Representing the Nondominated Set in Multi-objective Mixed-integer Programs

Ilgın Doğan

Department of Industrial Engineering and Operations Research, University of California, Berkeley, California 94720, USA

Banu Lokman*

Portsmouth Business School, Centre of Operations Research and Logistics, University of Portsmouth, Portsmouth PO1 3DE, UK

Murat Köksalan

Ross School of Business, University of Michigan, Ann Arbor, Michigan 48109, USA

Department of Industrial Engineering, Middle East Technical University, Ankara, 06800, Turkey

Abstract

In this paper, we consider generating a representative subset of nondominated points at a prespecified precision in multi-objective mixed-integer programs (MOMIPs). The number of nondominated points grows exponentially with problem size and finding all nondominated points is typically hard in MOMIPs. Representing the nondominated set with a small subset of nondominated points is important for a decision maker to get an understanding of the layout of solutions. The shape and density of the nondominated points over the objective space may be critical in obtaining a set of solutions that represent the nondominated set well. We develop an exact algorithm that generates a representative set guaranteeing a prespecified precision. Our experiments on a variety of problems demonstrate that our algorithm outperforms existing approaches in terms of both the cardinality of the representative set and computation times.

Keywords: multi-objective mixed-integer programming, representative set, coverage gap

1. Introduction

Complex decision problems require the evaluation of multiple conflicting objectives in order to make informed decisions. Multi-objective mixed-integer programs (MOMIPs) become prohibitive to solve and find the nondominated set as the problem size grows. Furthermore, the

*Corresponding author
Email address: banu.lokman@port.ac.uk (Banu Lokman)

difficulty of evaluating the solutions and searching for the most preferred solution still remains. Therefore, finding a subset of nondominated points that represents the nondominated set well is important for MOMIPs.

Ehrgott (2005, pp. 205-208) shows that multi-objective integer programs (MOIPs) are typically intractable, that is, the set of all nondominated points cannot be generated by an algorithm that is polynomially solvable in the size of the problem instance. However, there are efficient approaches that generate all nondominated points for multi-objective integer programs (MOIPs) as long as the problem size is not too large. These approaches utilize efficient decomposition methods to partition the objective space into subsets and search for a new nondominated point at each iteration (see Laumanns et al., 2006; Lokman and Köksalan, 2013; Kırlık and Sayın, 2014; Özlen et al., 2014; Dächert and Klamroth, 2015; Boland et al., 2016, 2017b; Dächert et al., 2017).

Evolutionary algorithms (EAs) are widely used to handle computationally-expensive multi-objective optimization problems. They are helpful in solving less structured complex problems. Many attempt to approximate the whole nondominated set (see Deb, 2001). The EAs are heuristic in nature and there is no guarantee for any of the generated solutions to be nondominated. They only provide approximations to nondominated sets without providing any performance guarantee.

In response to the prohibitive computational effort required for generating the nondominated set, an increasing number of researchers have been addressing the problem of generating a representative subset of nondominated points satisfying certain quality measures. The quality measures mainly consider two notions: how closely the representative points cover the nondominated set (coverage quality) and the number of representative points selected. These two notions are conflicting in nature in the sense that as the number of representative points increase, one expects to cover the nondominated set better. The perfect representation occurs only when each nondominated point becomes a representative point, implying that the number of representative points equals the number of nondominated points. Sayın (2003), Karasakal and Köksalan (2009), and Shao and Ehrgott (2016) generate discrete representations of the nondominated frontiers in multi-objective linear programs (MOLPs). Hamacher et al. (2007) and Vaz et al. (2015) address the problem of finding a representative subset for discrete optimization problems and design algorithms satisfying a number of quality measures. However, these algorithms are applicable for only bi-objective case and extension to multi-objective case is not straightforward. Kuhn and Ruzika (2017) extends the Box-Algorithm of Hamacher et al. (2007) to three-objective case in order to produce a representative set with desired coverage quality.

Since finding representations of the nondominated sets for MOMIPs with more than two objectives may become an intractable task, the existing literature addressing the problem of generating representative set that guarantees a prespecified precision is limited. The algorithms developed by Sylva and Crema (2007) and Masin and Bukchin (2008) use a similar procedure to each other to generate a set of nondominated points that represent the whole nondominated set of MOMIPs with a prespecified quality level. The quality of the representative subsets is measured based on a coverage gap measure. More recently, Ceyhan et al. (2019) developed efficient algorithms to generate well-spread representative subsets for MOMIPs. Their Subset-based Approach (SBA), implements the same concepts as Sylva and Crema (2007) and Masin and Bukchin (2008) with a computationally-efficient procedure. Their Territory-defining Algorithm (TDA) constructs a territory around each representative point to create prohibited regions for new representative points while maintaining the desired coverage quality. Their Surface Projection Algorithm (SPA) tries to generate a prespecified number of representative points in such a way to achieve the best coverage quality.

Although the existing approaches produce representative sets that guarantee a predetermined desired coverage gap value, they perform a greedy myopic search to find the next nondominated point and choose the one that offers the most immediate improvement in the coverage quality. Furthermore, they follow a generic approach without utilizing any problem specific information in the design of the algorithm. Özarik et al. (2020) demonstrate the distributions of the non-dominated points on a variety of multi-objective integer programs. Based on their experiments, they observe that the nondominated points are, typically, not uniformly distributed over the objective function space.

In this study, we develop an exact algorithm, Territory-excluded Supported Generating Algorithm (TSGA), that generates a set of nondominated points that represent the nondominated sets of MOMIPs satisfying a prespecified coverage gap value. Different from the existing studies, TSGA aims to minimize the size of the final representative set without exceeding the desired coverage quality. It solves a weighted sum problem, choosing the weights with a heuristic procedure that utilizes problem-specific information to guide the search towards dense regions of the nondominated set. The problem-specific weight-selection mechanism leads the search to representative points from dense regions of the nondominated set of the problem at hand. This in turn forms representative points that have high representation power and enables to satisfy the desired coverage with fewer points. Reducing the size of the representative set with a coverage performance guarantee also brings a computational advantage since finding each nondominated point in MOMIPs is costly.

The paper is organized as follows. In Section 2, we present the preliminaries and a brief review of relevant literature. We develop our approach in Section 3 and present an illustrative example in Section 4. We present computational results in Section 5 and make concluding remarks in Section 6.

2. Background

In this section, we provide relevant background information and review the state-of-the-art approaches with which we will compare our results.

2.1. Preliminaries

A MOMIP may be defined as:

$$\begin{aligned} &\text{“Max”} \ \mathbf{z}(\mathbf{x}) \\ &\text{s.t.} \quad \mathbf{x} \in X \\ &\qquad \mathbf{x} = (\mathbf{u}, \mathbf{v})^T \\ &\qquad \mathbf{u} \in \mathbb{Z}^p, \ \mathbf{v} \in \mathbb{R}^{d-p} \end{aligned}$$

where $\mathbf{z}(\mathbf{x}) = \{z_1(\mathbf{x}), z_2(\mathbf{x}), ..., z_m(\mathbf{x})\}$ is an m-dimensional vector in the objective space, $z_k(\mathbf{x})$ is the k^{th} objective function, p is the dimension of the integer-valued decision variables, $X \subseteq \mathbb{R}^d$ is the feasible decision space and $Z \subseteq \mathbb{R}^m$ is the feasible objective space. We assume that the feasible space is bounded and not empty. Without loss of generality, all objectives are of maximization type. We use quotation marks to emphasize that the maximization of a vector is not a well-defined mathematical operation.

Definition 1. *$\mathbf{x}^i \in X$ is an efficient solution if $\nexists \ \mathbf{x}^j \in X$ such that $z_k(\mathbf{x}^j) \geq z_k(\mathbf{x}^i)$ for all $k = 1, \ldots, m$ and $z_k(\mathbf{x}^j) > z_k(\mathbf{x}^i)$ for at least one k. If $\mathbf{x}^i \in X$ is an efficient solution, then $\mathbf{z}(\mathbf{x}^i)$ is said to be a nondominated point.*

The set of all efficient solutions is denoted as $X_E \subseteq X$, and the set of all nondominated points is denoted as $Z_{ND} \subseteq Z$.

Definition 2. *The ideal point $\mathbf{z}^{IP} = \{z_1^{IP}, z_2^{IP}, \ldots, z_m^{IP}\}$ is defined as $z_k^{IP} = \max_{\mathbf{z} \in Z}(z_k) \ \forall k$.*

Definition 3. *The nadir point $\mathbf{z}^{NP} = \{z_1^{NP}, z_2^{NP}, \ldots, z_m^{NP}\}$ is defined as $z_k^{NP} = \min_{\mathbf{z} \in Z_{ND}}(z_k) \ \forall k$.*

Minimizing over the nondominated set is not computationally straightforward for more than two objectives. Although there are efficient algorithms to find the nadir point (Köksalan and

Lokman, 2015; Kırlık and Sayın, 2015; Boland et al., 2017a), heuristic approaches are also used when the knowledge of the exact nadir values is not required. A common approach is to estimate $\mathbf{z}^{NP}$ from the payoff table.

Definition 4. *Let* $\mathbf{x}^{IP}(k) = \arg\max_{\mathbf{x}\in X}(z_k(\mathbf{x}))$ *and* $\tilde{z}_k^{NP} = \min_{t=1,\dots,m} z_k(\mathbf{x}^{IP}(t))$ $k = 1,\dots,m$. *Then the payoff nadir is* $\tilde{\mathbf{z}}^{NP} = \{\tilde{z}_1^{NP}, \tilde{z}_2^{NP}, \dots, \tilde{z}_m^{NP}\}$.

Definition 5. *Let* $\mathbf{x}^* \in X_E$. *If* $\exists\ \boldsymbol{\lambda} > 0$ *such that* $\mathbf{x}^* = \arg\max_{\mathbf{x}\in X} \sum_{k=1}^{m} \lambda_k z_k(\mathbf{x})$, *then* $\mathbf{x}^*$ *is called a supported efficient solution and* $\mathbf{z}(\mathbf{x}^*)$ *is called a supported nondominated point.*

For a given $\boldsymbol{\lambda} > 0$, there might exist more than one optimal solution to the weighted sum problem $\max_{\mathbf{x}\in X} \sum_{k=1}^{m} \lambda_k z_k(\mathbf{x})$. Definition 5 is valid for all such equivalent solutions. Since the feasible set is nonconvex in MOMIPs, unsupported nondominated points that cannot be found by solving a weighted sum problem might also exist (see Ehrgott, 2005, p. 204).

Definition 6. *Let* $\mathbf{y}^i, \mathbf{y}^j \in Z$ *and* $\alpha \in \mathbb{R}_{\geq}$. *If* $y_k^i < y_k^j + \alpha$ *for all* $k = 1,\dots,m$, $\mathbf{y}^i$ *is said to be strictly* α*-dominated by* $\mathbf{y}^j$.

2.2. Measure of the Quality of Representation

In order to assess the quality of the representative subsets of nondominated points, several quality metrics have been proposed. Sayın (2000) defines three quality measures: coverage, cardinality and spacing (see also Faulkenberg and Wiecek, 2010). Coverage measures evaluate the quality of the representation, cardinality is the size of the representative set, and spacing is a measure of the distances between representative points. Since the coverage measures quantify how well the nondominated set is represented, they are widely used by the researchers (for example, see Sylva and Crema, 2007; Masin and Bukchin, 2008; Ceyhan et al., 2019). We use this metric as defined by Zitzler et al. (2003) and refer to it as the coverage gap.

Definition 7. *Let* $R \subseteq Z_{ND}$ *be a representative subset and* $\mathbf{r}(\mathbf{z}) = (r_1(\mathbf{z}),\dots,r_m(\mathbf{z}))$ *be a representative point in* R. $\mathbf{r}(\mathbf{z}) \in R$ *is said to be the representative of nondominated point* $\mathbf{z} \in Z_{ND}$ *if* $\max_{k=1,\dots,m}(z_k - r_k(\mathbf{z})) = \min_{\mathbf{y}\in R} \max_{k=1,\dots,m}(z_k - y_k)$. *The error in the representation of* $\mathbf{z}$ *by* R *is defined as* $\alpha_R(\mathbf{z}) = \max_{k=1,\dots,m}(z_k - r_k(\mathbf{z}))$ *and the coverage gap of* R *is defined as* $\alpha_R = \max_{\mathbf{z}\in Z_{ND}} \alpha_R(\mathbf{z})$.

