 are:

Node 3 $\mathscr{S}_3 = \{x \in \mathscr{S}_2 \ : \ x_2 \geq 1\}$.

Node 4 $\mathscr{S}_4 = \{\, x \in \mathscr{S}_2 \ : \ x_2 \leq 0\}$.

In this situation, the process continues with solving problem (P^3) at node 3.

$$(P^3)\begin{cases} \min\ \phi = -x_1 + 4x_2, \\ \text{s.t.,} \\ x \in \mathscr{S}_3. \end{cases}$$

The obtained optimal integer solution is $x^2 = (4, 1)$. In this situation, x^2 is tested for feasibility and optimality for each scenario.

Step 3: Feasibility and Optimality tests. At first, we evaluate the feasibility of solution x^2 by solving the following two problems:

$$h(\xi^1) - T(\xi^1)x^2 = \begin{pmatrix} 3 \\ 5 \end{pmatrix} - \begin{pmatrix} 1 & 2 \\ -3 & 1 \end{pmatrix}\begin{pmatrix} 4 \\ 1 \end{pmatrix} = \begin{pmatrix} -3 \\ 16 \end{pmatrix}.$$
$$h(\xi^2) - T(\xi^2)x^2 = \begin{pmatrix} 4 \\ 1 \end{pmatrix} - \begin{pmatrix} 5 & 1 \\ -3 & 2 \end{pmatrix}\begin{pmatrix} 4 \\ 1 \end{pmatrix} = \begin{pmatrix} -6 \\ 11 \end{pmatrix}.$$

Scenario 1

$$\sigma(\xi^1)\begin{cases} \max\ -3\sigma_1^1 + 16\sigma_1^2, \\ -2\sigma_1^1 + 3\sigma_1^2 \leq 0, \\ -\sigma_1^1 + 2\sigma_1^2 \leq 0, \\ 2\sigma_1^1 - 5\sigma_1^2 \leq 0, \\ \sigma_1^1 - 6\sigma_1^2 \leq 0, \\ \sigma_1^1 + \sigma_1^2 \leq 1. \end{cases} \quad \sigma_1^{\mathsf{T}} = \left(\tfrac{2}{3}, \tfrac{1}{3}\right),$$

Scenario 2

$$\sigma(\xi^2)\begin{cases} \max\ -6\sigma_2^1 + 11\sigma_2^2, \\ -2\sigma_2^1 + 3\sigma_2^2 \leq 0, \\ -\sigma_2^1 + 2\sigma_2^2 \leq 0, \\ 2\sigma_2^1 - 5\sigma_2^2 \leq 0, \\ \sigma_2^1 - 6\sigma_2^2 \leq 0, \\ \sigma_2^1 + \sigma_2^2 \leq 1. \end{cases} \quad \sigma_2^{\mathsf{T}} = (0, 0).$$

Note that $\sigma_1^{\mathsf{T}}\left[h\left(\xi^1\right) - T\left(\xi^1\right)x^2\right] = \left(\frac{2}{3}, \frac{1}{3}\right)\binom{-3}{16} = \frac{10}{3} > 0$, which means that x^2 is not feasible for the first scenario ξ^1, In this case, we add the following feasibility cut to current problem (P^3).

$$\left(\tfrac{2}{3}, \tfrac{1}{3}\right)\begin{pmatrix} 1 & 2 \\ -3 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} > \left(\tfrac{2}{3}, \tfrac{1}{3}\right)\begin{pmatrix} 3 \\ 5 \end{pmatrix} = \frac{11}{3} \iff -\frac{1}{3}x_1 + \frac{5}{3}x_2 \geq \frac{11}{3}.$$

After adding the feasibility cut, we solve the following problem (P^5):

$$(P^5)\begin{cases} \min\ \phi = -x_1 + 4x_2, \\ \text{s.t.,} \\ x \in \mathscr{S}_5 = \{x \in \mathscr{S}_3 |\ -\frac{1}{3}x_1 + \frac{5}{3}x_2 \geq \frac{11}{3}\}. \end{cases}$$

The optimal solution is $x^3 = (\frac{39}{11}, \frac{32}{11})$. The branching process is then started again.

Step 2: The branching process. After branching on x^3, we obtaine the following search nodes:

Node 6 $\mathscr{S}_6 = \{x \in \mathscr{S}_5 \ : \ x_1 \geq 4\}$. No solution exists in this node. This node is **fathomed**.

Node 7 $\mathscr{S}_7 = \{ \ x \in \mathscr{S}_5 \ : \ x_1 \leq 3\}$.

The process continues in this step by solving problem (P^7) at node 7.

$$(P^7) \begin{cases} \min \ \phi = -x_1 + 4x_2, \\ \text{s.t.}, \\ x \in \mathscr{S}_7. \end{cases}$$

The obtained solution is $x^4 = (3, \frac{14}{5})$. The branching process begins again.

Step 2: The branching process. The search nodes obtained after branching on x^4 are:

Node 8 $\mathscr{S}_8 = \{x \in \mathscr{S}_7 \ : \ x_2 \geq 3\}$.

