Mathematical Methods of Operations Research (2024) 100:263–290
https://doi.org/10.1007/s00186-023-00847-8

ORIGINAL ARTICLE

An outer approximation algorithm for generating the Edgeworth–Pareto hull of multi-objective mixed-integer linear programming problems

Fritz Bökler[1] · **Sophie N. Parragh**[2] · **Markus Sinnl**[3] · **Fabien Tricoire**[4]

Received: 3 February 2023 / Revised: 14 November 2023 / Accepted: 6 December 2023 /
Published online: 4 January 2024

Abstract

In this paper, we present an outer approximation algorithm for computing the Edgeworth–Pareto hull of multi-objective mixed-integer linear programming problems (MOMILPs). It produces the extreme points (i.e., the vertices) as well as the facets of the Edgeworth–Pareto hull. We note that these extreme points are the extreme supported non-dominated points of a MOMILP. We also show how to extend the concept of geometric duality for multi-objective linear programming problems to the Edgeworth–Pareto hull of MOMILPs and use this extension to develop the algorithm. The algorithm relies on a novel oracle that solves single-objective weighted-sum problems and we show that the required number of oracle calls is polynomial in the number of facets of the convex hull of the extreme supported non-dominated points in the case of MOMILPs. Thus, for MOMILPs for which the weighted-sum problem is solvable in polynomial time, the facets can be computed with incremental-polynomial delay—a result that was formerly only known for the computation of extreme supported non-

✉ Markus Sinnl
markus.sinnl@jku.at

Fritz Bökler
fritz.boekler@uos.de

Sophie N. Parragh
sophie.parragh@jku.at

Fabien Tricoire
fabien.tricoire@wu.ac.at

1 Institute of Computer Science, Osnabrück University, Osnabrück, Germany

2 Institute of Production and Logistics Management/JKU Business School, Johannes Kepler University Linz, Linz, Austria

3 Institute of Business Analytics and Technology Transformation/JKU Business School, Johannes Kepler University Linz, Linz, Austria

4 Institute for Transport and Logistics Management, Vienna University of Economics and Business, Vienna, Austria

dominated points. Our algorithm can be an attractive option to compute lower bound sets within multi-objective branch-and-bound algorithms for solving MOMILPs. This is for several reasons as (i) the algorithm starts from a trivial valid lower bound set then iteratively improves it, thus at any iteration of the algorithm a lower bound set is available; (ii) the algorithm also produces efficient solutions (i.e., solutions in the decision space); (iii) in any iteration of the algorithm, a relaxation of the MOMILP can be solved, and the obtained points and facets still provide a valid lower bound set. Moreover, for the special case of multi-objective linear programming problems, the algorithm solves the problem to global optimality. A computational study on a set of benchmark instances from the literature is provided.

Keywords Multi-objective optimization · Outer approximation · Mixed-integer programming

1 Introduction and motivation

Let $A \in \mathbb{Q}^{m\times n}, C \in \mathbb{Q}^{p\times n}, b \in \mathbb{Q}^m$, $p \geq 2$ and $n = n_1+n_2$. Let $\mathcal{X} = \{x \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} : Ax \geq b\}$, where we assume that $\mathcal{X} \neq \emptyset$. We are interested in *multi-objective mixed-integer linear programming* problems (MOMILPs) which can be defined as follows:

$$\begin{aligned} &\min\ Cx \\ &\text{s.t. } x \in \mathcal{X}, \end{aligned} \tag{MOMILP}$$

where we assume that $\{C^i x : x \in \mathcal{X}\}$ is bounded from below for each row C^i of C. Note that if $n_1 = 0$, the problem is called a *multi-objective linear programming* problem (MOLP) and for $n_2 = 0$, it is called a *multi-objective integer linear programming* problem (MOILP).

For $k \in \mathbb{N}$, let $\mathbb{R}^k_{\geq 0} := \{z \in \mathbb{R}^k : z \geq 0\}$, i.e., $R^k_{\geq 0}$ is the non-negative orthant of dimension k. Let $f(\mathcal{X}) = \{Cx : x \in \mathcal{X}\}$. For (MOMILP) we define

$$\mathcal{Q}^+ := \operatorname{conv} f(\mathcal{X}) + \mathbb{R}^p_{\geq 0},$$

the *Edgeworth–Pareto hull* (Lotov et al. 2004; Ehrgott et al. 2016) that is a finitely generated rational polyhedron (see Lemma 1). In this work, we are interested in computing the *extreme points (i.e., vertices)* and *facets* of $\mathcal{Q}^+$. The set $\mathcal{Q}^+$ is illustrated in Fig. 1 for two objectives (y_1 and y_2). The extreme points are the red points with the black circles around them.

Our work is motivated by the fact that many practical problems involve several, often conflicting objectives, such as profitability or cost versus environmental concerns (Demir et al. 2014; Ramos et al. 2014; Eskandarpour et al. 2021) or customer satisfaction (Braekers et al. 2016). This means that there is, in the general case, no single optimal solution which optimizes all objectives simultaneously but a set of trade-off solutions which are better (according to some criteria) than other feasible solutions but incomparable among each other. A popular concept of optimality in this context

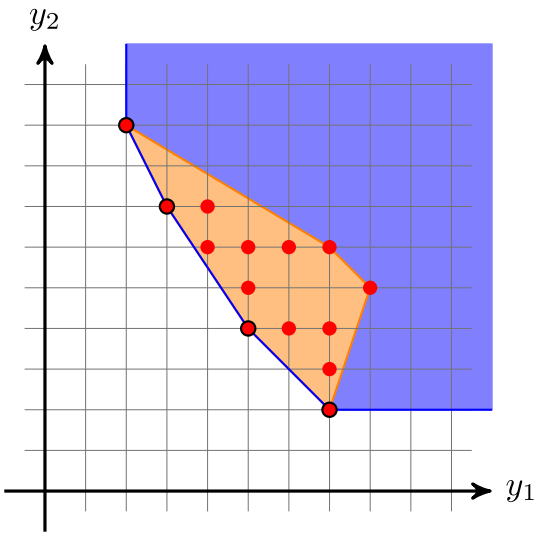


Fig. 1 Exemplary illustration of the polyhedron $\mathcal{Q}^+ = \text{conv}(\mathcal{Q}) + \mathbb{R}^2_{\geq 0}$. (Color figure online)

is *Pareto optimality* (Ehrgott 2005). A solution is called Pareto optimal (or *efficient*) if no objective function value can be improved without deteriorating another (for a formal definition of Pareto optimality and other concepts discussed in the introduction, we refer to Sect. 2). The images of these solutions in the space of objective function values (the *objective space* or *criterion space*) are called *non-dominated* points and they together form the *non-dominated frontier* or *Pareto frontier*. In multi-objective optimization, we usually aim to identify the non-dominated frontier and to identify at least one efficient solution for each non-dominated point. Depending on the concrete type of multi-objective problem, this task can present with several sources of difficulty: For MOLP, the non-dominated frontier contains an infinite number of points, as all the points between two extreme points of the non-dominated frontier are also contained in the non-dominated frontier. However, all the extreme points of the non-dominated frontier can be found by solving *weighted-sum scalarizations* of the MOLP and with these extreme points the remaining points (and associated solutions) can be recovered. For MOILP, there exist *non-supported non-dominated points*, which cannot be found using any weighted-sum scalarization. Finally, in case of MOMILP, both issues can occur. Figure 2 gives exemplary non-dominated frontiers with two objectives for each of the three problem types.