In order to have both the objective functions and the coverage values comparable, we scale all objectives in the same range as the coverage values, $[0, 1]$. For brevity, we keep the original notation in the rest of the paper, with the understanding that the values are scaled. Figure 1 demonstrates the coverage gap value on a two-objective example problem. Let $Z_{ND} = \{\mathbf{z}^1,\dots,\mathbf{z}^8\}$ and

$R = \{\mathbf{z}^3, \mathbf{z}^8\}$ be the representative set. Then, based on Definition 7, the representative point corresponding to each nondominated point is found as: $\mathbf{r}(\mathbf{z}^1) = \mathbf{r}(\mathbf{z}^2) = \mathbf{r}(\mathbf{z}^4) = \mathbf{r}(\mathbf{z}^5) = \mathbf{z}^3$, $\mathbf{r}(\mathbf{z}^6) = \mathbf{r}(\mathbf{z}^7) = \mathbf{z}^8$ and $\alpha_R(\mathbf{z}^1) = 0.2$, $\alpha_R(\mathbf{z}^2) = 0.1$, $\alpha_R(\mathbf{z}^4) = 0.1$, $\alpha_R(\mathbf{z}^5) = 0.2$, $\alpha_R(\mathbf{z}^6) = 0.3$, $\alpha_R(\mathbf{z}^7) = 0.1$. The point that has the worst representation, $\mathbf{z}^6$, determines the coverage gap, i.e. $\alpha_R = \alpha_R(\mathbf{z}^6) = 0.3$.

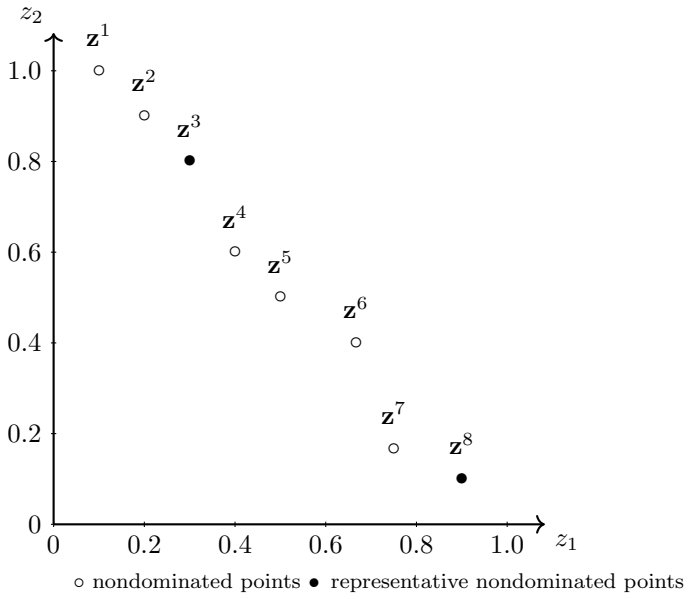


Figure 1: Demonstration of the coverage gap.

2.3. Existing Approaches

There are a number of algorithms in the literature that have been developed to generate representative subsets with a desired precision level for MOMIPs.

The Diversity Maximization Algorithm (DMA) of Masin and Bukchin (2008) and the algorithm developed by Sylva and Crema (2007) (SC) are identical in terms of the concepts with which they choose the representative points sequentially. Given a set of representative points, they both choose the next representative point as the nondominated point that has the worst representation. They keep adding representative points one at a time until a stopping condition is satisfied. Given a representative set R, to find the next representative point, $m|R|$ binary variables and $(m+1)|R|$ linear constraints are added to the original model to identify the next representative point. The computational burden grows prohibitively as the size of the representative set grows. The Subset-based Approach (SBA) of Ceyhan et al. (2019) improves the computational efficiency of these algorithms. Instead of solving the whole model with additional binary variables and constraints, SBA adapts the decomposition method of Lokman and

Köksalan (2013) and partitions the solution space into subspaces defined by imposing sets of lower bounds on the objective values. SBA conducts a search over these subsets to find the worst represented nondominated point as the new representative point. The algorithms terminate when the DM is satisfied with the coverage gap value or the size of the representative set exceeds a threshold value.

The Territory-defining Algorithm (TDA) of Ceyhan et al. (2019) generates a representative set that satisfies a prespecified coverage gap value, α. TDA incorporates the desired coverage gap value into the solution procedure by constructing territories to exclude not only the dominated regions but also α-dominated regions by the current representative set. Specifically, the territory of a representative point $\mathbf{y}$ is defined by using three sets, $\mathbf{y}_D = \{z_k(\mathbf{x}) \leq y_k,\ k = 1, \ldots, m \mid \mathbf{x} \in X\}$, $\mathbf{y}^D = \{z_k(\mathbf{x}) \geq y_k\ k = 1, \ldots, m \mid \mathbf{x} \in X\}$, and $\mathbf{y}_{\alpha D} = \{z_k(\mathbf{x}) \leq y_k + \alpha\ \ k = 1, \ldots, m \mid \mathbf{x} \in X\}$ where $\mathbf{y}_D$ is the set dominated by $\mathbf{y}$, $\mathbf{y}^D$ is the set dominating $\mathbf{y}$, and $\mathbf{y}_{\alpha D}$ is the set that is α-dominated by $\mathbf{y}$. The territory around $\mathbf{y}$ is defined as $T_{\mathbf{y}} = \{\mathbf{y}_{\alpha D} \backslash (\mathbf{y}_D \cup \mathbf{y}^D)\} \cup \{\mathbf{y}\}$ as depicted in Figure 5. TDA uses the same logic as DMA, SC, and SBA to choose the current least represented nondominated point as the next representative point. Maximizing $\alpha + \epsilon \sum_{i=1}^{k} z_k(\mathbf{x})$ value at each iteration, TDA finds the nondominated point that is farthest from the existing representative set, breaking ties in such a way to guarantee a nondominated point (using $\epsilon \sum_{i=1}^{k} z_k(\mathbf{x})$ where $\epsilon > 0$ is a sufficiently small positive constant). TDA uses the same idea as DMA, SC, and SBA in finding the representative points, but differs from the latter by creating territories in order to improve computational efficiency.

3. Territory-excluded Supported Generating Algorithm (TSGA)

There are two main concerns in choosing a representative subset of the nondominated set in MOMIPs: (i) The representation should cover the whole nondominated set well so that it gives a good understanding of the spectrum of nondominated points, (ii) the cardinality of the representative set should be small, as this is closely related with the computational burden. The computational burden in obtaining the representative set is an important concern as finding a nondominated point of a MOMIP from practice could be quite complex to solve. Furthermore, we will still need to study this set further in order to achieve our main goal of converging towards preferred solutions. Typically, the two concerns are conflicting in nature as the quality of representation improves with additional representative points.

3.1. The Solution Procedure

We develop an algorithm, Territory-excluded Supported Generating Algorithm (TSGA), that aims at producing a small number of representative points while satisfying a prespecified coverage gap value. The main strengths of TSGA over the existing approaches are that it estimates and takes into account the problem-specific distributional properties of the existing nondominated points and that it performs a non-myopic search over the nondominated frontier by accounting for the subsequent opportunities in improving the final cardinality of the generated representative set. We next provide a general overview of our algorithm. We will present the specifics of the algorithm in later sections.

TSGA iteratively generates representative points and reduces the feasible set by excluding the regions that are represented by those points as defined by a given coverage gap value, α. This process continues until the feasible set becomes empty, implying that all remaining nondominated points are represented within the coverage gap value of $\alpha > 0$. To do this, for each previously found representative point, $\mathbf{y}^j$, TSGA, as in TDA, constructs a territory $T_{\mathbf{y}^j}$, that includes the nondominated region that is strictly α-dominated by $\mathbf{y}^j$. Then, at any iteration n, TSGA reduces the feasible objective space by excluding $T_{\mathbf{y}^j}$ for all $j = 1, \ldots, n-1$ and generates a new representative point, $\mathbf{y}^n$, solving $(\boldsymbol{P_n^{TSGA}})$. While some of the existing studies generate the worst represented point at each iteration by maximizing the value of α, TSGA sets the value of α to the desired coverage gap value and solves a weighted sum problem on the truncated feasible set. In other words, TSGA finds the new representative point, y^n, maximizing a weighted sum of the individual objectives on the reduced feasible space. Since we maximize a linear objective function, $\mathbf{y}^n$ is a supported nondominated point for the reduced feasible space. However, $\mathbf{y}^n$ may be an unsupported nondominated point of the original feasible space.

$(\boldsymbol{P_n^{TSGA}})$:

$$
\begin{aligned}
& \text{Max} && \sum_{k=1}^{m} \lambda_k z_k(\mathbf{x}) && \\
\text{s.to} \quad & z_k(\mathbf{x}) \geq (y_k^j + \alpha) t_{kj} + M_k(1 - t_{kj}) && k = 1, \ldots, m,\ j = 1, \ldots, n-1 \\
& \sum_{k=1}^{m} t_{kj} \geq 1 && j = 1, \ldots, n-1 \\
& t_{kj} \in \{0, 1\} && k = 1, \ldots, m,\ j = 1, \ldots, n-1 \\
& \mathbf{x} \in \mathbf{X} &&
\end{aligned}
$$

In $(\boldsymbol{P_n^{TSGA}})$, $\lambda_k > 0$ and M_k denote the weight and lower bound (minimum value objective k

can take within the feasible space) for objective k, respectively. For $t_{kj} = 1$, the first constraint set imposes $z_k(\mathbf{x}) \geq y_k^j + \alpha$. For $t_{kj} = 0$, the corresponding constraint becomes redundant for those values of k and j since M_k is chosen so that $z_k(\mathbf{x}) \geq M_k$ is always satisfied. Since $\lambda_k > 0$, the resulting point, $\mathbf{y}^n$, is guaranteed to be nondominated and since the model forces $t_{kj} = 1$ for at least one k for each j, its error of representation by the existing representative set, $\mathbf{y}^j, j = 1, \ldots, n-1$, is at least α. If $\boldsymbol{(P_n^{TSGA})}$ is infeasible, then there are no nondominated points outside the territories of the current representative points and TSGA stops with a representative set of $R_n = \{\mathbf{y}^1, \ldots, \mathbf{y}^{n-1}\}$. This guarantees that the coverage gap of the representative set generated by TSGA is below the desired coverage gap, α. We next formalize these in Proposition 1.

Proposition 1. *Let $\alpha, \lambda_k \in \mathbb{R}_{>0}$, M_k be the lower bound for objective k, and $R_n = \{\mathbf{y}^1, \ldots, \mathbf{y}^{n-1}\}$ be the available set of representative points.*

(i) If $\boldsymbol{(P_n^{TSGA})}$ is feasible and $\mathbf{y}^n$ denotes the objective function vector corresponding to the optimal solution of $\boldsymbol{(P_n^{TSGA})}$, then $\mathbf{y}^n$ is a nondominated point that satisfies $\alpha_{R_n}(\mathbf{y}^n) \geq \alpha$.

(ii) If there does not exist a feasible solution to $\boldsymbol{(P_n^{TSGA})}$, the coverage gap value of R_n satisfies $\alpha_{R_n} < \alpha$.