Node 9 $\mathscr{S}_9 = \{ \ x \in \mathscr{S}_7 \ : \ x_2 \leq 2\}$. No solution exists in this node. Then, this node is **fathomed**.

We solve now problem (P^8) at node 8.

$$(P^8) \begin{cases} \min \ \phi = -x_1 + 4x_2, \\ \text{s.t.}, \\ x \in \mathscr{S}_8. \end{cases}$$

The obtained integer solution is $x^5 = (3,3)$. We continue the process by testing the feasibility and optimality of the solution x^5.

Step 3: Feasibility and Optimality tests. As previously mentioned, this step commences with a feasibility test, where the feasibility of the solution x^5 is examined by solving the following pair of problems.

Scenario 1

$$\sigma(\xi^1) \begin{cases} \max \ -6\sigma_1^1 + 11\sigma_1^2, \\ -2\sigma_1^1 + 3\sigma_1^2 \leq 0, \\ -\sigma_1^1 + 2\sigma_1^2 \leq 0, \\ 2\sigma_1^1 - 5\sigma_1^2 \leq 0, \\ \sigma_1^1 - 6\sigma_1^2 \leq 0, \\ \sigma_1^1 + \sigma_1^2 \leq 1. \end{cases} \quad \sigma_1^{\mathsf{T}} = (0,0),$$

Scenario 2

$$\sigma(\xi^2) \begin{cases} \max \ -14\sigma_2^1 + 4\sigma_2^2, \\ -2\sigma_2^1 + 3\sigma_2^2 \leq 0, \\ -\sigma_2^1 + 2\sigma_2^2 \leq 0, \\ 2\sigma_2^1 - 5\sigma_2^2 \leq 0, \\ \sigma_2^1 - 6\sigma_2^2 \leq 0, \\ \sigma_2^1 + \sigma_2^2 \leq 1. \end{cases} \quad \sigma_2^{\mathsf{T}} = (0,0).$$

Here, $\sigma_1^{\mathsf{T}} = \sigma_2^{\mathsf{T}} = 0$, which means that, x^5 is feasible for both scenarios. Following this, we perform the optimality test for x^5 by solving the following two problems.

Scenario 1 Scenario 2

$$\pi(\xi^1)\begin{cases} \max\ -6\pi_1^1+11\pi_1^2, \\ -2\pi_1^1+3\pi_1^2 \le 1, \\ -\pi_1^1+2\pi_1^2 \le 3, \\ 2\pi_1^1-5\pi_1^2 \le 6, \\ \pi_1^1-6\pi_1^2 \le 2. \end{cases} \quad \pi_1^{\intercal}=(7,5), \qquad \pi(\xi^2)\begin{cases} \max\ -14\pi_1^1+4\pi_1^2, \\ -2\pi_1^1+3\pi_1^2 \le 5, \\ -\pi_1^1+2\pi_1^2 \le 3, \\ 2\pi_1^1-5\pi_1^2 \le 2, \\ \pi_1^1-6\pi_1^2 \le 1. \end{cases} \quad \pi_2^{\intercal}=\left(-\tfrac{11}{3},-\tfrac{7}{9}\right).$$

Once the previous two problems have been solved, the process is followed by first calculating the value of the expected recourse function value $Q(x)$ by:

$$Q\left(x,\xi^1\right)=\pi_1^{\intercal}\left[h\left(\xi^1\right)-T\left(\xi^1\right)x^3\right]=13,$$
$$Q\left(x,\xi^2\right)=\pi_2^{\intercal}\left[h\left(\xi^2\right)-T\left(\xi^2\right)x^3\right]=\frac{434}{9},$$
$$Q(x)=\frac{2}{3}Q\left(x,\xi^1\right)+\frac{1}{3}Q\left(x,\xi^2\right)=\frac{668}{27}.$$

Note that: $Q(x)>\theta=-\infty$. Based on this, we construct the optimal cut as outlined below. We then add it to the current problem (P^8).

$$\theta \ge \frac{689}{27}-\frac{32}{3}x_1+\frac{295}{27}x_2.$$

After adding the optimality cut, we proceed to solve the new problem (P^{10}).

$$(P^{10})\begin{cases} \min\ \phi=-x_1+4x_2, \\ \text{s.t.}, \\ x\in\mathscr{S}_{10}=\{x\in\mathscr{S}_8 \mid \theta\ge \frac{689}{27}-\frac{32}{3}x_1+\frac{295}{27}x_2\}. \end{cases}$$

After solving problem (P^{10}), we obtain the same optimal solution $x^5=(3,3)$. As we have seen previously, this obtained solution is the feasible and optimal solution for both scenarios with the penalty value $\theta=\frac{668}{27}$, see Table 1.

The algorithm now proceeds to another essential step, namely the efficiency test. This test is performed to evaluate the efficiency of the solution x^5 for both bi-objective and multi-objective problems.