We note that the extreme points of $\mathcal{Q}^+$ coincide with the extreme supported non-dominated points of MOMILP.

1.1 Solution methods for multi-objective programming problems

Due to these different sources of difficulty, different types of solution algorithms exist for these problem families:

For MOLP, solution algorithms relying on solving a series of weighted-sum scalarizations are very popular. They originate in the seminal works (Aneja and Nair 1979;

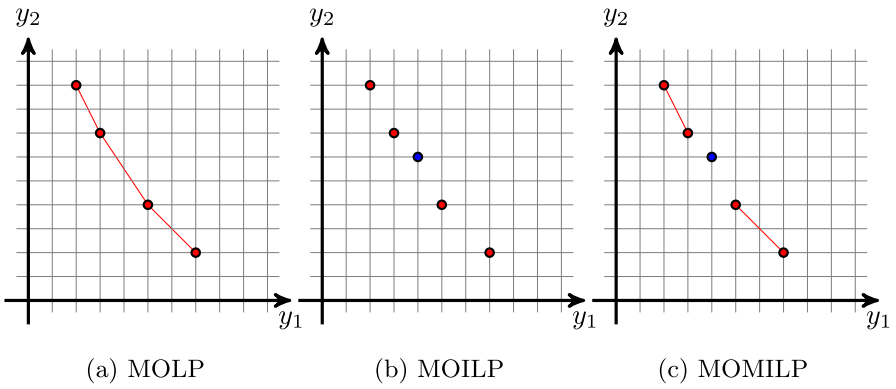


Fig. 2 Exemplary non-dominated frontiers for MOLPs, MOILPs and MOMILPs (two objectives). The non-dominated frontiers consist of the points and the lines. The red points are supported and the blue points are non-supported. (Color figure online)

Cohon 1978; Dial 1979) for the bi-objective case. Solution methods for MOLP can be classified as *inner approximation* (IA) or *outer approximation* (OA) methods. We note that they usually rely on some form of duality to work, thus existing methods are mostly restricted to MOLP or some other convex multi-objective programming problems. However there are some recent works (Özpeynirci and Köksalan 2010; Przybylski et al. 2010, 2019; Halffmann et al. 2020) which can be classified as IA approaches capable of dealing with MOMILP. Naturally, IA approaches cannot compute non-supported non-dominated points, but similar to the OA approach we present in this work, they aim to compute a representation of the Edgeworth–Pareto hull of a MOMILP. We give a detailed overview over existing work in Sect. 3.

For the purpose of computing a complete non-dominated frontier representation, existing solution algorithms can be classified into two groups, namely *decision space* algorithms and *objective space* algorithms. Objective space algorithms are of an iterative nature and add constraints in the objective space to cut off already discovered non-dominated points. The most popular objective space method is probably the ε-constraint method (Chankong and Haimes 2008), which is restricted to MOILPs. However, there exist objective space methods which are also suitable for MOMILPs with two objectives, such as the triangle-splitting method (Boland et al. 2015), the boxed-line method (Perini et al. 2020), the search-and-remove method (Soylu 2018) or the method by Rasmi and Türkay (2019).

Decision space methods are usually multi-objective extensions of the classical branch-and-bound algorithm (B&B) for single-objective mixed-integer programming. All recent successful implementations of multi-objective B&B algorithms rely on *lower bound sets* (Ehrgott and Gandibleux 2007), see, e.g., Gadegaard et al. (2019), Parragh and Tricoire (2019), De Santis et al. (2020), Forget et al. (2022b), Adelgren and Gupte (2022), Parragh et al. (2021) and Eichfelder et al. (2023) but they are typically either restricted to two objectives or they only address the pure integer and not the mixed integer case. Lower bound sets are sets in the objective space that either contain the non-dominated frontier or a set of points which, together, dominate

the non-dominated frontier. They are a natural multi-objective extension of the lower bounds obtained by, e.g., relaxations in single-objective optimization. In a bi-objective context, in more recent contributions, they are usually computed "from the inside" by IA methods (e.g. Parragh and Tricoire 2019; Adelgren and Gupte 2022). Stidsen et al. (2014) relied on solving a single weighted-sum scalarization of the linear relaxation to obtain what is called an objective function level curve and classifies as an OA. For tackling problems with more than two objectives, Forget et al. (2022a, b), Forget and Parragh (2023) use an OA scheme to compute bound sets "from the outside". They solve the *linear programming* (LP) relaxation of the MOMILP using an implementation of Benson's OA algorithm for MOLPs. The OA algorithm we are proposing can directly obtain such a lower bound set for MOMILPs without the need of solving a relaxation. This direct approach thus yields a potentially tighter approximation.

For a more detailed overview on solution methods for MOILPs and MOMILPs, we refer to Halffmann et al. (2022).

1.2 Contribution

In this work, we propose an OA algorithm which computes the extreme points and facets of $\mathcal{Q}^+$ for MOMILPs. Our algorithm is motivated by the OA approach of Benson (1998) for MOLPs. It uses an *oracle* which consists of solving single-objective *weighted-sum problems*. We show that our algorithm needs a number of oracle calls which is polynomial in the number of facets of $\mathcal{Q}^+$.

Our algorithm closes the following research gaps:

- We show that the OA paradigm can be extended to the MOMILP setting. In order to do this, we extend the concept of geometric duality proposed by Heyde and Löhne (2008) for MOLPs to the MOMILP setting.
- Our OA algorithm has several attractive features to be a key ingredient in the next generation of multi-objective B&B algorithms: Compared to IA algorithms, which need to be run to termination, our algorithm provides a valid lower bound set at any iteration. We also note that IA termination cases are subject to numerical instability. Moreover, as our OA directly operates on the MOMILP and not on the LP-relaxation of it (as for example the OA used in Forget et al. (2022a, b) does), it can potentially provide much tighter bound sets. In fact, the algorithm also provides a valid lower bound set when the oracle is called with any relaxation of the original problem, such as the LP-relaxation, potentially augmented with valid inequalities. This opens perspectives for multi-objective branch-and-cut algorithms. Finally, next to providing a lower bound set at any point of its execution, each oracle call may also produce a new efficient solution. This makes it attractive to use within multi-objective B&B also from the primal side.
- We also provide some results on theoretical runtime and close some open questions in this regard: Even for structurally simple MOLPs, the number of extreme points can be exponential in the size of the input (Ruhe 1988) and also for many well-known multiobjective combinatorial optimization (MOCO) problems the number of extreme points is super-polynomial. To separate some of the complexities of these problem classes, a new *output-sensitive complexity measure* has emerged (see

Sect. 2.3). In this context, Bökler and Mutzel (2015) show that the extreme points of the Edgeworth–Pareto hull of MOLP and MOCO (generalizing to MOMILP) can be computed efficiently, if the weighted-sum scalarization can be solved efficiently, i.e., in polynomial time. While there are efficient algorithms for the computation of non-dominated facets of an MOLP (Bökler 2018), no such theorem could yet be proven for the facet computation of the Edgeworth–Pareto hulls of discrete problems. Our work closes this research gap, as with the OA algorithm we propose in this work, we are able to propose a similar theorem to the one by Bökler and Mutzel (2015) for MOMILPs: If the weighted-sum scalarization of a given MOMILP is solvable in polynomial time, the facets can be computed with incremental-polynomial delay.