Proof: *(i)* Suppose the optimal point, $\mathbf{y}^n$, is dominated. Then, there should exist a nondominated point $\mathbf{z}(\mathbf{x}) \in Z$ such that $z_k(\mathbf{x}) \geq y_k^n$ for all $k = 1, \ldots, m$ and $z_k(\mathbf{x}) > y_k^n$ for at least one $k = 1, \ldots, m$. Let $Z_{\boldsymbol{(P_n^{TSGA})}} \in Z$ be the feasible objective space at iteration n and B_n represent the set of index pairs (k, j) for which $t_{kj} = 1$ at the optimal solution, i.e., $y_k^n \geq y_k^j + \alpha$ for $(k, j) \in B_n$. Since $z_k(\mathbf{x}) \geq y_k^n$ for all $k = 1, \ldots, m$, $z_k(\mathbf{x}) \geq y_k^n \geq y_k^j + \alpha$ holds for all $(k, j) \in B_n$. This, in turn, implies $\mathbf{z}(\mathbf{x}) \in Z_{\boldsymbol{(P_n^{TSGA})}}$. Since $\lambda_k > 0$ for all $k = 1, \ldots, m$, $\sum_{k=1}^m \lambda_k z_k(\mathbf{x}) > \sum_{k=1}^m \lambda_k y_k^n$ which contradicts the fact that $\mathbf{y}^n$ is the objective vector at the optimal solution. Therefore, $\mathbf{y}^n$ is nondominated. Now suppose that $\alpha_{R_n}(\mathbf{y}^n) < \alpha$ and $\mathbf{r}(\mathbf{y^n}) = \mathbf{y}^{j*}$ for some $\mathbf{y}^{j*} \in R_n$. Then, $\max\limits_{k=1,\ldots,m} (y_k^n - y_k^{j*}) = \min\limits_{\mathbf{y}^j \in R_n} \max\limits_{k=1,\ldots,m} (y_k^n - y_k^j)$ and $\alpha_{R_n}(\mathbf{y}^n) = \max\limits_{k=1,\ldots,m} (y_k^n - y_k^{j*}) < \alpha$ by Definition 7. Hence, $y_k^n < y_k^{j*} + \alpha$ implying that $t_{kj*} = 0$, for all $k = 1, \ldots, m$. This violates the constraint $\sum\limits_{k=1}^{m} t_{kj*} \geq 1$, contradicting the feasibility of $\mathbf{y}^n$. *(ii)* If $\boldsymbol{(P_n^{TSGA})}$ is infeasible and $\alpha_{R_n} \geq \alpha$, then there should exist some $\mathbf{z}(\mathbf{x}) \in Z_{ND}$ for which $\alpha_{R_n}(\mathbf{z}(\mathbf{x})) = \min\limits_{\mathbf{y}^j \in R_n} \max\limits_{k=1,\ldots,m} (z_k(\mathbf{x}) - y_k^j) \geq \alpha$ by Definition 7. That is, $z_k(\mathbf{x}) - y_k^j \geq \alpha$ is satisfied for each $j = 1, \ldots, n-1$, for some $k = 1, \ldots, m$ implying that $\mathbf{z}(\mathbf{x})$ satisfies the constraints imposed by $\boldsymbol{(P_n^{TSGA})}$. Existence of such a $\mathbf{z}(\mathbf{x})$ contradicts to $\boldsymbol{(P_n^{TSGA})}$ being infeasible. □

Although the first part of the proof of *(i)* is a well known result in the multi-objective optimization literature, we provide it here for the sake of completeness. Note that Proposition

1 does not need the existing representative points to be nondominated, it holds even if some or all of them were dominated. However, since TSGA starts with an empty set and generates representative points by solving $(\boldsymbol{P_n^{TSGA}})$ at each iteration, the representative set produced by TSGA does not contain any dominated points. In order to solve $(\boldsymbol{P_n^{TSGA}})$, TSGA adapts the decomposition and search procedure of Dächert et al. (2017) that has been developed to generate all nondominated points and shown to outperform existing methods in terms of computational efficiency. To find an optimal solution to $(\boldsymbol{P_n^{TSGA}})$ efficiently, we decompose the *search region*, i.e., the feasible set of $(\boldsymbol{P_n^{TSGA}})$ that we denote as $Z_{(\boldsymbol{P_n^{TSGA}})}$, into *search zones (subspaces)* and conduct a search for a new nondominated point over these zones by solving submodels. A *search zone* s is characterized by a lower bound vector $\mathbf{lb}^s = \{lb_1^s, \ldots, lb_m^s\}$ with m components corresponding to m objective functions and defined by the set $\{\mathbf{z} \in Z : z_k(\mathbf{x}) \geq lb_k^s,\ k = 1, 2, \ldots, m\}$. Therefore, we define the submodel $(\boldsymbol{P_{TSGA}^s})$ corresponding to search zone s as follows:

$(\boldsymbol{P_{TSGA}^s})$:

$$\text{Max} \quad \sum_{k=1}^{m} \lambda_k z_k(\mathbf{x})$$

s.t.

$$z_k(\mathbf{x}) \geq lb_k^s \quad k = 1, 2, \ldots, m$$

$$\mathbf{x} \in X$$

We refer to LB^n as a *lower bound set* for the search region $Z_{(\boldsymbol{P_n^{TSGA}})}$ such that the union of the search zones defined by $\mathbf{lb}^s \in LB^n$ constitutes $Z_{(\boldsymbol{P_n^{TSGA}})}$. A formal definition of the lower bound set is provided in Definition 8.

Definition 8. *Given a set of representative points* $R_n = \{\mathbf{y}^1, \ldots, \mathbf{y}^{n-1}\}$, LB^n *is defined as a lower bound set with respect to* R_n *if and only if* $Z_{(\boldsymbol{P_n^{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P_{TSGA}^s})}$ *where* $Z_{(\boldsymbol{P_{TSGA}^s})} = \{\mathbf{z}(\mathbf{x}) : z_k(\mathbf{x}) \geq lb_k^s, k = 1, \ldots, m, \mathbf{x} \in X\}$.

We now describe the initialization and update procedure of a lower bound set LB^n in each iteration of TSGA. An illustration of the lower bounds and search zones generated by TSGA can be found in Figure 5.

Since TSGA starts with an empty set, $R_1 = \emptyset$, the initial lower bound set at $n = 1$ consists of only a global lower bound vector, $\mathbf{lb}^1 = \{M_1, \ldots, M_m\}$, $LB^1 = (\mathbf{lb}^1)$. That is, the associated search zone corresponds to the original feasible set, i.e., $Z_{(\boldsymbol{P_1^{TSGA}})} = Z_{(\boldsymbol{P_{TSGA}^1})} = Z$ which is consistent with Definition 8. At the end of any iteration, n, TSGA generates the lower bound set for the next iteration, LB^{n+1}, using the existing lower bound set, LB^n, and the

newly generated representative nondominated point, $\mathbf{y}^n$. Since $Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} \setminus \{\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} : y_k^n + \alpha > z_k \ \forall k\}$, the existing lower bound set is updated to remove the regions that are strictly α-dominated point by $\mathbf{y}^n$ from the search zones. To do this, TSGA first identifies the set $\mathcal{L}^n = \{\mathbf{lb}^s \in LB^n : y_k^n + \alpha \geq lb_k^s \ \forall k\}$ including the lower bound vectors whose associated search zones contain strictly α-dominated regions. Then, TSGA finds LB^{n+1} by modifying only the lower bound vectors in $\mathcal{L}^n \in LB^n$, while keeping the remaining lower bound vectors in $LB^n \setminus \mathcal{L}^n$ as they are. We next formalize this procedure in Proposition 2.

Proposition 2. *Let LB^n be a lower bound set with respect to R_n, $\mathbf{y}^n$ be the optimal solution of $(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})$ and $\mathcal{L}^n = \{\mathbf{lb}^s \in LB^n : \hat{y}_k^n \geq lb_k^s \ \forall k\}$ where $\hat{y}_k^n = y_k^n + \alpha \ \forall k$. Then, LB^{n+1} is a lower bond set with respect to $R_{n+1} = R_n \cup \{\mathbf{y}^n\}$ if the following two properties are satisfied:*

(i) For any $\mathbf{lb}^s \in \mathcal{L}^n$, LB^{n+1} includes m new lower bound vectors $\mathbf{lb}^{s,k} = \left\{lb_1, \ldots, \hat{y}_k^n, \ldots, lb_m\right\}$ $k = 1, \ldots, m$.

(ii) Any $\mathbf{lb}^s \in LB^n \setminus \mathcal{L}^n$ is also included in LB^{n+1}.

Proof: **Part 1.** $Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} \setminus Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \{\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} : y_k^n + \alpha > z_k \ \forall k\}$.
Consider a point $\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})}$ but $\mathbf{z} \notin Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})}$. Then, by definition of the corresponding solution sets, $\mathbf{z}$ does not satisfy the following constraint set:

$$\begin{aligned} &z_k(\mathbf{x}) \geq (y_k^n + \alpha)t_{kn} + M_k(1 - t_{kn}) && k = 1, \ldots, m, \\ &\sum_{k=1}^{m} t_{kn} \geq 1 \quad t_{kn} \in \{0, 1\} && k = 1, \ldots, m, \end{aligned}$$

This implies, $t_{kn} = 0$ and $z_k < y_k^n + \alpha$ for all $k = 1, \ldots, m$.
Part 2. $Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^{n+1}} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$ if LB^{n+1} satisfies (i) and (ii).
Since LB^n is a lower bound set with respect to R_n, $Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$ holds by Definition 8. Using it together with part 1, we obtain:

$$Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} (Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \setminus \{\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} : y_k^n + \alpha > z_k \ \forall k\})$$

Let $Z_\alpha^s = \{\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} : y_k^n + \alpha > z_k \ \forall k\}$. Then, we have:

$$Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} (Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \setminus Z_\alpha^s)$$

If $\mathbf{lb}^s \in LB^n \setminus \mathcal{L}^n$, $lb_k^s > y_k^n + \alpha$ holds for at least one criterion, k, by definition. Then, for any point $\mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$ (i.e. $z_k \geq lb_k^s$ for all $k = 1, \ldots, m$), there exists some k for which

$z_k \geq lb_k^s > y_k^n + \alpha$. Hence $\mathbf{z} \notin Z_\alpha^s$, implying $Z_\alpha^s = \emptyset$ if $\mathbf{lb}^s \in LB^n \setminus \mathcal{L}^n$.

If $\mathbf{lb}^s \in \mathcal{L}^n$, $y_k^n + \alpha \geq lb_k^s$ holds for all $k = 1, \ldots, m$ by definition. Then, $(Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \setminus Z_\alpha^s)$ includes the set of feasible points satisfying $z_k \geq lb_k^s$ for all $k = 1, \ldots, m$ and $z_k \geq y_k^n + \alpha$ for at least one k. Then, for any point $\mathbf{z} \in (Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \setminus Z_\alpha^s)$, at least one of the following m new lower bound vectors hold, $\mathbf{lb}^{s,k} = \left\{lb_1, \ldots, \hat{y}_k^n, \ldots, lb_m\right\}$, $k = 1, \ldots, m$. This implies:

$$Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} (Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \setminus Z_\alpha^s) = \left[\bigcup_{\mathbf{lb}^s \in \mathcal{L}^n} \bigcup_{k=1}^{m} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s,k}})}\right] \bigcup \left[\bigcup_{\mathbf{lb}^s \in LB^n \setminus \mathcal{L}^n} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}\right]$$

If LB^{n+1} satisfies conditions in (i) and (ii), the equation above could be rewritten as $Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^{n+1}} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$ since conditions (i) and (ii) describe the lower bound vectors in the first and second parts of the term above, respectively. Then, by definition 8, LB^{n+1} is a lower bound set with respect to R_{n+1}. □

Using Proposition 2, TSGA finds a lower bound set LB^{n+1} with respect to R_{n+1} at the end of iteration n that satisfies $Z_{(\boldsymbol{P}_{\boldsymbol{n+1}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^{n+1}} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$. However, Dächert et al. (2017) show that LB^{n+1} may include *redundant* lower bound vectors that induce search zones included in a search zone associated to some other lower bound vector in LB^n. To identify and avoid these search zones, TSGA utilizes their "redundancy elimination" approach that constructs a neighborhood relationship structure between the existing lower bound vectors and uses this structure to detect redundant lower bound vectors. For further details of this neighborhood structure, we refer the reader to Dächert et al. (2017).

We next present the pseudocode for TSGA. In Steps 1-5, TSGA generates a new representative point, i.e., the optimal solution of $(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})$, by solving the submodels $(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})$ associated with the lower bound set LB^n. If $Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \neq \emptyset$, we denote the nondominated point corresponding to the optimal solution to $(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})$ as $\mathbf{z}^{lb^s}$ and show that the optimal solution to $(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})$ can be found by evaluating $\mathbf{z}^{lb^s}$, $\mathbf{lb}^s \in LB^n$, in Proposition 3. Note that we do not solve model $(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})$ if we detect that the solution will be identical to that of the models solved so far.

Proposition 3. *Let* $(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})$ *be feasible and* LB^n *be such that* $Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$. *Then, the optimal point of* $(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})$ *in the objective space is* $\mathbf{y}^n = \arg\max_{\mathbf{z}^{lb^s}} \left\{\sum_{k=1}^{m} \lambda_k z_k^{lb^s}\right\}$.