Step 5: Efficiency tests. We start first by testing the efficiency of x^5 for the bi-objective problem by solving the following problem:

$$(EK^1(x^5))\begin{cases} \max\omega=\omega_1+\omega_2, \\ \text{s.t.}, \\ x\in\mathscr{S}_8, \\ \theta\ge\frac{689}{27}+\frac{32}{3}x_1-\frac{295}{27}x_2, \\ -x_1+4x_2+\omega_1=9, \\ 2x_1+x_2+\omega_2=9, \\ x_1,x_2\in\mathbb{N},\ \theta,\omega_1,\omega_2\in\mathbb{R}^+. \end{cases}$$

Table 1 – Optimal simplex table at node 8.

B	Rhs	x_9	x_{10}
x_8	$\frac{1}{3}$	$\frac{-1}{3}$	$\frac{-5}{3}$
x_3	12	-1	-4
x_4	6	-1	3
x_5	1	-2	1
x_6	1	-1	0
x_2	3	0	-1
x_7	2	0	-1
θ	$\frac{668}{27}$	$\frac{32}{27}$	$\frac{295}{27}$
x_1	3	1	0
$\bar{Z}_1$	-9	5	2
$\bar{Z}_2$	3	-4	-3
$\bar{Z}_3$	-9	-1	-4
$\bar{d}_1$	9	1	4
$\bar{d}_2$	9	-2	1

Note that $\omega = 0$, this means that x^5 is efficient for the bi-objective problem. Following this, we test its efficiency for the multi-objective problem by solving the following problem:

$$(EK^2(x^5)) \begin{cases} \max \Psi = \psi_1 + \psi_2 + \psi_3, \\ \text{s.t.,} \\ x \in \mathscr{S}_8, \\ \theta \geq \frac{689}{27} + \frac{32}{3}x_1 - \frac{295}{27}x_2, \\ -5x_1 + 2x_2 + \psi_1 = -9, \\ 4x_1 - 3x_2 + \psi_2 = 3, \\ x_1 - 4x_2 + \psi_3 = -9, \\ x_1, x_2 \in \mathbb{N},\ \theta, \psi_1, \psi_2, \psi_3 \in \mathbb{R}^+. \end{cases}$$

Here, $\Psi = 0$, that means that $x^5 = (3,3)$ is efficient for the multi-objective problem. Furthermore, we notice that x^5 is efficient for both bi-ovjective and multi-objective problems. Thus, the solution x^5 is added to the efficent set $\mathscr{X}_B = \{(3,3)\}$ and the algorithm continue the process through the last step.

Step 6: The efficient cuts. We start first by constructing the two sets $\mathscr{H}_1$ and $\mathscr{H}'_1$ by using the decreasing direction of each criterion $\widetilde{Z}_{i\in\{1,\ldots,3\}}$ for the multi-objective problem, and the decreasing direction of each criterion $\widetilde{\phi}_{i\in\{1,2\}}$ for the bi-objective problem, respectively.

From Table 1, $\mathscr{H}_1 = \{9,10\} \neq \emptyset$ and $\mathscr{H}'_1 = \{9\} \neq \emptyset$.

After adding the cuts $x_9 + x_{10} \geq 1$ and $x_9 \geq 1$ to the current problem (P^{10}), the algorithm continues the search process for other efficient solutions by solving the following new problem.

$$(P^{11}) \begin{cases} \min\ \phi = -x_1 + 4x_2, \\ \text{s.t.}, \\ x \in \mathscr{S}_{11} = \ \{x \in \mathscr{S}_{10} \mid x_9 + x_{10} \geq 1,\ x_9 \geq 1\}. \end{cases}$$

The algorithm continues the process by following the same previous steps to generate all the efficient solutions to the main problem, as shown below in the efficient set $\mathscr{X}_B$.

Final result:

The final efficient set found is $\mathscr{X}_B = \{(3,3),(2,3)\}$.

Where:

- $\mathscr{X}_C = \{(3,3),(2,3),(1,3),(0,3)\}$, the efficient solutions in terms of d^1 and d^2.
- $\mathscr{X}_E = \{(3,3),(2,3),(3,4),(2,4),(1,4),(0,4),(2,5),(1,5),(0,5),(0,6)\}$, the efficient set of the deterministic equivalent problem (5).

4.1 The search tree

The search tree presented in Figure (1) summarizes and illustrates the different steps and states of the nodes throughout the algorithm process in the previous Example 4.

5 COMPUTATIONAL RESULTS

To the best of our knowledge, this particular problem has not been studied in existing literature. Indeed, according to the available literature, no benchmark instance has been proposed to test the performance of this type of problem. For this reason, we thought of creating randomly generated instances to test the performance of our method for solving this problem. Moreover, the algorithm has been implemented in the MATLAB 2019 environment without making use of any solver and tested over the randomly generated instances, as for the efficiency and optimality tests we have used the linprog solver. In addition, all proposed solution procedures of the computational results section are performed on a computer with Intel(R) Core(TM) i3-3120M CPU @ 2.50GHz 2.50 GHz processor and 4 GB RAM.
Regarding deterministic data (objective functions and constraint coefficients) they are uncorrelated and uniformly distributed. Each component of the vector b, the entries of the matrix A, and the coefficients of the objective functions C and d were drawn randomly from discrete uniform distributions in the following ranges $[100,200], [1,100], [-100,\ 100]$ and $[-100,100]$.
The stochastic data are generated in the same way as the deterministic ones. For each component of the recourse matrix W, the penalties for constraint violations q, the matrice T and the vector h are in the intervals $[-50,50], [1,50], [-50,50]$ and $[-50,50]$, respectively. The probability for each scenario is randomly generated with the sum of probabilities equal to 1.