Moreover, we note that for MOLPs the extreme points and facets of $\mathcal{Q}^+$ coincide with the non-dominated frontier. Consequently, our OA algorithm solves MOLPs to global optimality.

Finally, we observe that computing the extreme points and facets of $\mathcal{Q}^+$ also corresponds to the problem addressed in parametric integer linear programming with parametrization in the objective function (Geoffrion and Nauss 1977), which is an interesting problem on its own. Moreover, a related problem is to compute the extreme points and/or facets of the convex hull of a linear projection of a mixed-integer problem onto a small subspace. Our algorithm is also capable of computing this projection. However, our focus in this paper will be on the optimization side.

1.3 Outline

In Sect. 2 we provide notation, definitions and other preliminaries. Section 3 provides an overview of existing inner and outer approximation schemes. Section 4 provides a general outline of OA algorithms for multi-objective optimization. They require a point separation oracle specific to the addressed problem class. In Sect. 5 we first present the generalization of geometric duality to the MOMILP setting in Sect. 5.1 and then propose two separation oracles for MOMILPs in Sects. 5.2 and 5.3. The theoretical runtime of the algorithm when using the presented oracles is discussed in Sect. 5.4. Section 6 contains a computational study on instances from the literature and also discusses implementation details. We give empirical results on the numerical accuracy of our approach and provide a comparison with `PolySCIP` (Borndörfer et al. 2016), which is an IA solver for MOMILPs. Section 7 concludes the paper.

2 Notation, definitions and preliminaries

2.1 Polyhedra and faces

Given $a \in \mathbb{R}^n$, $a \neq 0$, and $\alpha \in \mathbb{R}$, the set $H^a_\alpha := \{a^\mathsf{T} x = \alpha\}$ is called a *hyperplane*. Let $P = \{Ax \geq b\}$ for $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, m, n \in \mathbb{N}$. P is called a *polyhedron*. A polyhedron is called *rational*, if there are $A' \in \mathbb{Q}^{m \times n}$ and $b' \in \mathbb{Q}^m$, such that $P = \{A'x \geq b'\}$. We say H^a_α is *valid* for P, if $a^\mathsf{T} x \geq \alpha$ for all $x \in P$. A *face* of P

is either a set $P \cap H^a_\alpha$ for any valid hyperplane H^a_α, or P itself. A valid hyperplane *supports* P *in* F if $F := P \cap H^a_\alpha \neq \emptyset$. A face with affine dimension 0 is called an *extreme point (vertex)*, one with affine dimension $\dim P - 1$ is called a *facet*. Rational polyhedra are well-known to have rational-valued extreme points only. Since for full-dimensional polyhedra, the coefficients (a, α) of a hyperplane supporting a polyhedron in a facet are uniquely determined up to scaling, we often identify a facet with its defining (in)equality. We denote such a hyperplane as *facet supporting*.

A vector $r \in \mathbb{R}^n$ is called a *ray* of P, if and only if for every $x \in P$ and $\gamma \geq 0$ the point $x + \gamma r \in P$. The set of all rays of P is called its *recession cone* or rec P. If rec $P = \{0\}$, then we call P a *polytope*.

For two sets $A, B \subseteq \mathbb{R}^n$, we write $A + B = \{a + b : a \in A, b \in B\}$. For a linear map $M \in \mathbb{R}^{m \times n}$, we define $M \cdot A = \{Ma : a \in A\}$. The set conv A is the convex hull of A, the set cone A is the conical hull of A. A convex set P is a polyhedron if and only if there exist finite sets $E \subseteq \mathbb{R}^n$ and $D \subseteq \mathbb{R}^n$ such that $P = \text{conv } E + \text{cone } D$. This representation is called the *generator representation* of P. If a convex set has a representation as above, it is said to be *finitely generated*. Observe that with these definitions, polyhedra are always closed, convex and finitely generated.

For more background on polyhedral theory, we refer to Ziegler (2012) and Conforti et al. (2014).

2.2 Multi-objective optimization

For $y, y' \in \mathbb{R}^p$, the relations $y = y'$, $y \leq y'$, and $y < y'$ apply component-wise. We call the set $\mathcal{Q} := f(\mathcal{X})$ the *feasible set in the objective space*. A point $y \in \mathcal{Q}$ is *non-dominated*, if and only if there is no $\hat{y} \in \mathcal{Q}$ with $\hat{y} \leq y$ and $y \neq \hat{y}$. We call the point $y^I \in \mathbb{R}^p$ with $y^I_i = \min\{y_i : y \in \mathcal{Q}\}$, $i = 1, \ldots, p$ the *ideal point*. Note that not every MOMILP has an ideal point. However in this work as in most other works dealing with MOMILPs, we always assume an ideal point exists except for Sect. 5.1. A feasible solution $x \in \mathcal{X}$ is called *efficient*, if and only if Cx is non-dominated. A feasible solution x is a *supported efficient solution*, if and only if there is a $w \in \mathbb{R}^p$ with $w > 0$, such that x is an optimal solution to the *weighted-sum problem* or *weighted-sum scalarization*:

$$\min \left\{ w^\mathsf{T} Cx : x \in \mathcal{X} \right\} \tag{WSUM}$$

For every supported *efficient* solution x, we call $y = Cx$ a *supported non-dominated* point. A point $y \in Q$ is an *extreme supported non-dominated point* if it is a supported non-dominated point which cannot be obtained as the convex combination of other supported non-dominated points.

Supported non-dominated points can also be characterized by the Edgeworth–Pareto hull: A point $y \in \mathcal{Q}$ is an extreme supported non-dominated point if and only if it is an extreme point of $\mathcal{Q}^+$ (Ehrgott et al. 2016). In our problem setting, we are interested in the facets of $\mathcal{Q}^+$ as they provide us with a lower bound set; as well as in *efficient* solutions x such that Cx is an extreme point of $\mathcal{Q}^+$, as they provide us with primal solutions. Note that in the context of MOLP and linear vector optimiza-

tion in general, the Edgeworth–Pareto hull is usually called the *upper image* (Löhne 2011). The different name can be motivated by the fact that in MOLP the image $\mathcal{Q}$ is already convex and the additional convex hull operation in the definition of the Edgeworth–Pareto hull is superfluous.