Proof: Since $Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})}$, $\max\left\{\sum_{k=1}^{m} \lambda_k z_k : \mathbf{z} \in Z_{(\boldsymbol{P}_{\boldsymbol{n}}^{\boldsymbol{TSGA}})}\right\} = \max\left\{\sum_{k=1}^{m} \lambda_k z_k^{lb^s} : \mathbf{lb}^s \in LB^n,\ Z_{(\boldsymbol{P}_{\boldsymbol{TSGA}}^{\boldsymbol{s}})} \neq \emptyset\right\}$. Then $\mathbf{y}^n = \arg\max_{\mathbf{z}^{lb^s}} \left\{\sum_{k=1}^{m} \lambda_k z_k^{lb^s}\right\}$. □

TSGA terminates when the feasible set becomes empty. We show in Proposition 4 that this is equivalent to $Z_{(\boldsymbol{P^s_{TSGA}})} = \emptyset$ for all $\mathbf{lb}^s \in LB^n$.

Proposition 4. *Let* LB^n *be such that* $Z_{(\boldsymbol{P^{TSGA}_n})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P^s_{TSGA}})}$. $\boldsymbol{(P^{TSGA}_n)}$ *is infeasible if and only if* $\boldsymbol{(P^s_{TSGA})}$ *is infeasible for all* $\mathbf{lb}^s \in LB^n$.

Proof: Since $Z_{(\boldsymbol{P^{TSGA}_n})} = \bigcup_{\mathbf{lb}^s \in LB^n} Z_{(\boldsymbol{P^s_{TSGA}})}$, $Z_{(\boldsymbol{P^{TSGA}_n})} = \emptyset$ if and only if $Z_{(\boldsymbol{P^s_{TSGA}})} = \emptyset$ for all $\mathbf{lb}^s \in LB^n$. □

TSGA: Territory-excluded Supported Generating Algorithm

Input: $\alpha \in [0,1]$ and $\lambda_k \in \mathbb{R}_{>0}$

Output: Representative subset R

Initialize: $R_1 = \emptyset$; $n = 1$; $LB^1 = (\{M_1, \ldots, M_m\})$

1: **for** $\mathbf{lb}^s \in LB^n$ **do**

2: find $\mathbf{z}^{lb^s}$ solving $\boldsymbol{(P^s_{TSGA})}$;

3: **if** there exists any feasible solution **then**

4: $\mathbf{y}^n = \arg\max_{\mathbf{z}^{lb^s}} \left\{ \sum_k \lambda_k z_k^{lb^s} \right\}$

5: $R_{n+1} = R_n \cup \{\mathbf{y}^n\}$; generate LB^{n+1}

6: $n \leftarrow n+1$ go to Step 1.

7: **else** stop. $R = R_n$

TSGA keeps generating new representative points by solving $\boldsymbol{(P^{TSGA}_n)}$ at each iteration, n, utilizing the decomposition and search procedure adapted from Dächert et al. (2017) as described above in detail. TSGA is robust in the sense that any procedure that solves $\boldsymbol{(P^{TSGA}_n)}$ can be employed. The number of zones searched (equivalently, models solved) at each iteration, n, is bounded from above by $|LB^n|$ for which Klamroth et al. (2015) provide an upper bound of $\mathcal{O}(|R_n|^{\lfloor \frac{m}{2} \rfloor})$ using the work of Kaplan et al. (2008) for an arbirary $m \geq 2$. They state that the actual value in practice tends to be much smaller. Furthermore, Klamroth et al. (2015) and Dächert et al. (2017) show that the efficiency of the algorithms could be substantially improved utilizing efficient data structures in keeping and updating the lower bound sets. Our empirical results are also in line with these observations where we end up with solving only a fraction of models of $|LB^n|$.

3.2. The Weight Selection Procedure (WSP)

The algorithms of Masin and Bukchin (2008), Sylva and Crema (2007) and Ceyhan et al. (2019) choose the worst-represented point by the current representative set as the next representative point. In this sense, they can be considered as myopic greedy algorithms. While they

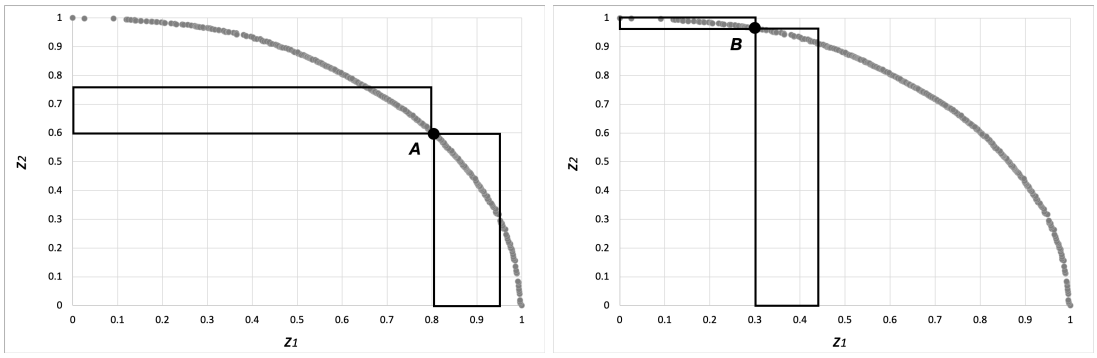
make the maximum improvement in the coverage gap value at each iteration, given the current representative set, they overlook the missed opportunities for future selections due to the current selection. In contrast, TSGA searches for a nondominated point from dense regions of the nondominated set, while keeping a distance of at least α from existing representative points, with the aim of achieving a small final representative set that satisfies quality requirements. To this end, TSGA solves a weighted-sum problem at each iteration, choosing weights based on a problem-specific procedure that guides the search towards points with greater "individual representation power" values. Given a prespecified coverage gap value, we measure "individual representation power" of a nondominated point by the percentage of nondominated points that lie in its territory.

Definition 9. *Given a desired coverage gap value of α, the individual representation power of $\mathbf{y} \in \mathbf{Z}_{ND}$ is defined as* $IRP(\mathbf{y}) = \dfrac{|\mathbf{Z}_{ND} \cap T_{\mathbf{y}}|}{|\mathbf{Z}_{ND}|}$.

IRP is a key measure for TSGA as it quantifies the representation power of each nondominated point. Since the nondominated set is not available at the outset, we first predict the regions of the nondominated points with greater IRP values and then estimate weights of the criteria to guide the search towards these regions. Köksalan (1999), Karasakal and Köksalan (2009), and Köksalan and Lokman (2009), through extensive experimentation, show that the nondominated sets for multi-objective optimization problems could be approximated by a hypersurface that is typically convex for a minimization problem and concave for a maximization problem. The shape becomes more representative as the problem size grows and the nondominated set grows. Therefore, the quality of approximation improves for large-sized problems for which generating the whole nondominated set becomes unpractical and prohibitive, and generating a representative set is more meaningful. Özarik et al. (2020) study the distribution characteristics of the nondominated sets of MOIPs extensively. They find a center of density for each problem and show that the central parts of a nondominated frontier of a minimization (maximization) problem are typically denser due to approximate convexity (concavity) of the frontier.

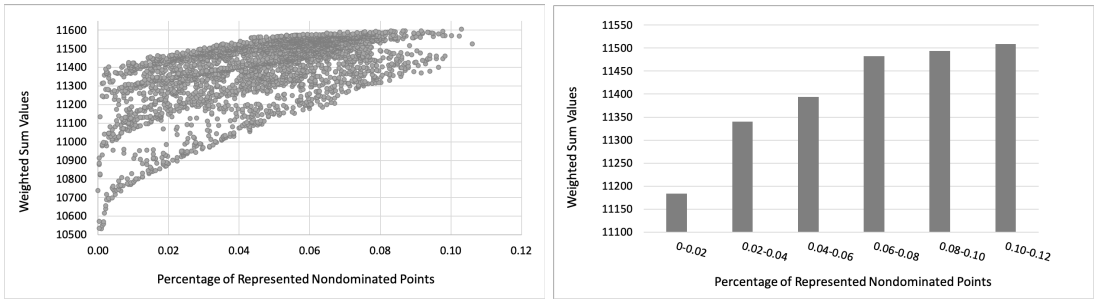
Based on these observations and findings, we develop a new weight selection procedure (WSP) that estimates the dense regions of a nondominated set and determines weight values that would guide the search process towards these regions. Since the IRP values of the nondominated points in dense regions will be relatively larger, as shown in Figure 2, this allows TSGA to generate "more powerful" representative points and reduces the size of the final representative set while maintaning the desired coverage level. Figure 3 illustrates on an example problem that, based

on the weights found with our WSP, the larger the weighted-sum objective function value of a nondominated point is, the larger is its IRP value.



(a) Large IRP value in the dense region (b) Small IRP value in the sparse region

Figure 2: Bi-objective knapsack problem instance, $|Z_{ND}| = 431$, $\alpha = 0.15$. $T_{\mathbf{B}}$ includes fewer number of nondominated points than $T_{\mathbf{A}}$, i.e. $IRP(\mathbf{A}) = 0.49 > IRP(\mathbf{B}) = 0.16$.



(a) Weighted sum values vs. IRP values (b) Average weighted sum values vs. IRP values

Figure 3: The relation between weighted sum values and individual representation power values of nondominated points of a 3-objective knapsack problem, $|Z_{ND}| = 3253$, $\alpha = 0.5$, $\boldsymbol{\lambda} = (0.315, 0.313, 0.372)$(found by WSP).

WSP first approximates the locations of nondominated points by using the procedure of Köksalan and Lokman (2009) and fitting L_q function of the form:

$$L_q = \Big\{ \mathbf{z} \in \mathbb{R}^m_{\geq 0} : \Big[\sum_{k=1}^{m} (z_k)^q \Big]^{1/q} = 1 \quad q > 0 \Big\}$$

where q value is calculated based on a "central reference point", $\mathbf{z}^C$, that is at minimum Tchebycheff distance from the ideal point. WSP then finds the weight vector $\boldsymbol{\lambda}$ using the hyperplane tangent to the fitted L_q function at point $\mathbf{z}^C$. We next present the steps of WSP and illustrate it on an example problem in Figure 4.

WSP: Weight Selection Procedure

Input: $\mathbf{z}^{IP}$

Output: $\boldsymbol{\lambda}$

1: Find $\mathbf{z}^C$ by minimizing the Tchebycheff distance from the ideal point $\mathbf{z}^C = \arg\min_{\mathbf{z}\in Z}\left\{\max_{k=1,\ldots,m}(z_k^{IP} - z_k)\right\}$

2: Calculate the q value of the L_q function passing through $\mathbf{z}^C$ by solving $\left[\sum_{k=1}^{m}(z_k^C)^q\right]^{1/q} = 1$

3: Find the hyperplane tangent to the fitted L_q function at point $\mathbf{z}^C$ and set $\lambda_k = (z_k^C)^{q-1} \times \left(1 - \sum_{t=1}^{m-1}(z_t^C)^q\right)^{\frac{1}{q}-1}$ $k = 1, \ldots, m-1$ and $\lambda_m = 1$.

4: Normalize the $\boldsymbol{\lambda}$ values to set their sum to 1, $\lambda_k \leftarrow \frac{\lambda_k}{\sum_{k=1}^{m}\lambda_k}$ $k = 1, \ldots, m-1$.

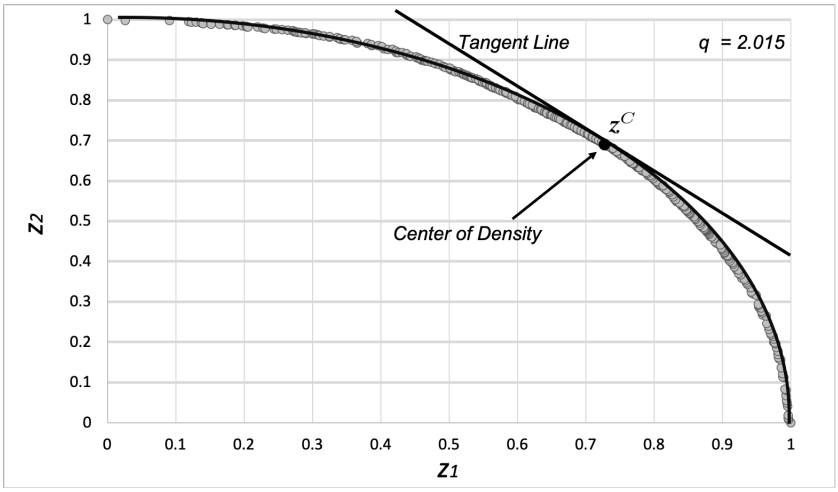


Figure 4: The fitted L_q surface and illustration of finding the weights, $\boldsymbol{\lambda}$, for a bi-objective knapsack problem instance, $|Z_{ND}| = 431$, $\mathbf{z}^C = (0.739, 0.681)$, $\boldsymbol{\lambda} = (0.514, 0.486)$.