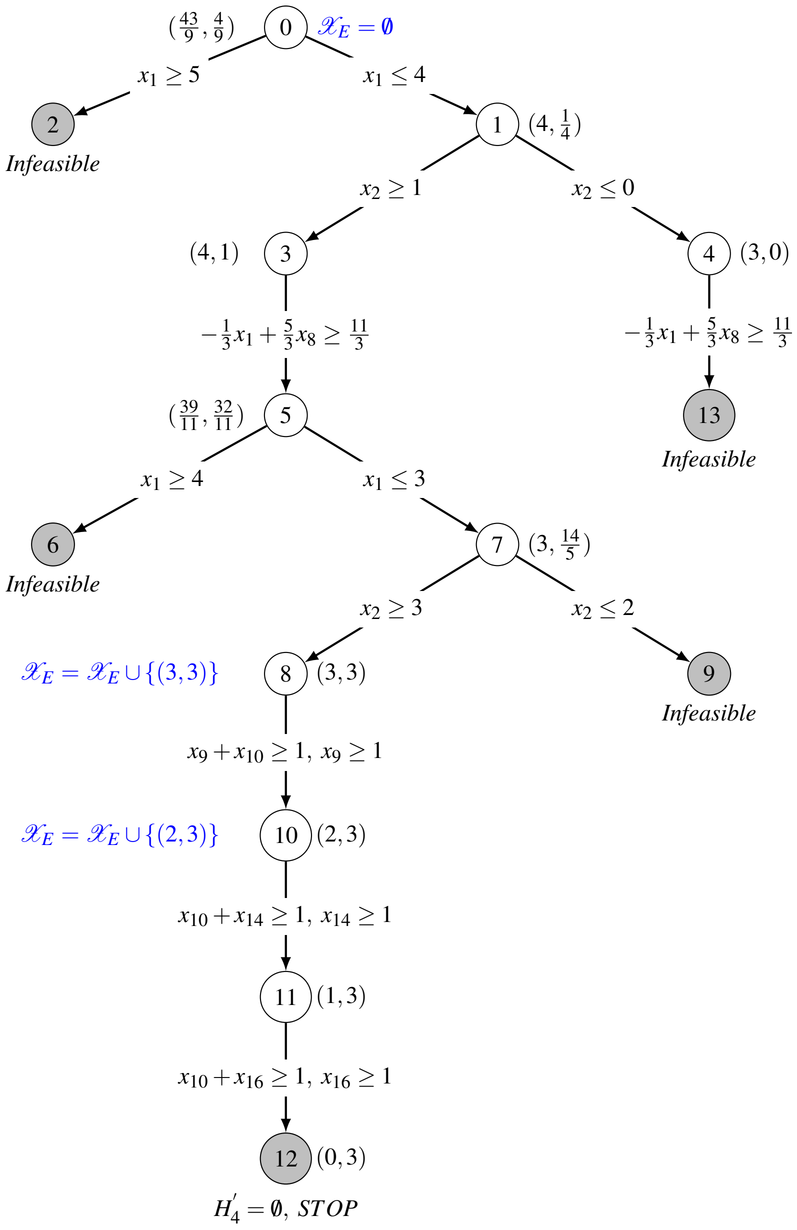
$(\frac{43}{9}, \frac{4}{9})$
0
$\mathscr{X}_E = \emptyset$
$x_1 \geq 5$
$x_1 \leq 4$
2
Infeasible
1
$(4, \frac{1}{4})$
$x_2 \geq 1$
$x_2 \leq 0$
$(4,1)$
3
4
$(3,0)$
$-\frac{1}{3}x_1 + \frac{5}{3}x_8 \geq \frac{11}{3}$
$-\frac{1}{3}x_1 + \frac{5}{3}x_8 \geq \frac{11}{3}$
$(\frac{39}{11}, \frac{32}{11})$
5
13
Infeasible
$x_1 \geq 4$
$x_1 \leq 3$
6
Infeasible
7
$(3, \frac{14}{5})$
$x_2 \geq 3$
$x_2 \leq 2$
$\mathscr{X}_E = \mathscr{X}_E \cup \{(3,3)\}$
8
$(3,3)$
9
Infeasible
$x_9 + x_{10} \geq 1,\ x_9 \geq 1$
$\mathscr{X}_E = \mathscr{X}_E \cup \{(2,3)\}$
10
$(2,3)$
$x_{10} + x_{14} \geq 1,\ x_{14} \geq 1$
11
$(1,3)$
$x_{10} + x_{16} \geq 1,\ x_{16} \geq 1$
12
$(0,3)$
$H_4' = \emptyset,\ STOP$

Figure 1 – Search tree of the example.

We have worked on 44 distinct groups of instances of the (n,m,k,R) type, where n represents the number of variables, m denotes the number of constraints, k indicates the number of objectives, and R signifies the number of scenarios, with $R = 2,3,5,10$. Additionally, for each group, we generated five instances randomly, resulting in a total of 220 instances. Our method successfully solved these instances, and the corresponding results are presented in the following Table 2.