For more background on multi-objective optimization in general, we refer to, e.g., Chinchuluun and Pardalos (2007) and Ehrgott (2005).

Lemma 1 *From the definition of $\mathcal{Q}^+$ we get the following facts:*

1. *rec $\mathcal{Q}^+ = \mathbb{R}^p_{\geq 0}$*
2. *$\mathcal{Q}^+$ is a rational polyhedron.*

Proof 1. It is clear that every extreme ray of $\mathbb{R}^p_{\geq 0}$ is an extreme ray of $\mathcal{Q}^+$. The existence of any other extreme ray of $\mathcal{Q}^+$ is a contradiction to the assumption that an ideal point exists.
2. For every MOMILP, the set conv $\mathcal{X}$ is a rational polyhedron (Conforti et al. 2014). Because C is a linear map, we have $C(\text{conv}\,\mathcal{X}) = \text{conv}(C\mathcal{X}) = \text{conv}\,\mathcal{Q}$. Hence, the set conv $\mathcal{Q}$ is a linear image of conv $\mathcal{X}$ and thus is a polyhedron. As the matrix C consists of rational entries only, conv $\mathcal{Q}$ is also rational. Together with (1.), adding $\mathbb{R}^p_{\geq 0}$ preserves this property.

□

2.3 Running times and output-sensitive complexity

Since the number of facets and extreme points of the Edgeworth–Pareto hull (also called *parametric complexity*) of many problems can grow exponentially in the input size, the traditional aim of a polynomial running time for algorithms computing this entities is not useful. Instead, we usually regard an algorithm as efficient, if its running time can be bounded by a polynomial in the input and the output size. We call such an algorithm an *output-polynomial time* algorithm.

A more restrictive notion is an *incremental-polynomial delay* algorithm: The kth-delay of an enumeration algorithm is defined as the time between producing the kth and $(k+1)$th output, including the time to produce the first output and the time after the last output until termination. In the incremental-polynomial delay model, we require the kth delay to be bounded by a polynomial in the input size and k. In contrast to output-polynomial time algorithms, incremental-polynomial delay algorithms are not allowed to print all outputs at the end, but at certain time points there have to be updates. However, the model still allows the delays between the outputs to grow depending on the size of the output so far.

This is remedied by the most restrictive model we discuss: A *polynomial delay* algorithm is an enumeration algorithm where every delay is bounded by a polynomial in the input size.

Naturally, a polynomial delay algorithm is an incremental-polynomial delay algorithm and an incremental-polynomial delay algorithm is also an output-polynomial algorithm but the reverse implications do not hold in general. See also Bökler et al. (2017) for a more in-depth exposition of the subject.

3 State-of-the-art

In this section, we provide an overview of existing IA and OA algorithms for multi-objective optimization problems. Since $\mathcal{Q}^+$ is a polyhedron, there are two general ways to represent it: by its generators or by inequalities. In the first representation, it suffices to compute the extreme points, since as we assume $\mathcal{Q}^+$ to be bounded from below, it does not contain a line and the extreme rays of $\mathcal{Q}^+$ are always the unit vectors. In a multi-objective or parametric optimization setting, the extreme point representation is more compelling, since it provides us with objective value vectors of efficient solutions. In our setting however, we are more interested in computing inequality representations. In bi-objective problems, there is not much of a difference between these representations, but with a higher number of objectives, the computational cost of switching between them grows exponentially in the worst case.

To the best of our knowledge, the first works that were concerned with computing extreme points of $\mathcal{Q}^+$ of MOMILP were independently conducted by Aneja and Nair (1979), Cohon (1978), and Dial (1979). It is usually referenced as the *dichotomic approach* today and only works in the bi-objective case. The dichotomic approach can be implemented with polynomial delay (Bökler et al. 2017). For biobjective MOLP there exists an even more efficient algorithm that runs in polynomial delay and needs only polynomial space in the input size (Bökler et al. 2017).

In the seminal work of Benson (1998), the first outer approximation algorithm for solving MOLPs in the objective space is given. It forms the basis of the open source vector linear programming solver `bensolve` (Löhne and Weißing 2017). Its key idea is that $\mathcal{Q}^+$ can be approximated from the "outside", by iteratively intersecting the current approximation with face supporting halfspaces, until the approximation S is equal to $\mathcal{Q}^+$. Initially, the approximation $S = y + \mathbb{R}^p_{\geq 0}$, where y is the ideal point, and an interior point $\hat{r} \in \operatorname{int}\mathcal{Q}^+$ are required. Then, in each step a point that lies on the intersection of the boundary of $\mathcal{Q}^+$ and the line connecting one of the current extreme points y^* of S and $\hat{r}$ is determined. It is used to obtain a supporting hyperplane of $\mathcal{Q}^+$, by solving an LP, or to prove that y^* is an extreme point of $\mathcal{Q}^+$. This algorithm is capable of computing the extreme points of an MOLP in output-polynomial time and the facets even with incremental-polynomial delay (Bökler 2018).

Extending the work by Benson, Ehrgott et al. (2012) develop a dual algorithm based on the geometric duality theory of MOLP by Heyde and Löhne (2008). It operates in a dual space of the objective space. In this dual space, Heyde and Löhne define a *lower image* $\mathcal{D}$ that is geometrically dual to the upper image $\mathcal{Q}^+$. The dual algorithm can then be described as a version of Benson's outer approximation algorithm for $\mathcal{D}$. This dual algorithm has a crucial property that makes it more attractive in theory and practice: instead of solving an involved LP to find separating hyperplanes, in this representation, it suffices to solve weighted-sum LPs (Bökler and Mutzel 2015).

In spite of its dual description, the dual algorithm can be described as operating in the objective space in the following way: The initial dual outer approximation is essentially a non-dominated extreme point y of the problem with an attached non-negative orthant in the objective space, i.e., $y + \mathbb{R}^p_{\geq 0}$. Due to geometric duality theory, choosing an extreme point of the current dual outer approximation is the same as

choosing a facet of the current inner approximation in the objective space. Finding a new supporting hyperplane in dual space is again equivalent to finding a new supported point in the objective space. In this view, the dual of Benson's algorithm is also the first instance of an inner approximation for MOLP with a general number of objective functions. It computes the extreme points of an MOLP with incremental-polynomial delay and the facets in output-polynomial time (Bökler 2018). Interestingly, the outer approximation is thus more efficient in computing facets and the inner approximation is more efficient in computing extreme points.

Very recently, Csirmaz (2021) introduced a unifying perspective on inner and outer approximation schemes for MOLP, giving skeletal algorithms for both. Each scheme starts from an initial approximation S of $\mathcal{Q}^+$. Then, the inner (outer) approximation scheme relies on a plane (point) separating oracle determining if an extreme point (facet) is already part of $\mathcal{Q}^+$. If yes, it is marked as final, otherwise, S is updated. The output to either scheme is $\mathcal{Q}^+$ in double description format (i.e., extreme points and facets of $\mathcal{Q}^+$). More details on the outer approximation perspective is given in Sect. 4. Csirmaz (2021) shows that Benson's algorithm and its variants follow the outer approximation scheme whereas the dual variant of Ehrgott et al. (2012) falls into the described inner approximation scheme.