WSP is designed as an efficient mechanism to estimate a weight vector that would guide the search process in TSGA for any problem instance. We demonstrate the effects of our weight generation procedure (WSP) against using equal weights with some computational results in Section 5.

As the algorithm proceeds and new nondominated points are found, it is possible to improve the fitted L_q function and update the weight vector. This could potentially somewhat improve the quality of the approximation. However, Köksalan and Lokman (2009) show with extensive experiments on a variety of MOIPs that the performance of the approximation is not very sensitive to the number of reference points used for estimating the q value. As an extreme case, they fit a L_q function using all nondominated points and they show that the quality of

approximation improves only slightly even in this extreme case. Therefore, in our experiments we fit the L_q function using a single central point, $\mathbf{z}^C$. In our experiments, we also find the minimum possible number of representative points for some of the problems. We then compare the performance of TSGA against the corresponding cases where the desired coverage gap level is achieved with the minimimum number of representative points. Our results show that the representative set size of TSGA deviates very little, from the optimal size, leaving little room for further improvement.

4. Illustrative Example

In this section, we illustrate TSGA and DMA on an example problem. As DMA, SC, SBA, and TDA are practically the same in terms of the representative sets they generate, it is sufficient to demonstrate one of them.

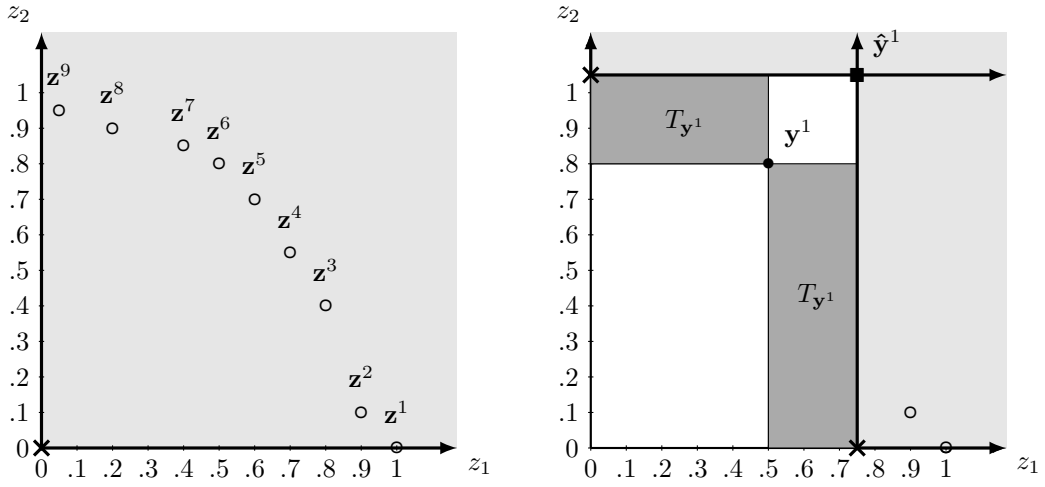
Table 1: The Nondominated Points of Illustrative Example

$\mathbf{z}^j$	1	2	3	4	5	6	7	8	9
z_1^j	1.00	0.90	0.80	0.70	0.60	0.50	0.40	0.20	0.05
z_2^j	0.00	0.10	0.40	0.55	0.70	0.80	0.85	0.90	0.95

Let $\alpha = 0.25$ for the bi-objective knapsack problem instance that has the nondominated points given in Table 1 and shown in Figure 5a. The points have been scaled in the objective space, i.e., $z_k(\mathbf{x}) \in [0,1],\ k = 1,\ldots,m,\ \forall \mathbf{x} \in X$. We normally do not know the nondominated set in applying the algorithms and present them here just for demonstration purposes.

Let R_{TSGA} and R_{DMA} be the representative sets generated by TSGA and DMA, respectively. As described above, we first calculate the objective weights λ_k to be used in $(\boldsymbol{P^s_{TSGA}})$, favoring the dense regions of the nondominated frontier of the given problem instance. The central nondominated point is that minimizes the Tchebycheff distance from $\mathbf{z}^{IP} = (1,1)$ is $\mathbf{z}^C = \mathbf{z}^5 = (0.60, 0.70)$. We next fit the L_q surface solving equation $L_q(\mathbf{z}^C) = 0.60^q + 0.70^q = 1$ and obtain $q = 1.614$. The equation of the tangent line to $L_{1.614}$ curve at $\mathbf{z}^C$ is $0.910z_1 + 1.000z_2 = 1.246$. Normalizing the coefficients in this equation, we obtain $\lambda_1 = 0.477$ and $\lambda_2 = 0.524$. TSGA solves $\max_{\mathbf{x}\in X} f(\mathbf{x}) = 0.477z_1(\mathbf{x}) + 0.524z_2(\mathbf{x})$ and finds $\mathbf{y}^1 = \mathbf{z}^6 = (0.50, 0.80)$. The territory of $\mathbf{y}^1$, $T_{\mathbf{y}^1}$, is shown in Figure 5b. We keep maximizing $f(\mathbf{x})$ excluding the territories of current representative points until there are no new solutions in the reduced space as shown in Figure 5c. The resulting representative subset is $R_{TSGA} = \{\mathbf{z}^6, \mathbf{z}^3\}$ satisfying $\alpha_R \leq \alpha = 0.25$.

Suppose DMA also starts with the initial point $\mathbf{y}^1 = \mathbf{z}^6$. It next finds the worst represented nondominated point, $\mathbf{y}^2 = \mathbf{z}^1 = (0,1)$ that has $\alpha(\mathbf{z}^1) = 0.50$. Given these two representative



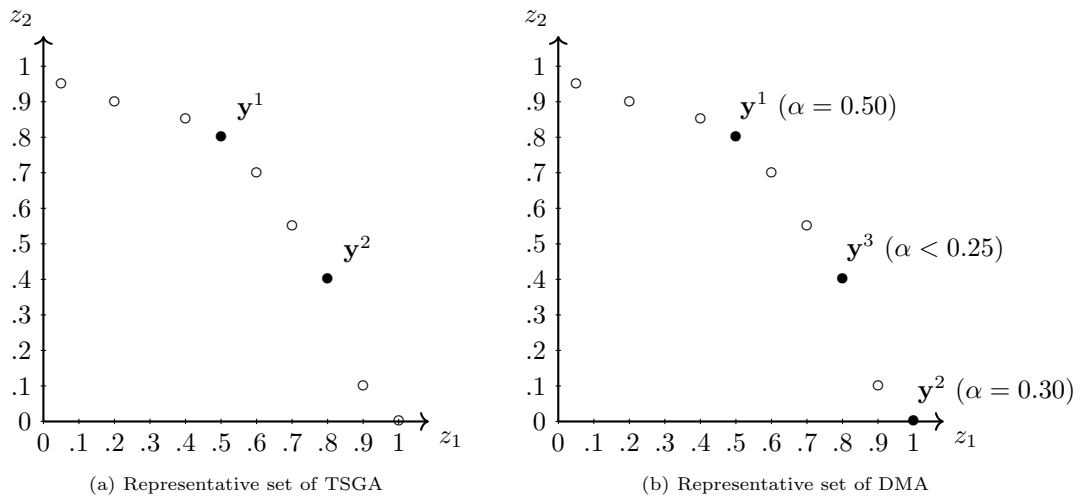
(a) The set of nondominated points

(b) Search zones and territories at $n = 1$

(c) Search zones and territories at $n = 2$

●: representative point, ■: shifted representative point, $\mathbf{x}$: lower bound

Figure 5: The graphical display of the constructed territories and lower bounds by TSGA for a bi-objective knapsack problem. Dark gray regions represent territories while light gray regions represent search zones.



(a) Representative set of TSGA

(b) Representative set of DMA

∘ nondominated points • representative nondominated points

Figure 6: Graphical demonstrations of the representative sets generated by TSGA and DMA for a bi-objective knapsack problem.

points, the next worst represented point is $\mathbf{y}^3 = \mathbf{z}^3$ that has $\alpha(\mathbf{z}^3) = 0.30$. The algorithm terminates at this point since the coverage gap of the remaining nondominated points are within the desired level $\alpha = 0.25$. The generated representative set is $R_{DMA} = \{\mathbf{z}^6, \mathbf{z}^3, \mathbf{z}^1\}$.

The final representative sets generated by TSGA and DMA are shown in Figure 6. As illustrated in this example, while DMA (as well as SC, SBA, and TDA) finds the farthest (in terms of the coverage measure) point from the previously generated nondominated points at each iteration, TSGA generates nondominated points that are located in the dense regions outside the restricted territories of previously generated points. TSGA satisfies the desired coverage gap $\alpha = 0.25$ by generating only two representative points, while DMA generates three representative points for this small example problem. We show in the next section how TSGA outperforms the other algorithms in terms of all performance measures by extensive computational experiments on larger size instances.

5. Computational Results

In this section, we compare TSGA with the existing approaches, DMA, SBA and TDA. Recall that DMA and SC are essentially the same, except for some minor differences in presentation. We conduct computational experiments on randomly generated test instances of the Multi-Objective Knapsack Problem (MOKP), the Multi-Objective Assignment Problem (MOAP), and the multi-objective Mixed-Integer Knapsack Problem (MIKP) with 3, 4, and 5 objectives.

To demonstrate the performance of our approach on continuous decision variables, we conduct additional experiments on Multi-objective Linear Knapsack Problem (MLKP), where fractional parts of items can be placed in the knapsack.

We follow the random generation scheme of Köksalan and Lokman (2015) to create the instances of knapsack problems (KPs) with three objectives and assignment problems (APs) with three, four, and five objectives. In KPs with three objectives, the objective function coefficients and weight coefficients of the items are generated randomly from the discrete uniform distribution in the interval $[10, 100]$ and the capacity of the knapsack is set to half of the total weight of all items. In KPs with four and five objectives, we use the test instances of Kırlık and Sayın (2014). In APs, we generate the assignment costs from a discrete uniform distribution in the interval $[1, 20]$. In order to generate all nondominated points of these problems, we used TSGA, setting $\alpha = 0$ to force territory sizes to zero for each objective. We also solve multi-objective mixed-integer programs. We conducted computational experiments with mixed-integer knapsack test instances with three (3MIKP100), four (4MIKP40) and five objectives (5MIKP20) using the generation scheme of Ceyhan et al. (2019). Multi-objective MIKP is similar to MOKP except that, rather than having all variables binary, half of the variables are restricted to be binary and the other half continuous in the interval $[0, 1]$.

Although SBA and TDA are originally designed to use the decomposition and search procedure of Lokman and Köksalan (2013), we replace it with the more efficient procedure of Dächert et al. (2017) in SBA and TDA as well, in order to utilize the same efficient mechanism to generate the nondominated points when comparing to TSGA. This allows us to directly measure the effects of the mechanisms for representative selection in comparing the computational efficiencies of different methods, isolating confounding factors.

In our experiments with MOKP, MOAP, and MIKP, we work on the scaled objective values and generate representative sets for different coverage gap values, $\alpha = 0.05$, 0.10, 0.15, 0.20 and 0.25. In order to scale the objectives, we use the true ideal and nadir values in all objectives for MOKP and MOAP test instances. However, in the experiments on MIKPs, since the true nadir point is not known, we use the payoff nadir point. For each problem instance, we also attempt to find the smallest possible representative set (the optimal representative set) for a given coverage gap value. When we have all nondominated points available explicitly, we can solve $\boldsymbol{(P^*_\alpha)}$ to determine the minimal possible representative set for a given α value so long as the problem size is manageable. In the summary tables that we present below for our numerical experiments, we indicate the problem types for which the minimal possible representative sets cannot be generated within a reasonable solution time. This is mainly related with the total

number of nondominated points, N. In our experiments, finding the optimal solution to $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ within 4 hours becomes problematic beyond $N = 500$.