5.1 Table Contents

Table 2 shows the results achieved by our method on five instances for each group.

- Column 1 represents the number of existing scenarios R.
- Column 2 presents the size of each instance ($n \times m \times k$), where n is the number of variables, m is the number of constraints and k is the number of objectives.
- Column 3 displays the minimum (Min), the average (Mean), and the maximum (Max) of the CPU time (in seconds).
- Column 4 reports the minimum (Min), the average (Mean), and the maximum (Max) of the number of nodes required (Nbr of Nodes) to get the final set $\mathscr{X}_B$.
- The last column ρ represents the average (Mean) of $|\mathscr{X}_B/\mathscr{X}_E|$.

The elements and cardinality of $\mathscr{X}_E$ are calculated using the algorithm presented in Ouaïl et al. (2017). The elements and cardinality of $\mathscr{X}_B$ are calculated by using the present method. In addition, we have reported a global average (Glob Avrg) in the last line for each R scenario case, which represents the total mean for all instances studied in that R case. The results from Table 2 show that average CPU time generally increases with the size of the instance and the number of scenarios. For example, for $R = 2$ the average CPU time increases from 2.47 seconds for the instance $(5 \times 3 \times 3)$ to 1407.96 seconds for the instance $(100 \times 50 \times 20)$. Similarly, for $R = 2$, the average CPU time for the instance $(20 \times 10 \times 5)$ is 8.51 seconds, while for $R = 10$ it rises to 23.96 seconds. However, this trend is not always consistent. For example, the instance $(90 \times 45 \times 20)$ shows a decrease in average CPU time from 1070.44 seconds for $R = 2$ to 954.20 seconds for $R = 3$. This is due to the fact that the average CPU time is mainly influenced by the number of visited efficient solutions. In the same instance $(90 \times 45 \times 20)$, the average number of nodes is higher for $R = 2$ at 2667.60 compared to 2415.20 for $R = 3$, which suggests that more efficient solutions were visited for $R = 2$. Thus, the average CPU time is mainly related to the number of efficient solutions explored.

Table 2 – The behavior of our method on medium and large scale instances in different scenarios and based on different criteria.

		CPU (Sec)			*Nbr of Nodes*			ρ
R	$n\times m\times k$	*Min*	*Mean*	*Max*	*Min*	*Mean*	*Max*	*Mean*
2	$5\times 3\times 3$	0.41	2.47	6.43	10	42	88	0.40
	$10\times 5\times 3$	2.08	4.49	6.80	40	79	116	0.39
	$20\times 10\times 5$	5.39	8.51	12.58	134	197.20	338	0.09
	$30\times 15\times 5$	21.15	30.11	39.15	322	490	652	0.10
	$40\times 20\times 5$	55.80	82.15	164.64	608	721.20	954	0.10
	$50\times 25\times 10$	76.74	113.55	135.27	754	918	1044	0.27
	$60\times 30\times 10$	138.55	181.98	243.79	916	1112.80	1478	0.12
	$70\times 35\times 10$	346.84	405.40	489.74	1580	1822.40	2344	0.09
	$80\times 40\times 20$	513.05	565.49	649.62	1446	1770	2206	0.11
	$90\times 45\times 20$	783.20	1070.44	1395.43	2546	2667.60	2978	0.10
	$100\times 50\times 20$	1003.98	1407.96	1897.15	2288	2816	3438	0.12
Glob	$Avrg(R=2)$	267.93	352.05	458.24	967.64	930.49	1421.45	0.17
3	$5\times 3\times 3$	0.97	3.51	11.65	8	33.20	98	0.57
	$10\times 5\times 3$	1.06	4.23	7.50	18	66.40	118	0.37
	$20\times 10\times 5$	3.54	11.78	18.64	106	190.80	256	0.29
	$30\times 15\times 5$	18.64	23.62	26.31	322	398.80	454	0.13
	$40\times 20\times 5$	38.28	60.72	88.62	576	693.20	904	0.30
	$50\times 25\times 10$	71.32	104.45	187.82	538	862.40	1424	0.19
	$60\times 30\times 10$	110.91	191.95	358.30	832	1111.60	1634	0.06
	$70\times 35\times 10$	205.65	340.08	492.87	864	1425.20	1798	0.07
	$80\times 40\times 20$	349.53	612.98	876.32	1180	2019.20	2844	0.08
	$90\times 45\times 20$	689.63	954.20	1466.57	1706	2415.20	3040	0.06
	$100\times 50\times 20$	273.65	1402.61	2272.41	746	2535.60	4096	0.12
Glob	$Avrg(R=3)$	160.24	337.29	527.91	337.29	626.91	1515.09	0.20
5	$5\times 3\times 3$	1.84	3.94	6.70	10	26.40	46	0.29
	$10\times 5\times 3$	5.65	7.89	14.33	58	84.40	154	0.40
	$20\times 10\times 5$	7.60	11.75	20.91	152	183.20	238	0.27
	$30\times 15\times 5$	22.31	27.64	33.54	302	361.20	450	0.13
	$40\times 20\times 5$	36.35	72.61	118.72	384	648.80	992	0.08
	$50\times 25\times 10$	101.96	143.70	213.36	826	1036.80	1434	0.08
	$60\times 30\times 10$	154.29	267.86	360.80	1048	1419.60	1816	0.07
	$70\times 35\times 10$	244.93	350.09	471.33	1324	1523.20	1804	0.05
	$80\times 40\times 20$	545.48	812.29	993.73	1860	2357.20	2882	0.04
	$90\times 45\times 20$	913.20	1033.18	1240.23	1944	2503.60	2966	0.04
	$100\times 50\times 20$	873.47	1282.77	1674.22	1744	2548.80	3398	0.07
Glob	$Avrg(R=5)$	264.28	364.88	467.98	877.45	1153.93	1470.91	0.14
10	$5\times 3\times 3$	1.40	5.12	8.36	8	21.20	34	0.33
	$10\times 5\times 3$	3.44	9.40	16.91	18	56	104	0.27
	$20\times 10\times 5$	8.59	23.96	44.38	68	231.20	434	0.07
	$30\times 15\times 5$	18.74	41.13	63.64	172	419.60	552	0.11
	$40\times 20\times 5$	67.11	86.19	127.58	598	762.80	994	0.16
	$50\times 25\times 10$	68.85	135.97	170.81	570	1040	1288	0.07
	$60\times 30\times 10$	167.50	261.94	304.53	904	1348	1844	0.07
	$70\times 35\times 10$	290.50	373.66	485.72	1404	1499.20	1752	0.07
	$80\times 40\times 20$	427.21	596.96	759.83	1462	1771.20	2274	0.05
	$90\times 45\times 20$	621.69	1032.79	1626.56	1934	2383.20	3042	0.06
	$100\times 50\times 20$	477.19	1263.33	1814.29	1180	2535.60	3724	0.07
Glob	$Avrg(R=10)$	195.66	348.22	492.96	756.18	1097.09	1458.36	0.12