While Benson's algorithm and its variants address the multi-objective linear case, some algorithms for obtaining the extreme supported non-dominated points of MOMILPs have also been proposed in the past years. To the best of our knowledge, none of them approximates $\mathcal{Q}^+$ from the outside and as such classifies as an OA algorithm. Przybylski et al. (2010) propose a recursive scheme that resorts to solving bi-objective problems via dichotomic search derived from the weights associated with extreme points of a given facet of the weight-set polytope. The weight set is the set of all eligible weights in a weighted-sum scalarization. Decomposing it into weight-set components, for which a given extreme supported image is optimal, results in a weight-set decomposition. Computing this decomposition motivates the development of the proposed scheme. Przybylski et al. (2010) show that their method works for three objectives and in theory also for more. An enumerative procedure is due to Özpeynirci and Köksalan (2010). It relies on what the authors call stages which are defined by p points. In each step, the normal weight vector to the hyperplane H defined by a given stage R is used in the weighted-sum scalarization of the MOMILP to obtain a new point y. If y corresponds to one of the points of R, R is identified as a facet defining stage. Otherwise, the new point y is added to the set of non-dominated points and the list of to be explored stages is updated with y. A main drawback of the method is that it requires the generation of dummy points to initialize the search.

Bökler and Mutzel (2015) propose IA algorithms for MOCO problems. Their work builds on the dual variant of Benson's algorithm, observing that the LPs that need to be solved in the MOLP case can be replaced by solving weighted-sum problems of the original MOCO problem, showing that the geometric duality theory for MOLP by Heyde and Löhne (2008) can also be applied to non-convex problems if the interest lies in the Edgeworth–Pareto hull of the problem. Bökler and Mutzel (2015) show that for every fixed number of objectives and MOCO problem with a polynomially solvable weighted-sum scalarization, the extreme points can be computed with incremental-polynomial delay. This result can be straightforwardly extended to MOMILPs under

the assumption that an ideal point exists: By the same arguments of the classic proof that (the decision version of) MOMILP $\in$ **NP**, we see that the encoding lengths of optimal extreme point solutions of the integer hull remain polynomially bounded in the input size. Optimizing over the integer hull of an MOMILP thus is analogous to the combinatorial case.

Borndörfer et al. (2016) develop a general purpose solver named `PolySCIP` for MOILP and MOLP with an arbitrary number of objectives as part of the constraint integer programming suite SCIP. To compute the extreme supported non-dominated points of MOILP and MOLP, they rely on a lifted weight space polyhedron. For the special case of MOLP, this polyhedron is also implicitly defined by Luc (2011) in their parametric duality. It is shown by Dörfler and Löhne (2018) that it is equivalent to the geometric dual polyhedron by Heyde and Löhne (2008). Hence in this case, the lifted weight space polyhedron shares the duality properties of the geometrically dual polyhedron. However, in Borndörfer et al. (2016) it is used without any theoretical considerations. In the MOILP case, the lifted weight space polyhedron is an algorithmically easier to exploit tool compared to the weight set decomposition introduced by Przybylski et al. (2010). However, no duality characteristics are known for it and also no extensions to MOMILPs. Algorithmically, `PolySCIP` employs a dual Benson outer approximation algorithm on this lifted weight space/dual space, extending the work of Bökler and Mutzel (2015) from MOCO to MOILP. Hence, `PolySCIP` also falls into the category of IA algorithms.

Even more recently, Halffmann et al. (2020) have proposed an IA algorithm for three-objective mixed integer linear programs, with the purpose of determining the weight-set decomposition. It relies on solving bi-objective subproblems via the dichotomic approach. Finally, building on the work of Özpeynirci and Köksalan (2010) and Przybylski et al. (2010), Przybylski et al. (2019) have also proposed a new IA algorithm for MOMILPs. In each step, it generates the convex hull of the previously identified extreme supported non-dominated points. Then, a yet unexplored facet is chosen and used to define the next weighted-sum scalarization to be solved. The solution may either be a new extreme supported non-dominated point or identify the facet to be (partially) part of $\mathcal{Q}^+$.

Our OA algorithm relies on the idea of cutting a point from a polyhedron by a hyperplane in a generic fashion. While this has not yet been used to generate the Edgeworth–Parto hull of MOMILPs in the objective space, it is at the core of Fenchel cuts (Boyd 1993, 1994). These are generic cuts developed for single-objective integer programming, where they are used to cut-off infeasible LP relaxation solutions in decision space. Their generalization to local cuts (Applegate et al. 2001; Chvátal et al. 2013) is also well known in the single-objective domain. Applications of the local cut paradigm in the single-objective domain include Avella et al. (2010) where it is used to find generic cuts for the generalised assignment problem and Kaparis and Letchford (2010) where it is used in the context of the knapsack problem. In our work, we transfer these developments, including the LPs used for generic separation, from the single-objective domain to the multi-objective domain.

Finally we note that for multi-objective mixed integer convex optimization problems, an outer approximation based B&B algorithm has recently been proposed by De Santis et al. (2020). In the lower bound computation, building on the work of Ehrgott

et al. (2011) and Löhne et al. (2014), a convex relaxation-based outer approximation is used. It relies on the computation of supporting hyperplanes. This computation is steered by the available local upper bounds within the B&B algorithm. Eichfelder et al. (2023) generalize the ideas of De Santis et al. (2020) to the non-convex case. Also here the computation of the outer approximation is steered by local upper bounds, i.e., in the fathoming step of the B&B scheme. It relies on a relaxation of the original problem. The computation of the complete Edgeworth–Pareto hull via a cutting plane method is not the aim of their study.

4 The outer approximation algorithm

In this section, we give an outline of the general OA algorithm. Moreover, we do this in terms of the oracle view by Csirmaz (2021) and generalize this view from MOLP to MOMILP. The key ingredient in implementing such algorithms for concrete problem classes like MOLPs or MOMILPs is the specification and implementation of a *point separating oracle* for the respective problem class.

Definition 2 (Point separating oracle) A point separating oracle for a polyhedron $P \subset \mathbb{R}^p$ is a black box algorithm which takes as input a point $y^* \in \mathbb{R}^p$ and returns a tuple $(status, H)$.
The output is as follows: i) $(inside, \emptyset)$, if $y^* \in P$, or ii) $(outside, H)$, where H is a supporting hyperplane $H = \{y \in \mathbb{R}^p : w^T y = \alpha\}$ of P such that $w^T y^* < \alpha$ and $w^T \widehat{y} \geq \alpha$ for each $\widehat{y} \in P$ (i.e., H separates y^* from P).