$(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$:

$$\begin{aligned}
& \text{Min} && \sum_{j=1}^{N} r_j && \\
& \text{s.to} && (z^i_k - z^j_k)\beta_{ij} \leq \alpha && \forall i, \forall j,\ k = 1, ..., m \\
& && \sum_{i=1}^{N} \beta_{ij} \leq N r_j && \forall j \\
& && \sum_{j=1}^{N} \beta_{ij} = 1 && \forall i \\
& && r_j, \beta_{ij} \in \{0, 1\} && i, j = 1, ..., N
\end{aligned}$$

In $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$, N denotes the size of the nondominated set, $N = |Z_{ND}|$. $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ determines which nondominated points are covered by which representative points. The binary variable r_j takes the value of 1 when $\mathbf{z}^j$ is selected as a representative point. The binary variable β_{ij} takes the value of 1 when $\mathbf{z}^j$ is selected as a representative point, i.e. $r_j = 1$, and $\mathbf{z}^i$ is represented by $\mathbf{z}^j$. Note that a nondominated point $\mathbf{z}^i$ is prohibited from being represented by another nondominated point $\mathbf{z}^j$ that has not been selected as a representative point, i.e. $r_j = 0$. The model finds the minimum number of representative points that guarantee a coverage value of α.

We implemented TSGA and the existing approaches (DMA, SBA, TDA) in C programming language using Microsoft Visual Studio 2017 Professional. We run the algorithms on computers with Intel(R)Core(TM)i7-477OS CPU @3.10 GHz, 16 GB RAM and Windows 10. We generate smallest representative subsets by solving $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ with a commercial software (GAMS 23.9.5). We use the callable library of IBM ILOG CPLEX 12.5 to solve all the mathematical models.

5.1. Results and Analyses

We present the results in Tables 3 - 10. We report the size of the representative set and the solution time for each problem instance. Since we could generate all nondominated points for the MOAPs and MOKPs, we attempted to solve $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ to find the optimal representative subsets for these problems for comparison purposes. We report the optimal values as well for the cases we could solve $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ within 4 hours of CPU time. DMA, SBA, and TDA use the same logic and generate the same number of representative points, except for very small occasional differences due to selecting different points when there are ties. Therefore, we present only the number of representative points produced by TDA to represent all three methods. However, we provide the computational times of the algorithms separately since they differ substantially.

Since the computational complexity of DMA grows by progressively adding binary variables to the models solved at each iteration, it is sensitive to problem size, and even becomes prohibitive for some large-sized problems.

We observe that the size of the representative set and the computational times tend to increase with decreasing α and increasing problem size for all methods. Hence, as α gets closer to zero, the task of generating a representative set becomes prohibitive, especially as problem size grows. For $\alpha = 0$, all nondominated points must be generated, which is computationally difficult even for few criteria. In order to keep the number of representative points at a reasonable level, we set the α values to be at least 0.05, as is commonly done in the literature. However, to demonstrate the effect of a small α value, we conduct an additional experiment by setting $\alpha = 0.01$ for small sized (20 and 40 items) 3-objective knapsack problems. We report the corresponding total number of nondominated points and the sizes of the representative sets generated by TSGA ($|R_{TSGA}|$) for $\alpha = 0.01$ and $\alpha = 0.05$ in Table 2. These results show that the number of generated representative points is a large percentage of the total number of nondominated points when $\alpha = 0.01$ and is not consistent with the original purpose of generating representative points to alleviate the computational burden.

Table 2: Averages (Avg) and Standard Deviations (StDev) of Number of Nondominated Points (N) and Representative Points $|R_{TSGA}|$

			$\lvert R_{TSGA}\rvert$			
	N		$\alpha = 0.01$		$\alpha = 0.05$	
Problem	Avg	StDev	Avg	StDev	Avg	StDev
3MOKP20	37.40	26.95	29.50	18.46	14.40	6.28
3MOKP40	229.20	101.76	121.30	50.84	26.70	8.67

Our results, overall, show that the average sizes of the representative sets generated by TSGA is smaller than that of DMA, SBA and TDA. When the problem sizes are small, TSGA obtains almost as small a representative set as the optimal set. For larger problem sizes, TSGA still obtains results close to the optimal set and substantially outperforms the other methods. TSGA substantially outperforms the other methods in terms of the solution times as well. The performances of the methods are more critical as the problem difficulty increases (i.e., for smaller α and larger problem sizes). TSGA's superiority to other methods becomes even more pronounced as the difficulty level of the problem increases.

We summarize the results in the two plots of Figures 7 and 8. Recall that DMA, SBA, and TDA are essentially identical in terms of the representative sets they generate. Therefore, we present their average results by a single line (marked as D-S-T) in Figure 7. The figure shows

that the average size of the representative set of TSGA is only a fraction of that of D-S-T. Specifically, from Table 3, when we take the ratio of the number of representatives required for the algorithms for each coverage value and average them out over different coverage values for 3MOKP100, we see that TSGA requires only 43% of the number of representatives of D-S-T, on average. In a similar manner, we calculate from Table 3 for the same problem that average solution times of TSGA are 19%, 41%, and 44% of those of DMA, SBA and TDA, respectively. Figure 8 shows the average solution times of each algorithm for different sizes of 3-objective knapsack problems and reveals that TSGA outperforms all of them by far. Tables 4 to 9 show similar results in all sizes of all problem types.

Table 3: Performance Results on MOKP with three objectives*

			$\|R\|$			Solution Time (secs)			
Problem	N	α	D-S-T	TSGA	Optimal**	DMA	SBA	TDA	TSGA
3MOKP10	9.30	0.05	7.80	7.20	7.20	0.39	0.43	0.44	0.42
		0.10	6.50	5.70	5.50	0.30	0.34	0.32	0.32
		0.15	5.30	4.50	4.20	0.24	0.22	0.25	0.25
		0.20	4.30	3.50	3.40	0.22	0.17	0.19	0.18
		0.25	3.90	3.00	3.00	0.20	0.15	0.17	0.15
3MOKP20	37.40	0.05	17.10	14.40	14.30	1.37	1.44	1.42	1.20
		0.10	10.70	8.00	7.00	0.67	0.74	0.74	0.49
		0.15	7.90	5.30	4.60	0.48	0.55	0.47	0.31
		0.20	5.70	3.60	3.20	0.34	0.36	0.32	0.20
		0.25	4.20	2.80	2.50	0.25	0.25	0.22	0.15
3MOKP30	137.90	0.05	40.70	27.70	25.30	16.73	3.45	4.01	2.54
		0.10	19.40	12.20	9.80	1.98	1.48	1.41	0.86
		0.15	11.90	6.80	5.40	0.90	0.85	0.73	0.45
		0.20	8.60	4.90	3.50	0.58	0.61	0.51	0.28
		0.25	5.60	3.30	2.50	0.35	0.36	0.29	0.19
3MOKP40	229.20	0.05	41.70	26.70	22.30	29.47	3.97	4.43	2.37
		0.10	20.40	10.10	7.50	3.17	1.69	1.60	0.73
		0.15	12.50	5.70	4.50	1.07	0.97	0.85	0.37
		0.20	8.30	3.90	3.00	0.58	0.61	0.52	0.23
		0.25	4.90	3.00	2.30	0.32	0.34	0.29	0.16
3MOKP50	520.90	0.05	53.80	31.30	20.00 (4)	89.75	5.61	5.92	3.37
		0.10	22.80	11.30	6.75 (4)	4.70	2.18	1.95	0.92
		0.15	13.60	6.20	3.25 (4)	1.45	1.15	0.92	0.43
		0.20	9.60	4.10	2.50 (4)	0.77	0.72	0.65	0.29
		0.25	6.00	3.00	2.00 (4)	0.46	0.48	0.38	0.20
3MOKP100	3768.20	0.05	82.30	45.30	-	1029.92	10.83	11.82	6.37
		0.10	31.80	13.30	-	11.74	3.64	3.38	1.35
		0.15	20.00	6.80	-	3.69	1.99	1.76	0.55
		0.20	13.10	4.30	-	1.62	1.23	1.04	0.35
		0.25	7.90	3.90	-	0.77	0.73	0.59	0.37

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, and α is the coverage gap value.

** Some of the replications of $(\boldsymbol{P}^*_\alpha)$ could not be solved to optimality within 4 hours. For those problems, the values in parentheses show the numbers of replications that could be solved to optimality, where the averages are taken over those problems only. For brevity, no parentheses are used when all 10 replications could be solved to optimality and the cell is left blank when none of the 10 replications could be solved to optimality.

Table 4: Performance Results on MOKP with four objectives*

			$\|R\|$			Solution Time (secs)			
Problem	N	α	D-S-T	TSGA	Optimal**	DMA***	SBA	TDA	TSGA
4MOKP10	11.60	0.05	8.40	7.90	7.80	0.45	0.78	0.67	0.73
		0.10	6.50	5.70	5.50	0.32	0.52	0.46	0.52
		0.15	5.40	4.50	4.20	0.25	0.35	0.35	0.41
		0.20	4.00	3.40	3.10	0.19	0.21	0.24	0.34
		0.25	3.00	2.90	2.40	0.15	0.14	0.16	0.30
4MOKP20	136.80	0.05	43.60	32.80	31.80	22.49	8.87	10.12	6.75
		0.10	22.80	13.00	11.90	2.33	3.43	3.82	1.59
		0.15	14.40	7.60	6.40	0.97	1.68	1.82	0.77
		0.20	8.40	5.00	4.00	0.51	0.88	0.84	0.40
		0.25	5.30	3.30	2.80	0.33	0.51	0.44	0.24
4MOKP30	397.60	0.05	82.60	54.10	48.00	375.46	18.65	25.73	13.40
		0.10	34.80	17.60	13.60	11.38	6.78	6.77	2.53
		0.15	19.00	9.00	6.38 (8)	2.15	18.26	2.73	0.85
		0.20	11.50	5.60	3.20 (5)	0.87	34.44	1.28	0.44
		0.25	7.00	3.80	2.40 (5)	0.47	0.77	0.67	0.28
4MOKP40	1808.60	0.05	128.50	76.30	39.50 (4)	1987.14	40.98	51.55	21.20
		0.10	44.90	20.50	10.00 (3)	90.30	10.25	10.68	3.13
		0.15	22.90	10.40	-	6.40	3.77	3.45	1.22
		0.20	13.50	6.00	-	1.61	1.77	1.64	0.52
		0.25	7.90	3.70	-	0.70	44.80	0.74	0.31

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, and α is the coverage gap value.

** Some of the replications of $(\boldsymbol{P}^{\star}_{\boldsymbol{\alpha}})$ could not be solved to optimality within 4 hours. For those problems, the values in parentheses show the numbers of replications that could be solved to optimality, where the averages are taken over those problems only. For brevity, no parentheses are used when all 10 replications could be solved to optimality and the cell is left blank when none of the 10 replications could be solved to optimality.

*** Fewer than 10 replications are reported per cell for 4MOKP40 instances since DMA could not be finalized within 12 hours.

Table 5: Performance Results on MOKP with five objectives*

			$\|R\|$			Solution Time (secs)			
Problem	N	α	D-S-T	TSGA	Optimal	DMA	SBA	TDA	TSGA
5MOKP10	16.20	0.05	14.10	13.70	13.70	0.87	2.92	2.49	2.42
		0.10	10.70	9.30	9.30	0.51	1.65	1.51	1.33
		0.15	8.40	7.20	6.90	0.39	1.05	0.96	0.96
		0.20	6.60	5.70	5.20	0.32	0.76	0.68	0.61
		0.25	4.40	4.70	3.90	0.21	0.38	0.34	0.44
5MOKP20	161.20	0.05	54.40	46.60	44.40	45.74	31.81	37.56	28.68
		0.10	23.80	18.60	14.30	4.30	9.13	9.25	5.73
		0.15	12.90	9.40	7.00	1.23	3.46	2.69	1.51
		0.20	8.00	4.90	3.80	0.55	1.48	1.24	0.65
		0.25	6.30	3.60	2.40	0.38	0.87	0.80	0.41

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, and α is the coverage gap value.