6 CONCLUSION

This paper presents an exact method for dealing with a new type of stochastic environment problem. The problem considered involves two or more decision-makers, each seeking to optimize their utility functions over the efficient solution set in a multi-objective stochastic integer linear programming (MOSILP) problem. In addition, the proposed method can be used to identify the intersection between the efficient sets of the two stochastic multi-objective problems. Moreover, by optimizing two functions over the efficient set, decision-makers obtain a set of best compromise solutions in a finite number of iterations. It is also worth noting that the proposed method obtains this set without having to explore the whole efficient set of the MOSILP problem, which reduces the computational complexity compared to solving the MOSILP problem.

The proposed solution approach mainly relies on the combination of two techniques: the first is the L-shaped method, and the second involves an adapted branch-and-bound strategy. This combined approach is further enhanced by incorporating efficient tests and cuts to iteratively reduce the search space of the MOSILP problem. Moreover, the algorithm can be readily extended to accommodate multiple decision-makers by integrating their respective preferences into the efficient cuts. Experimental results showcase the effectiveness of the method, demonstrating excellent performance in terms of the number of iterations required while maintaining reasonable computation times. This contribution has prompted us to contemplate several future avenues of research. Our focus will be directed towards addressing the nonlinear aspect of the problem by considering the adaptation of our method to handle nonlinear multi-objective models, incorporating nonlinear preference functions. We remain committed to continuous improvement of the method in future endeavors.

References

ABBAS M & BELLAHCENE F. 2006. Cutting plane method for multiple objective stochastic integer linear programming. *European Journal of operational research*, **168**(3): 967–984.

ABBAS M & CHAABANE D. 2006. Optimizing a linear function over an integer efficient set. *European Journal of Operational Research*, **174**(2): 1140–1161.

ABDELAZIZ FB & MASRI H. 2010. A compromise solution for the multiobjective stochastic linear programming under partial uncertainty. *European Journal of Operational Research*, **202**(1): 55–59.

ADELGREN N & GUPTE A. 2022. Branch-and-bound for biobjective mixed-integer linear programming. *INFORMS Journal on Computing*, **34**(2): 909–933.

ADEYEFA AS, LUHANDJULA MK ET AL. 2011. Multiobjective stochastic linear programming: an overview. *American Journal of Operations Research*, **1**(04): 203.

AMROUCHE S & MOULAÏ M. 2012. Multi-objective stochastic integer linear programming with fixed recourse. *International Journal of Multicriteria Decision Making 9*, **2**(4): 355–378.

BELKHIRI H, CHERGUI MEA & OUAÏL FZ. 2022. Optimizing a linear function over an efficient set. *Operational Research*, pp. 1–19.

BEN ABDELAZIZ F & MASMOUDI M. 2012. A multiobjective stochastic program for hospital bed planning. *Journal of the Operational Research Society*, **63**(4): 530–538.

BENSON HP. 1984. Optimization over the efficient set. *Journal of Mathematical Analysis and Applications*, **98**(2): 562–580.

BIRGE JR & LOUVEAUX F. 2011. *Introduction to stochastic programming*. Springer Science & Business Media.