Note that in the corresponding definition of Csirmaz (2021) a point separating oracle must give back a hyperplane H which induces a facet of P. In our definition we do not make this restriction. Naturally, the convergence-behaviour of an OA algorithm depends on the used point separating oracle. In the original work by Benson (1998) for MOLP, the supporting hyperplanes to $\mathcal{Q}^+$ were only face supporting. The first facet supporting hyperplane producing oracles were given in Bökler (2018); Csirmaz (2021) for MOLP. Hence, we are interested in facet supporting hyperplanes and we show in Sect. 5 that our two proposed point separating oracles for MOMILPs produce facets of $\mathcal{Q}^+$.

Remark 3 Note that the point separating oracle must be called for $\mathcal{Q}^+$ even though a complete description of $\mathcal{Q}^+$ is of course only known at the end of the outer approximation algorithm. However, as we show in Sect. 5, given a MOMILP instance, we can construct point separating oracles for $\mathcal{Q}^+$ without having a complete description of $\mathcal{Q}^+$ available.

The OA algorithm is described in Algorithm 1. The approximations $\mathcal{S} = \mathcal{S}_0 \supset \mathcal{S}_1 \supset \ldots$ used within the algorithm are considered to be stored in a double description format, i.e., as extreme points and facets. The algorithm starts with an initial approximation $\mathcal{S}$. This initial approximation consists of the ideal point y^I together with the non-negative orthant, i.e., $\mathcal{S} := y^I + \mathbb{R}^p_{\geq 0}$. It proceeds in an iterative fashion by checking the extreme points y^* of the current approximation $\mathcal{S}_i$ for containment in $\mathcal{Q}^+$ using

the point separating oracle. This checking of the extreme points is done repeatedly until for some $\mathcal{S}_{i'}$ the oracle answers with status $inside$ for each extreme point, which means $\mathcal{S}_{i'} = \mathcal{Q}^+$.

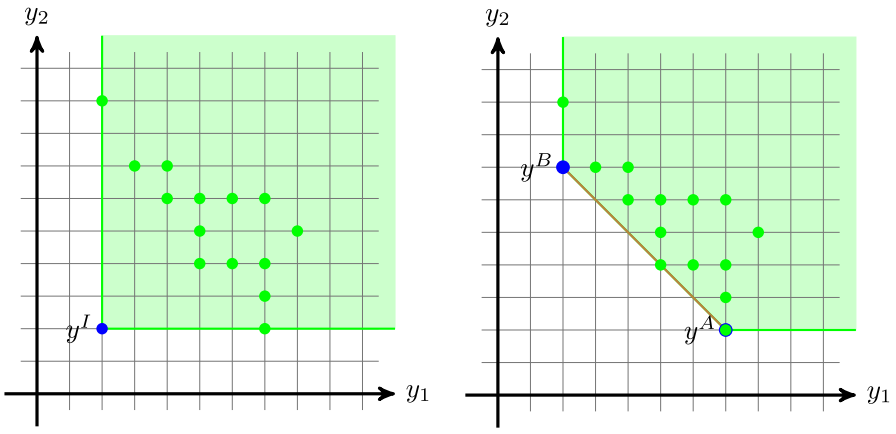
Algorithm 1 Generic OA algorithm for MOMILPs

input: MOMILP instance $inst$, initial approximation $\mathcal{S} := y^I + \mathbb{R}^p_{\geq 0}$ specified by double description (extreme points and facets)
output: $\mathcal{Q}^+$ specified by double description
1: $\mathcal{S}_0 \leftarrow \mathcal{S}$
2: $extremePoints(\mathcal{Q}^+) \leftarrow \emptyset$
3: $i \leftarrow 0$
4: **while** $\exists\, y^* \in extremePoints(\mathcal{S}_i) : y^* \notin extremePoints(\mathcal{Q}^+)$ **do**
5: $(status, w^T y = \alpha) \leftarrow pointSeparatingOracle(y^*, inst)$
6: **if** $status = inside$ **then**
7: $extremePoints(\mathcal{Q}^+) \leftarrow extremePoints(\mathcal{Q}^+) \cup \{y^*\}$
8: **else**
9: $\mathcal{S}_{i+1} \leftarrow \mathcal{S}_i \cap \{y \in \mathbb{R}^p : w^T y \geq \alpha\}$
10: $i \leftarrow i + 1$
11: **end if**
12: **end while**
13: $\mathcal{Q}^+ \leftarrow \mathcal{S}_i$
14: **return** $\mathcal{Q}^+$

An iteration of the OA algorithm is illustrated in Fig. 3. It shows the image of all feasible solutions to an MOILP as green dots. Figure 3a depicts the initial approximation $\mathcal{S}_0$, consisting of the ideal point y^I and $\mathbb{R}^p_{\geq 0}$, in green. Then, the point separating oracle is called and the hyperplane H is returned which separates y^I from $\mathcal{Q}^+$, as shown in Fig. 3b. The new approximation $\mathcal{S}_1$ has two extreme points depicted, i.e., $extremePoints(\mathcal{S}_1) = \{y^A, y^B\}$. An oracle call for the extreme point filled with green, y^A, will return $inside$, since it belongs to $\mathcal{Q}^+$, whereas for the second extreme point y^B, the oracle will produce a new hyperplane, separating y^B from $\mathcal{Q}^+$.

Remark 4 It is easy to see, that at any point of the algorithm, we have $\mathcal{Q}^+ \subseteq \mathcal{S}_i$. Moreover, the algorithm still gives ans OA of $\mathcal{Q}^+$ when the point separating oracle does not separate with respect to $\mathcal{Q}^+$ but to any superset $S(\mathcal{Q}^+) \supset \mathcal{Q}^+$. This means that $\mathcal{Q}^+$ can be replaced with any relaxation of the polyhedron $\mathcal{Q}^+$ in the point separating oracle when the goal is to obtain lower bound sets for multi-objective algorithms needing such lower bound sets.

Remark 5 Let us briefly discuss the key running-time insights of the general OA algorithm. Let us therefore assume, that a point separating oracle is available, produces facets only, and its running time is a constant as a simplification. In this case, it is easy to see that for every extreme point of $\mathcal{Q}^+$ obtained by the algorithm, i.e., the situation of the first if-branch, we ask the point separating oracle once. And for every facet of $\mathcal{Q}^+$ obtained by the algorithm, i.e., the situation of the else-branch, we also ask the oracle once. Moreover, for every facet of $\mathcal{Q}^+$ we have to employ a vertex enumeration algorithm that incurs a running time of $\mathcal{O}(f^{\lfloor \frac{p}{2} \rfloor} + f \log f)$ when there are f facets already present (Chazelle 1993). No facet and no extreme point is found twice.



(a) $\mathcal{S}_0$, $y^* = y^I$ (b) $\mathcal{S}_1$, H, $extremePoints(\mathcal{S}_1)$

Fig. 3 An iteration of the outer approximation algorithm

Thus, we see that the overall running time is polynomial in the number of facets and extreme points of $\mathcal{Q}^+$, and the running time of the point separating oracle, assuming the number of objectives to be a constant.