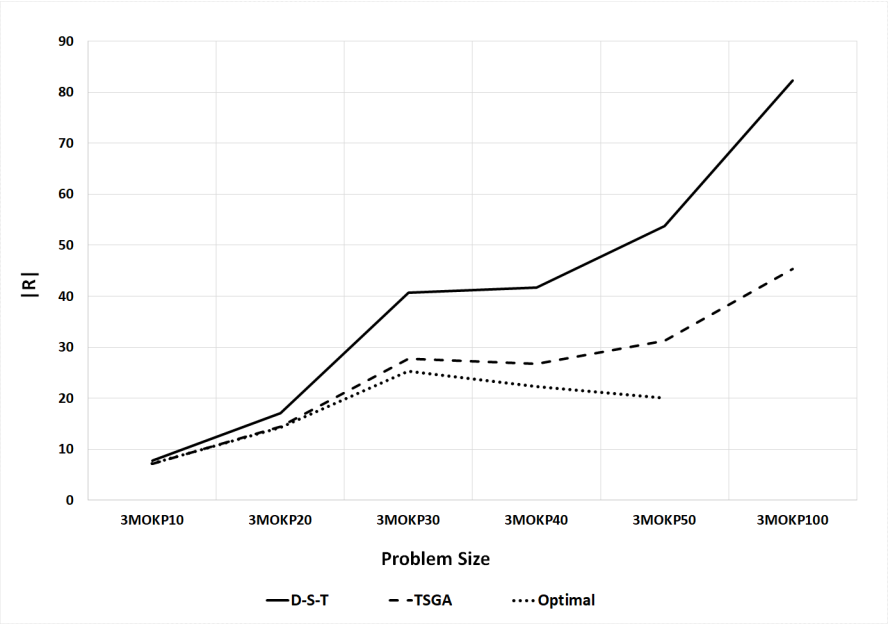


Figure 7: Sizes of the representative sets of different-sized 3MOKPs (average of 10 replications, $\alpha = 0.05$ - the optimal size for 3MOKP100s is not reported since $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ is not solvable within 4 hours).

Table 6: Performance Results on MOAP with three objectives*

			$\|R\|$			Solution Time (secs)			
Problem	N	α	D-S-T	TSGA	Optimal**	DMA	SBA	TDA	TSGA
3MOAP5	18.00	0.05	14.30	13.30	13.10	0.73	0.62	0.70	0.53
		0.10	11.10	9.40	8.80	0.48	0.39	0.48	0.36
		0.15	8.30	6.80	6.60	0.35	0.29	0.33	0.24
		0.20	5.80	4.90	4.30	0.25	0.20	0.27	0.18
		0.25	4.50	4.00	3.10	0.20	0.15	0.20	0.15
3MOAP10	189.40	0.05	53.90	34.60	31.40	12.36	4.72	5.35	2.74
		0.10	23.80	14.00	10.30	1.74	1.72	1.82	0.91
		0.15	14.90	7.20	5.67 (9)	0.93	1.05	0.96	0.44
		0.20	9.10	5.10	3.56 (9)	0.53	0.54	0.50	0.28
		0.25	5.70	3.30	2.44 (9)	0.33	0.33	0.28	0.18
3MOAP15	608.50	0.05	68.40	39.30	26.00 (4)	66.42	8.68	9.27	4.13
		0.10	28.40	14.20	-	4.10	3.16	3.20	1.24
		0.15	17.10	7.30	-	1.52	1.66	1.63	0.61
		0.20	10.60	4.90	-	0.85	1.00	0.88	0.40
		0.25	6.10	3.00	-	0.46	0.55	0.47	0.24
3MOAP20	1908.50	0.05	92.40	56.70	-	217.16	16.13	17.89	8.05
		0.10	34.90	16.90	-	9.24	5.42	5.59	1.98
		0.15	21.40	8.60	-	3.52	3.17	2.78	0.94
		0.20	13.60	4.80	-	1.60	1.70	1.45	0.51
		0.25	8.50	3.70	-	0.90	0.96	0.79	0.35

* Average of 10 replications are reported per cell. N represents the average number of non-dominated points, and α is the coverage gap value.

** Some of the replications of $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ could not be solved to optimality within 4 hours. For those problems, the values in parentheses show the numbers of replications that could be solved to optimality, where the averages are taken over those problems only. For brevity, no parentheses are used when all 10 replications could be solved to optimality and the cell is left blank when none of the 10 replications could be solved to optimality.

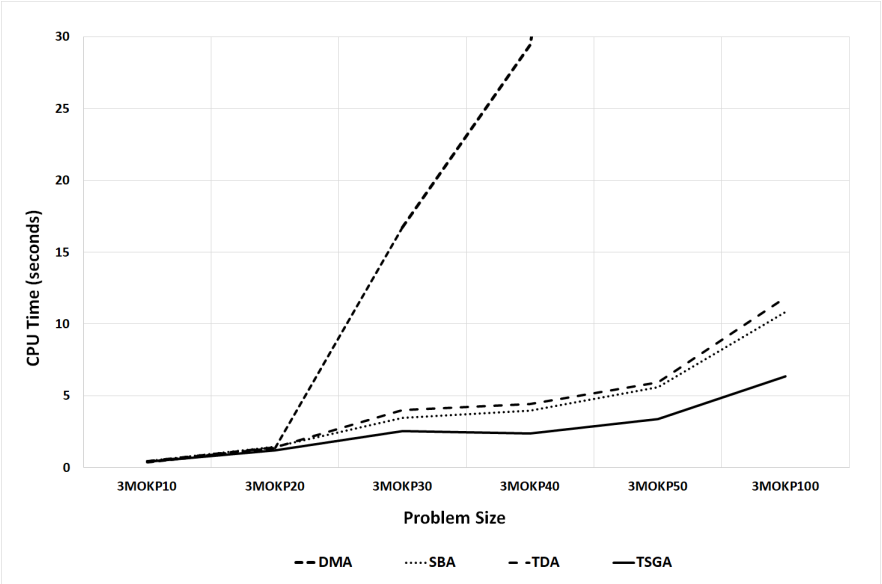


Figure 8: Solution times of different problem sizes of 3MOKPs (average of 10 replications, $\alpha = 0.05$)

Table 7: Performance Results on MOAP with four objectives*

			$\|R\|$			Solution Time (secs)			
Problem	N	α	D-S-T	TSGA	Optimal**	DMA***	SBA	TDA	TSGA
4MOAP5	37.50	0.05	27.30	27.00	26.50	2.21	3.86	3.50	2.70
		0.10	19.30	16.90	16.00	1.19	2.32	2.30	1.48
		0.15	13.10	10.70	9.40	0.68	1.48	1.32	0.85
		0.20	9.60	7.80	5.70	0.44	0.88	0.83	0.54
		0.25	6.70	5.50	3.70	0.33	0.53	0.49	0.36
4MOAP10	1527.00	0.05	162.30	104.30	75.33 (6)	7596.01	54.35	65.58	33.45
		0.10	55.20	29.50	-	32.10	13.65	14.93	5.44
		0.15	27.60	14.50	-	4.04	5.29	5.26	1.81
		0.20	15.90	8.00	-	1.28	2.38	2.30	0.75
		0.25	9.10	5.10	-	0.66	1.15	0.98	0.45
4MOAP15	6351.80	0.05	272.90	147.30	-	-	151.03	176.79	52.24
		0.10	76.40	33.50	-	644.76	30.52	33.54	7.03
		0.15	35.90	13.00	-	22.63	11.12	11.14	1.97
		0.20	20.60	7.50	-	4.00	4.77	4.62	0.93
		0.25	11.70	4.90	-	1.22	2.26	1.82	0.55

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, and α is the coverage gap value.

** Some of the replications of $(\boldsymbol{P}_{\boldsymbol{\alpha}}^{\boldsymbol{*}})$ could not be solved to optimality within 4 hours. For those problems, the values in parentheses show the numbers of replications that could be solved to optimality, where the averages are taken over those problems only. For brevity, no parentheses are used when all 10 replications could be solved to optimality and the cell is left blank when none of the 10 replications could be solved to optimality.

*** Results are not reported for 4MOAP15 instances ($\alpha = 0.05$) since DMA could not be finalized within 12 hours.

Table 8: Performance Results on MOAP with five objectives*

Problem	N	α	$\|R\|$			Solution Time (secs)			
			D-S-T	TSGA	Optimal**	DMA***	SBA	TDA	TSGA
5MOAP5	50.00	0.05	36.70	35.80	35.30	5.46	10.25	10.24	8.73
		0.10	23.30	21.00	20.00	1.71	19.94	5.87	3.68
		0.15	14.90	12.80	11.90	0.84	5.94	3.19	1.91
		0.20	11.20	8.30	7.10	0.55	1.77	2.06	0.97
		0.25	8.00	6.30	4.90	0.41	1.01	1.12	0.68
5MOAP10	3368.20	0.05	417.20	279.70	-	-	709.61	848.09	445.72
		0.10	108.90	60.60	-	962.44	127.65	130.43	40.66
		0.15	46.10	23.40	-	30.48	33.47	30.93	7.46
		0.20	25.60	12.30	-	5.56	13.36	12.16	2.88
		0.25	15.00	8.00	-	1.57	5.08	4.90	1.39

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, and α is the coverage gap value.

** The blank cells indicate that $(\boldsymbol{P}^*_{\boldsymbol{\alpha}})$ could not be solved to optimality within 4 hours in none of the 10 replications.

*** Results are not reported for 5MOAP10 instances ($\alpha = 0.05$) since DMA could not be finalized within 12 hours.

Table 9: Performance Results on multi-objective MIKP*

Problem	α	$\|R\|$		Solution Time (secs)		
		S-T	TSGA	SBA	TDA	TSGA
3MIKP100	0.05	112.80	63.90	6.64	8.96	3.72
	0.10	37.60	16.80	2.13	2.63	0.93
	0.15	22.10	7.60	1.21	1.34	0.41
	0.20	14.90	4.90	0.72	0.81	0.26
	0.25	9.90	3.60	0.46	0.52	0.20
4MIKP40	0.05	369.70	196.80	73.13	98.30	25.67
	0.10	86.60	37.00	12.53	16.13	3.37
	0.15	39.00	14.70	5.05	5.30	1.17
	0.20	23.70	8.00	2.57	2.55	0.54
	0.25	14.60	5.20	1.33	1.34	0.33
5MIKP20	0.05	351.50	238.70	245.56	317.65	119.94
	0.10	75.60	42.40	30.67	38.39	10.76
	0.15	34.70	17.10	8.99	10.80	2.95
	0.20	19.40	9.10	3.94	4.20	1.17
	0.25	11.70	5.60	2.17	1.93	0.55

* Average of 10 replications are reported per cell. N represents the average number of nondominated points, $|R|$ is the size of the representative subset, and α is the coverage gap value.

Table 10: Performance Results on MLKP*

	$\lvert R\rvert$									
	$\alpha = 0.05$		$\alpha = 0.10$		$\alpha = 0.15$		$\alpha = 0.20$		$\alpha = 0.25$	
3LKP40 Problem	S-T	TSGA	S-T	TSGA	S-T	TSGA	S-T	TSGA	S-T	TSGA
1	85.50	60	25.50	15	15.00	7	9.50	4	8.00	4
2	112.50	80	33.50	20	20.50	9	14.50	5	9.50	4
3	105.50	73	33.50	19	18.00	10	11.00	5	9.00	3
4	110.00	81	38.00	20	23.50	9	15.00	6	9.50	4
5	64.00	43	28.00	11	16.00	5	9.50	4	4.50	3
6	97.50	64	30.00	16	20.50	7	13.00	4	10.00	4
7	101.50	80	35.50	20	20.50	9	14.00	6	8.50	3
8	91.00	51	24.50	13	12.50	6	10.00	4	6.00	3
9	76.50	41	25.00	13	15.00	5	10.50	5	8.00	4
10	107.00	81	32.50	22	19.50	10	12.50	6	8.50	4

* S-T represents the average cardinality generated by SBA and TDA for each instance.

We also solve multi-objective linear programs to demonstrate the performance of TSGA in this setting. MLKP is similar to MOKP except that, rather than having all variables binary, now all variables are continuous in the interval $[0, 1]$. We conducted computational experiments with linear knapsack test instances with three objectives (3LKP40) using the generation scheme of Ceyhan et al. (2019). We report the results in Table 10. We see that TSGA requires roughly 50% of the number of representatives of the existing approaches (S-T), on average, over all instances and all coverage values.

As described in detail in Section 3, TSGA decomposes the search region into smaller search zones to efficiently find an optimal solution of $(\boldsymbol{P_n^{TSGA}})$. To demonstrate the contribution of this decomposition procedure to the computational efficiency of TSGA, we compare it against solving $(\boldsymbol{P_n^{TSGA}})$ without splitting it into submodels on 3MOKP40. We report the average and standard deviation values of the solution times on 10 different instances of the problem in Table 11. Results show that the decomposition procedure takes only a small fraction of running the algorithm without splitting. Furthermore, the computational times with the splitting procedure show very little variation, as measured by its standard deviations, in contrast with those obtained without splitting.