BOLAND N, CHARKHGARD H & SAVELSBERGH M. 2015. A criterion space search algorithm for biobjective integer programming: The balanced box method. *INFORMS Journal on Computing*, **27**(4): 735–754.

BOLAND N, CHARKHGARD H & SAVELSBERGH M. 2016. The L-shape search method for triobjective integer programming. *Mathematical Programming Computation*, **8**(2): 217–251.

BOLAND N, CHARKHGARD H & SAVELSBERGH M. 2017. A new method for optimizing a linear function over the efficient set of a multiobjective integer program. *European journal of operational research*, **260**(3): 904–919.

BONGO MF & SY CL. 2023. A Bi-Objective Integer Linear Optimization Model for Post-Departure Aircraft Rerouting Problem. In: *Intelligent and Transformative Production in Pandemic Times: Proceedings of the 26th International Conference on Production Research*. pp. 453–462. Springer.

BOZORGI-AMIRI A, JABALAMELI MS & MIRZAPOUR AL-E HASHEM S. 2013. A multi-objective robust stochastic programming model for disaster relief logistics under uncertainty. *OR spectrum*, **35**: 905–933.

CABALLERO R, CERDÁ E, DEL MAR MUNOZ M & REY L. 2004. Stochastic approach versus multiobjective approach for obtaining efficient solutions in stochastic multiobjective programming problems. *European Journal of Operational Research*, **158**(3): 633–648.

CABALLERO R, CERDÁ E, MUNOZ M, REY L & STANCU-MINASIAN I. 2001. Efficient solution concepts and their relations in stochastic multiobjective programming. *Journal of Optimization Theory and Applications*, **110**: 53–74.

CAO C, LI C, YANG Q, LIU Y & QU T. 2018. A novel multi-objective programming model of relief distribution for sustainable disaster supply chain in large-scale natural disasters. *Journal of Cleaner Production*, **174**: 1422–1435.

CHAABANE D & MEBREK F. 2014. Optimization of a linear function over the set of stochastic efficient solutions. *Computational Management Science*, **11**(1): 157–178.

CHAIBLAINE Y & MOULAÏ M. 2021. An exact method for optimizing a quadratic function over the efficient set of multiobjective integer linear fractional program. *Optimization Letters*, pp. 1–15.

CHAIBLAINE Y, MOULAÏ M & CHERFAOUI Y. 2020. An exact method for optimizing two linear fractional functions over the efficient set of a Multiobjective Integer Linear Fractional Program. *arXiv preprint arXiv:2003.05364*, .

CHERFAOUI Y & MOULAÏ M. 2021. Biobjective optimization over the efficient set of multiobjective integer programming problem. *Journal of Industrial & Management Optimization*, **17**(1): 117.

DOS SANTOS FSP & OLIVEIRA F. 2019. An enhanced L-Shaped method for optimizing periodic-review inventory control problems modeled via two-stage stochastic programming. *European Journal of Operational Research*, **275**(2): 677–693.

DRICI W, OUAIL FZ & MOULAÏ M. 2018. Optimizing a linear fractional function over the integer efficient set. *Annals of Operations Research*, **267**(1): 135–151.

ECKER JG & KOUADA I. 1978. Finding all efficient extreme points for multiple objective linear programs. *Mathematical Programming*, **14**(1): 249–261.

ECKER JG & SONG JH. 1994. Optimizing a linear function over an efficient set. *Journal of Optimization Theory and Applications*, **83**(3): 541–563.

FARKAS J. 1902. Theorie der einfachen Ungleichungen. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, **1902**(124): 1–27.

GOICOECHEA A, DUKSTEIN L & BULFIN R. 1976. Multiobjective stochastic programming the PROTRADE-method. *Operation Research Society of America*, .

HALFFMANN P, SCHÄFER LE, DÄCHERT K, KLAMROTH K & RUZIKA S. 2022. Exact algorithms for multiobjective linear optimization problems with integer variables: A state of the art survey. *Journal of Multi-Criteria Decision Analysis*, .

HIGLE JL & SEN S. 1991. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of operations research*, **16**(3): 650–669.

JAHANSHAHLOO GR, LOTFI FH, SHOJA N & TOHIDI G. 2004. A method for generating all the efficient solutions of a 0-1 multi-objective linear programming problem. *Asia-Pacific Journal of Operational Research*, **21**(01): 127–139.

JORGE JM. 2009. An algorithm for optimizing a linear function over an integer efficient set. *European Journal of Operational Research*, **195**(1): 98–103.

KALL P. 1976. Stochastic linear programming. Econometrics and operations research, vol XXI.

KALL P, WALLACE SW & KALL P. 1994. *Stochastic programming*. Springer.

KIRLIK G & SAYIN S. 2014. A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. *European Journal of Operational Research*, **232**(3): 479–488.

LI C & GROSSMANN IE. 2018. An improved L-shaped method for two-stage convex 0–1 mixed integer nonlinear stochastic programs. *Computers & Chemical Engineering*, **112**: 165–179.

LOKMAN B. 2021. Optimizing a linear function over the nondominated set of multiobjective integer programs. *International Transactions in Operational Research*, **28**(4): 2248–2267.