5 Separation oracles for MOMILP

5.1 Geometric duality for Edgeworth–Pareto hulls of MOMILP

The techniques we use in this paper are heavily influenced by the theory on MOLP. Note that for this subsection only, we forgo the assumption that an ideal point exists. This is because for the geometric duality of MOLP and—as we will see—for the geometric duality of MOMILP, this assumption is not necessary. For MOLP, a duality theory (Heyde and Löhne 2008) exists that is built around a polyhedron $\mathcal{D}$, the lower image, that is geometrically dual to $\mathcal{Q}^+$ (which is the upper image polyhedron in case of MOLP). To define $\mathcal{D}$, let for $w \in \mathbb{R}^{p-1}$, $\lambda(w) := \left(w_1, \dots, w_{p-1}, 1 - \sum_{i=1}^{p-1} w_i\right) \in \mathbb{R}^p$. For a feasible MOLP with $A \in \mathbb{Q}^{m\times n}$, $b \in \mathbb{Q}^m$, and $C \in \mathbb{Q}^{p\times n}$, the *lower image* is defined as

$$\mathcal{D} := \Bigg\{ \left(w_1, \dots, w_{p-1}, \alpha\right) \in \mathbb{R}^p : \\ w_i \geq 0, \sum_{i=1}^{p-1} w_i \leq 1, u^\mathsf{T} A = \lambda(w)^\mathsf{T} C, u \in \mathbb{R}^m, \alpha \leq b^\mathsf{T} u \Bigg\}.$$

The following characterization of $\mathcal{D}$ was already given by Heyde and Löhne (2008), we state it here for the sake of completeness. It characterizes the lower image inde-

pendently of A, b, and C but only with respect to the upper image polyhedron $\mathcal{Q}^+$ in the following way.

Proposition 6 *Let a feasible MOLP be given. Let $\mathcal{Q}^+$ be defined as above. Then*

$$\mathcal{D} = \left\{ (w_1, \ldots, w_{p-1}, \alpha) \in \mathbb{R}^p : \right.$$
$$\left. w_i \geq 0, \sum_{i=1}^{p-1} w_i \leq 1, \forall y \in \mathcal{Q}^+ : \alpha \leq \lambda(w)^\mathsf{T} y \right\}$$

holds.

Proof We assume the MOLP to be given by $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, and $C \in \mathbb{Q}^{p \times n}$. Let $\widehat{z} = (\widehat{w}_1, \ldots, \widehat{w}_{p-1}, \widehat{\alpha}) \in \mathcal{D}$. By the definition of $\mathcal{D}$, for $\widehat{z}$ there exists a $\widehat{u} \in \mathbb{R}^m$ with $\widehat{u}^\mathsf{T} A = \lambda(\widehat{w})^\mathsf{T} C$ and $\alpha \leq b^\mathsf{T} \widehat{u}$. Consequently, $\widehat{u}$ is feasible for the dual of (WSUM) of the given MOLP with weight $\lambda(\widehat{w})$. Moreover, $b^\mathsf{T} \widehat{u}$ is the dual objective value of $\widehat{u}$ and is asserted to be at least $\widehat{\alpha}$. By weak duality, we have $\widehat{\alpha} \leq b^\mathsf{T} \widehat{u} \leq \lambda(\widehat{w})^\mathsf{T} y$ for every feasible $y \in \mathcal{Q}$. Observe that for any $v, v' \in \mathbb{R}^p$, we have $v \leq v' \implies w^\mathsf{T} v \leq w^\mathsf{T} v'$ for every $w \in \mathbb{R}^p$ with $w \geq 0$. Hence, $\widehat{\alpha} \leq \lambda(\widehat{w})^\mathsf{T} y$ is also true for every $y \in \mathcal{Q}^+$.

Let now $\widehat{z} = (\widehat{w}_1, \ldots, \widehat{w}_{p-1}, \widehat{\alpha})$ be a member of the right set. As $\widehat{\alpha} \leq \lambda(\widehat{w})^\mathsf{T} y$ for every $y \in \mathcal{Q}^+$, this is also true for every $y \in \mathcal{Q}$ that minimizes $\lambda(\widehat{w})^\mathsf{T} y$ over $\mathcal{Q}$. Let $\widehat{y}$ be such a minimizer. Note that $\lambda(\widehat{w})^\mathsf{T} \widehat{y}$ is thus the optimal value of (WSUM) with weight $\lambda(\widehat{w})$. Let $\widehat{u}$ be an optimal solution to the dual of (WSUM). Then we have $\widehat{\alpha} \leq \lambda(\widehat{w})^\mathsf{T} \widehat{y} = b^\mathsf{T} \widehat{u}$ by strong duality. Using $\widehat{u}$ as u, we see that $\widehat{z} \in \mathcal{D}$.

□

Note that this characterization of a lower image is purely geometric. Our goal is to apply the geometric duality theory of MOLP to Edgeworth–Pareto hulls of MOMILPs. To this end, we show that for every MOMILP, there is a MOLP so that the upper image of the latter coincides with the Edgeworth–Pareto hull of the former.

Lemma 7 *Let a feasible MOMILP be given. Let the Edgeworth–Pareto hull of this MOMILP be $\mathcal{Q}^+ \subseteq \mathbb{R}^p$. There exists a MOLP with upper image $\overline{\mathcal{Q}^+}$ such that $\mathcal{Q}^+ = \overline{\mathcal{Q}^+}$.*

Proof We construct the MOLP in the following way: As $\mathcal{Q}^+$ is finitely generated and thus polyhedral, there exists a finite hyperplane representation $\{A'x \geq b'\}$ of it. Let C' be the $p \times p$ identity matrix. Recall that rec $\mathcal{Q}^+ = \mathbb{R}^p_{\geq}$. Then the problem $\min\{C'x : A'x \geq b'\}$ is an MOLP with upper image $\mathcal{Q}^+$. □

This lemma justifies the definition of a generalized lower image for MOMILP.

Definition 8 Let $\mathcal{Q}^+$ be the Edgeworth–Pareto hull of an instance of a MOMILP. The *generalized lower image* of the MOMILP is defined as:

$$\mathcal{D} := \left\{ (w_1, \ldots, w_{p-1}, \alpha) \in \mathbb{R}^p : \right.$$

$$w_i \geq 0, \sum_{i=1}^{p-1} w_i \leq 1, \forall y \in \mathcal{Q}^+ : \alpha \leq \lambda(w)^\mathsf{T} y \Big\}$$

By replacing the MOMILP by the corresponding MOLP, we can apply the geometric duality theory of MOLP to Edgeworth–Pareto hulls of a MOMILP. In particular, there exists an inclusion reversing one-to-one map from the k-dimensional faces of $\mathcal{Q}^+$ to the $\mathcal{K}$-maximal $(p-k-1)$-dimensional faces of $\mathcal{D}$. Where $\mathcal{K}$-maximal means the faces are maximal for the ordering $x \leq_{\mathcal{K}} y \iff x_p \leq y_p \wedge \forall i \in [p-1]: x_i = y_i$ for $x, y \in \mathcal{D}$. Observe that $\mathcal{K}$-maximality is only needed for faces of dimension larger than 0, as every extreme point of $\mathcal{D}$ is $\mathcal{K}$-maximal. This can be easily seen as $(0, \ldots, 0, -1)$ is an extreme ray of $\mathcal{D}$. From this, we obtain the following important corollaries.