Table 11: Effect of Splitting $(\boldsymbol{P_n^{TSGA}})$

	Solution Time (secs) for 3MOKP40			
	Without splitting		With splitting	
α	Avg	StDev	Avg	StDev
0.05	613.16	488.17	2.38	0.95
0.10	60.39	36.88	0.73	0.24
0.15	24.83	14.15	0.37	0.12
0.20	15.11	8.68	0.23	0.04
0.25	9.02	6.24	0.16	0.07

To compare the actual number of submodels solved with the number of lower bound vectors, we observed these values in 3-objective knapsack problems with 40, 100, and 500 items when $\alpha = 0.05$. The numbers of lower bound vectors generated throughout the algorithm are 50, 67, and 110, while the numbers of submodels solved are 20, 28, and 39, for 3MOKP40, 3MOKP100, and 3MOKP500 instances, respectively. These results show that, by decomposing the solution set and eliminating redundant search zones, we only need to solve a fraction of the submodels created based on the lower bound vectors. This, together with the simpler submodels lead to substantial savings in the computational times as shown in Table 11.

Overall, we observe that TSGA outperforms other methods in both performance measures for all sizes of all problem types without any exceptions. Requiring much smaller representative sets than the other algorithms and doing this in fractions of their computational times for most problem-size combinations is a major improvement for this line of research. Furthermore, for most cases where we could find the optimal solution, we observe that the number representative points generated by TSGA is very close to the optimal. This indicates, at least for those problems, that there is very little room for further improvement.

5.2. Further Performance Analyses on TSGA and WSP

5.2.1. Dynamic Coverage Gap

Although TSGA requires the desired coverage gap value as an input to the process, it is directly applicable to a dynamic case where the desired coverage gap value may change during the solution process. In this case, α value needs to be updated until reaching the ultimate coverage gap value. To demonstrate the performance of TSGA changing the coverage gap value dynamically, we conduct additional experiments for TSGA for the 3-objective, 40-item KP, by changing the α value from 0.20 to 0.10 to 0.05 dynamically. Specifically, we start the algorithm with $\alpha = 0.20$, and keep reducing it to $\alpha/2$ once its current value is satisfied by all the representative points generates so far, until it achieves the value of 0.05. At the end of the algorithm, we ensure that the final representative set generated by TSGA satisfies the latest specified coverage gap value of $\alpha = 0.05$.

Table 12 compares the cardinality of the existing approaches (D-S-T) for $\alpha = 0.05$, TSGA for $\alpha = 0.05$, and TSGA for dynamic α values. As can be seen, TSGA still outperforms the existing approaches (DMA, SBA, and TDA) by far when the coverage gap starts with larger values and reduces to the desired value progressively for TSGA, whereas it is set to the desired value at the beginning for the other approaches. The results show that, converging to the desired coverage value progressively, increases the size of the representative set only slightly in these problems.

These results demonstrate the robustness of TSGA with respect to the value of the coverage gap that might be dynamically changing during the course of the algorithm. Notice that the dynamic change in the coverage value causes a lucky sequence of representative points in one of the instances (instance 8) and results in a fewer number of representative points.

Table 12: Cardinality Results on Dynamic Coverage Gap

			$\|R\|$		
Problem	Instance	N	D-S-T ($\alpha = 0.05$)	TSGA ($\alpha = 0.05$)	TSGA ($\alpha = 0.20 \rightarrow 0.10 \rightarrow 0.05$)
3MOKP40	1	245	44.00	30	31
	2	168	30.50	16	17
	3	175	28.00	17	19
	4	112	33.50	26	27
	5	212	46.00	32	33
	6	252	43.50	30	30
	7	269	45.00	26	31
	8	349	53.00	38	34
	9	90	24.50	14	18
	10	420	59.00	38	42

5.2.2. Effectiveness of WSP

As mentioned earlier, the problem-specific weights generated by WSP allows TSGA to generate representative points with high IRP values by taking into account the dense regions of the nondominated frontier of any problem instance. To demonstrate the effect of WSP on the performance of TSGA, we conduct additional numerical experiments to compare WSP against setting equal weights for all objectives, i.e., $\lambda_k = 1/m,\ k = 1, \ldots, m$.

In practice, different objectives, typically, have different ranges. To capture this aspect, we allow for different ranges in our computations here. We solve 3-objective knapsack problems with a large number of items (3MOKP500). We generate the objective coefficients from discrete uniform distributions in the ranges $[10, 100]$, $[100, 130]$, and $[30, 40]$, for objectives 1, 2, and 3, respectively. In order to demonstrate that the size of the representative set grows substantially for small α, we solve these problems for $\alpha = 0.01$. Table 13 presents the results. As can be seen in Table 13, the average cardinality of the representative sets generated by TSGA using WSP weights is less than that of using equal weights in all cases. Overall average reduction is 17.73%. These results demonstrate the power of our weighting scheme.

Table 13: Comparison of WSP Weights and Equal Weights ($\alpha = 0.01$)

		$\|R\|$		
Problem	Instance	TSGA w/ Equal Weights	TSGA w/ WSP Weights	(%) Reduction
3MOKP500	1	179	157	12.29
	2	139	107	23.02
	3	180	143	20.56
	4	199	171	14.07
	5	225	182	19.11
	6	209	170	18.66
	7	150	131	12.67
	8	257	218	15.18
	9	166	129	22.29
	10	175	141	19.43

6. Conclusions

In MOMIPs, as the number of objectives and the problem size increase, typically the number of nondominated points increases substantially. MOMIPs from practice tend to be large-scale problems and could have prohibitively many nondominated points. Generating a representative subset with desired properties not only facilitates the decision making process but also helps to keep the computational burden manageable. In this paper, we develop an algorithm, TSGA, that efficiently generates a representative subset guaranteeing any desired coverage gap value with a relatively small number of points. In this sense, TSGA makes the search through nondominated points of large-scale MOMIPs more manageable. In contrast with the existing literature, TSGA tries to capture the distributional properties of the nondominated points of the problem at hand and tries to concentrate towards dense regions using an effective weighting scheme, WSP.

Our extensive computational experiments on MOAPs, MOKPs, MIKPs, and MLKPs demonstrate that TSGA outperforms the existing approaches in terms of the size of the representative set and computational effort by a large margin for every level of coverage gap value we experimented with. The substantial improvement obtained by TSGA is mainly based on identifying and generating representative points from the regions that are dense in nondominated points, rather than using a greedy approach that finds the represent point that brings the immediate maximum benefit, as done by the existing approaches. Identifying the dense regions is achieved by approximating the nondominated set and estimating a set of weights for a linear function that navigates toward the dense regions of the approximated set. We experimentally demonstrate the benefits of these mechanisms on a variety of problems.

Another advantage of our approach is its flexibility to allow for changing the coverage gap requirement dynamically throughout the solution process. This allows a decision maker to start with higher coverage requirements and continue reducing until satisfied with the obtained cov-

erage while the computational burden stays reasonable. We demonstrate that such a flexibility has very little extra cost in terms of the increase in the representative set size corresponding to an eventual desired coverage gap value.

TSGA is designed in a way that the weights used in the objective function are estimated at the beginning of the algorithm. As an extension, the preferences of the DM can be incorporated into the solution process as well if it is desired to generate more representatives from preferred regions. If the DM can establish ranges of indifference for each objective, then the size of territories and the weights for each objective could be set accordingly. If the DM wishes to favor certain regions, it is possible to establish smaller territory sizes for such regions, creating differential coverage gap restrictions for different regions. We believe that our approach captures an important factor, the distributional properties of nondominated points, that has a big impact on the effective search for representative points. We expect this to attract new research efforts and further improvements that will enhance the study of MOMIPs.

Acknowledgement

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK), Program: 1001 Grant No: 215M844.

References

Boland, N., Charkhgard, H., Savelsbergh, M., 2016. The l-shape search method for triobjective integer programming. Mathematical Programming Computation 8, 217–251.

Boland, N., Charkhgard, H., Savelsbergh, M., 2017a. A new method for optimizing a linear function over the efficient set of a multiobjective integer program. European journal of operational research 260, 904–919.

Boland, N., Charkhgard, H., Savelsbergh, M., 2017b. The quadrant shrinking method: A simple and efficient algorithm for solving tri-objective integer programs. European Journal of Operational Research 260, 873–885.

Ceyhan, G., Köksalan, M., Lokman, B., 2019. Finding a representative nondominated set for multi-objective mixed integer programs. European Journal of Operational Research 272, 61–77.

Dächert, K., Klamroth, K., 2015. A linear bound on the number of scalarizations needed to solve discrete tricriteria optimization problems. Journal of Global Optimization 61, 643–676.

Dächert, K., Klamroth, K., Lacour, R., Vanderpooten, D., 2017. Efficient computation of the search region in multi-objective optimization. European Journal of Operational Research 260, 841–855.

Deb, K., 2001. Multi-objective optimization using evolutionary algorithms. volume 16. John Wiley & Sons.

Ehrgott, M., 2005. Multicriteria optimization. volume 491. Springer Science & Business Media.

Faulkenberg, S.L., Wiecek, M.M., 2010. On the quality of discrete representations in multiple objective programming. Optimization and Engineering 11, 423–440.

Hamacher, H.W., Pedersen, C.R., Ruzika, S., 2007. Finding representative systems for discrete bicriterion optimization problems. Operations Research Letters 35, 336–344.

Kaplan, H., Rubin, N., Sharir, M., Verbin, E., 2008. Efficient colored orthogonal range counting. SIAM Journal on Computing 38, 982–1011.

Karasakal, E., Köksalan, M., 2009. Generating a representative subset of the nondominated frontier in multiple criteria decision making. Operations research 57, 187–199.

Kırlık, G., Sayın, S., 2014. A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. European Journal of Operational Research 232, 479–488.

Kırlık, G., Sayın, S., 2015. Computing the nadir point for multiobjective discrete optimization problems. Journal of Global Optimization 62, 79–99.

Klamroth, K., Lacour, R., Vanderpooten, D., 2015. On the representation of the search region in multi-objective optimization. European Journal of Operational Research 245, 767–778.

Köksalan, M., Lokman, B., 2009. Approximating the nondominated frontiers of multi-objective combinatorial optimization problems. Naval Research Logistics (NRL) 56, 191–198.

Köksalan, M., Lokman, B., 2015. Finding nadir points in multi-objective integer programs. Journal of Global Optimization 62, 55–77.

Köksalan, M.M., 1999. A heuristic approach to bicriteria scheduling. Naval Research Logistics (NRL) 46, 777–789.

Kuhn, T., Ruzika, S., 2017. A coverage-based box-algorithm to compute a representation for optimization problems with three objective functions. Journal of Global Optimization 67, 581–600.

Laumanns, M., Thiele, L., Zitzler, E., 2006. An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. European Journal of Operational Research 169, 932–942.

Lokman, B., Köksalan, M., 2013. Finding all nondominated points of multi-objective integer programs. Journal of Global Optimization 57, 347–365.

Masin, M., Bukchin, Y., 2008. Diversity maximization approach for multiobjective optimization. Operations Research 56, 411–424.

Özarik, S., Lokman, B., Köksalan, M., 2020. Distribution based representative sets for multi-objective integer programs. European Journal of Operational Research 284, 632–643. doi:10.1016/j.ejor.2020.01.001.

Özlen, M., Burton, B.A., MacRae, C.A., 2014. Multi-objective integer programming: An improved recursive algorithm. Journal of Optimization Theory and Applications 160, 470–482.

Sayın, S., 2000. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. Mathematical Programming 87, 543.

Sayın, S., 2003. A procedure to find discrete representations of the efficient set with specified coverage errors. Operations Research 51, 427–436.

Shao, L., Ehrgott, M., 2016. Discrete representation of non-dominated sets in multi-objective linear programming. European Journal of Operational Research 255, 687–698.

Sylva, J., Crema, A., 2007. A method for finding well-dispersed subsets of non-dominated vectors for multiple objective mixed integer linear programs. European Journal of Operational Research 180, 1011–1027.

Vaz, D., Paquete, L., Fonseca, C.M., Klamroth, K., Stiglmayr, M., 2015. Representation of the non-dominated set in biobjective discrete optimization. Computers & Operations Research 63, 172–186.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G., 2003. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on evolutionary computation 7, 117–132.