LOKMAN B & KÖKSALAN M. 2013. Finding all nondominated points of multi-objective integer programs. *Journal of Global Optimization*, **57**(2): 347–365.

MOAYEDI M & SADEGHIAN R. 2023. A multi-objective stochastic programming approach with untrusted suppliers for green supply chain design by uncertain demand, shortage, and transportation costs. *Journal of Cleaner Production*, **408**: 137007.

MOTAHARI R, ALAVIFAR Z, ANDARYAN AZ, CHIPULU M & SABERI M. 2023. A multi-objective linear programming model for scheduling part families and designing a group layout in cellular manufacturing systems. *Computers & Operations Research*, **151**: 106090.

MOULAÏ M & DRICI W. 2018. An indefinite quadratic optimization over an integer efficient set. *Optimization*, **67**(8): 1143–1156.

OBAL TM, VOLPI NMP & MILOCA SA. 2013. Multiobjective approach in plans for treatment of cancer by radiotherapy. *Pesquisa Operacional*, **33**: 269–282.

OUAÏL F, EA M C & MOULAÏ M. 2017. An exact method for optimizing a linear function over an integer efficient set. *WSEAS Transactions on Circuits and Systems*, **16**(3): 141–148.

ÖZLEN M & AZIZOĞLU M. 2009. Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operational Research*, **199**(1): 25–35.

PHILIP J. 1972. Algorithms for the vector maximization problem. *Mathematical programming*, **2**(1): 207–229.

RAMEZANI M, BASHIRI M & TAVAKKOLI-MOGHADDAM R. 2013. A new multi-objective stochastic model for a forward/reverse logistic network design with responsiveness and quality level. *Applied mathematical modelling*, **37**(1-2): 328–344.

RASMI SAB & TÜRKAY M. 2019. GoNDEF: an exact method to generate all non-dominated points of multi-objective mixed-integer linear programs. *Optimization and Engineering*, **20**: 89–117.

REN C, XIE Z, ZHANG Y, WEI X, WANG Y & SUN D. 2021. An improved interval multi-objective programming model for irrigation water allocation by considering energy consumption under multiple uncertainties. *Journal of Hydrology*, **602**: 126699.

SHIDPOUR H, SHAHROKHI M & BERNARD A. 2013. A multi-objective programming approach, integrated into the TOPSIS method, in order to optimize product design; in three-dimensional concurrent engineering. *Computers & Industrial Engineering*, **64**(4): 875–885.

SIERRA ALTAMIRANDA A & CHARKHGARD H. 2019. A new exact algorithm to optimize a linear function over the set of efficient solutions for biobjective mixed integer linear programs. *INFORMS Journal on Computing*, **31**(4): 823–840.

SOYLU B. 2015. Heuristic approaches for biobjective mixed 0–1 integer linear programming problems. *European Journal of Operational Research*, **245**(3): 690–703.

SYLVA J & CREMA A. 2004. A method for finding the set of non-dominated vectors for multiple objective integer linear programs. *European Journal of Operational Research*, **158**(1): 46–55.

TAMBY S & VANDERPOOTEN D. 2021. Enumeration of the nondominated set of multiobjective discrete optimization problems. *INFORMS Journal on Computing*, **33**(1): 72–85.

TEGHEM J. 1983. Multiobjective and stochastic linear programming. *Foundations of Control Engineering*, **8**(3–4): 225–232.

TEGHEM J. 1990. “Strange”: An Interactive Method for Multiobjective Stochastic Linear Programming, and “Strange-Momix” Its Extension to Integer Variables. In: *Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty*. pp. 103–115. Springer.

TORRES JJ, LI C, APAP RM & GROSSMANN IE. 2022. A review on the performance of linear and mixed integer two-stage stochastic programming software. *Algorithms*, **15**(4): 103.

URLI B & NADEAU R. 1990. Multiobjective stochastic linear programming with incomplete information: a general methodology. In: *Stochastic versus fuzzy approaches to multiobjective mathematical programming under uncertainty*. pp. 131–161. Springer.

UŠPURIENĖ A, SAKALAUSKAS L & GRICIUS G. 2018. Modified L-Shaped Decomposition Method with Scenario Aggregation for a Two-Stage Stochastic Programming Problem. *Information Technology and Control*, **47**(4): 728–738.

VAN SLYKE RM & WETS R. 1969. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, **17**(4): 638–663.

ZERDANI O & MOULAI M. 2011. *Optimization over an integer efficient set of a multiple objective linear fractional problem.*

ZHANG Y, FU Z, XIE Y, LI Z, LIU Y, HU Q & GUO H. 2021. Multi-objective programming for energy system based on the decomposition of carbon emission driving forces: A case study of Guangdong, China. *Journal of Cleaner Production*, **309**: 127410.

How to cite

BADAOUI I, MOULAÏ M, CHAIBLAINE Y & CHAABANE D. 2024. Biobjective integer stochastic optimization over the integer stochastic efficient set. *Pesquisa Operacional*, **44**: e281853. doi: 10.1590/0101-7438.2023.043.00281853.