Corollary 9 *Let a MOMILP be given. Let $\mathcal{Q}^+$ be its Edgeworth–Pareto hull and $\mathcal{D}$ its generalized lower image. For every extreme point $(\widehat{w}_1, \ldots, \widehat{w}_{p-1}, \widehat{\alpha})$ of $\mathcal{D}$, the hyperplane $\left\{y \in \mathbb{R}^p : \lambda(\widehat{w})^\mathsf{T} y = \widehat{\alpha}\right\}$ is facet supporting to $\mathcal{Q}^+$.*

Corollary 10 *For every feasible MOMILP, $\mathcal{D}$ is finitely generated.*

Note that the definition of the generalized lower image is very similar to the lifted weight space as defined in Borndörfer et al. (2016). But our generalized lower image applies to the more general problem class of MOMILP. Moreover, our characterization allows to apply the geometric duality theory to this generalized lower image.

5.2 A first point separating oracle

Let w_i be the coefficient of objective $i, i = 1, \ldots, p$ in the desired hyperplane H, and α be the right-hand-side of H. A point separating oracle can then be defined as follows.

$$\begin{aligned}
(\text{Sep-}y^*) \quad & \min\ (y^*)^\mathsf{T} w - \alpha & & \\
& s.t.\ y^\mathsf{T} w - \alpha \geq 0 & \forall y \in \mathcal{Q} \quad & \text{(FEAS)} \\
& \sum_{i=1,\ldots,p} w_i = 1 & & \text{(WSUM1)} \\
& w_i \geq 0 & i = 1, \ldots, p \quad & \text{(WSUM2)} \\
& \alpha \in \mathbb{R} & &
\end{aligned}$$

The LP (Sep-y^*) encodes that (w, α) should induce a separating hyperplane by enforcing that all $y \in \mathcal{Q}$ should be on the non-negative side of the hyperplane using constraints (FEAS). Constraint (WSUM1) is a normalisation constraint for the coefficients of the obtained hyperplane. It will be used in the proofs later on. We note that the LP (Sep-y^*) shares similarities with the LPs used in local cut separation for single-objective mixed-integer linear programming (see, e.g., Applegate et al. (2001), Chvátal et al. (2013) for general references on local cut separation). In particular, it has the same structure as the LP used in Avella et al. (2010) for separating (local) cuts for the single-objective generalized assignment problem.

If the objective function value of the optimal solution $(\widehat{w}, \widehat{\alpha})$ of (Sep-y^*) is negative for a given y^*, we get that $y^* \notin \mathcal{Q}^+$ and $(\widehat{w}, \widehat{\alpha})$ gives the coefficients of the corresponding separating hyperplane. In the following, we first show that the LP above models a separating hyperplane that is facet supporting to $\mathcal{Q}^+$ and then we show that an algorithm exists to compute an optimal solution to it.

Let us first prove that an optimal solution to this LP always encodes a separating hyperplane.

Lemma 11 *Let $y^* \in \mathbb{R}^p$ and $(\widehat{w}, \widehat{\alpha})$ be an optimal solution to (Sep-y^*). We have $(y^*)^\mathsf{T}\widehat{w} - \widehat{\alpha} < 0$ if and only if $y^* \notin \mathcal{Q}^+$.*

Proof If $(y^*)^\mathsf{T}\widehat{w} - \widehat{\alpha} < 0$, then $\widehat{w}^\mathsf{T}(y^*) < \widehat{\alpha}$ and $\widehat{w}^\mathsf{T} y \geq \widehat{\alpha}$ for all $y \in \mathcal{Q}$ by constraints (FEAS).

Since $w_i \geq 0$, this is also true for all $y \in \mathcal{Q}^+$. In other words, $H^{\widehat{w},\widehat{\alpha}} := \left\{y \in \mathbb{R}^p : \widehat{w}^T y = \widehat{\alpha}\right\}$ separates y^* from $\mathcal{Q}^+$ and thus $y^* \notin \mathcal{Q}^+$. If y^* is not in $\mathcal{Q}^+$, then by the hyperplane separation theorem, there exists (at least) a hyperplane H^{w^*,α^*}, separating y^* from $\mathcal{Q}^+$, i.e., $(w^*)^\mathsf{T}(y^*) < \alpha^*$ and $(w^*)^\mathsf{T} y \geq \alpha^*$ for all $y \in \mathcal{Q}^+$. Moreover, we can assume w^* to be normalized, i.e., $\sum_{i=1}^p w_i^* = 1$. As rec $\mathcal{Q}^+ = \mathbb{R}^p_{\geq 0}$, we can also assume all w_i^* to be at least 0. Consequently, (w^*, α^*) is a feasible solution to (Sep-y^*) and its value $(y^*)^\mathsf{T} w^* - \alpha^*$ is strictly smaller than 0 and an upper bound on the optimal value. □

Let us now show why the hyperplanes are facet supporting. In the following lemma we make use of the fact that we can always assume a theoretical or practical (single-objective) LP solver to deliver an optimal extreme point solution in decision space to a (single-objective) LP if one exists. For the theoretical reasoning, see Grötschel et al. (1993).

Lemma 12 *For every optimal extreme point solution $(\widehat{w}, \widehat{\alpha})$ of (Sep-y^*), we have*

$$H^{\widehat{w},\widehat{\alpha}} := \left\{y \in \mathbb{R}^p : \widehat{w}^\mathsf{T} y = \widehat{\alpha}\right\}$$

is facet supporting for $\mathcal{Q}^+$. Moreover, the feasible set of (Sep-y^*) is finitely generated.

Proof We show that the feasible set of (Sep-y^*) is isomorphic to $\mathcal{D}$. Then both claims follow immediately from Corollaries 9 and 10.

To this end, we see that for every feasible solution $(w_1, \ldots, w_p, \alpha)$ of (Sep-y^*), the constraints (FEAS) can also be written as $y^\mathsf{T} w \geq \alpha$ for all $y \in \mathcal{Q}^+$ as in $\mathcal{D}$. Let therefore $y \in \mathcal{Q}^+$. There is $y' \in \mathcal{Q}$ with $y' \leq y$. If $y \in \mathcal{Q}$, then this is clear. If $y \notin \mathcal{Q}$, the existence of such a y' follows from the definition of $\mathcal{Q}^+$. With (FEAS), we have $y'^\mathsf{T} w \geq y^\mathsf{T} w \geq \alpha$.

With $w_p = 1 - \sum_{i=1}^{p-1} w_i$, it is now easy to construct an affine isomorphism between the feasible set of (Sep-y^*) and $\mathcal{D}$. □

Note that as mentioned before, $\mathcal{Q}$ is of course not known when calling the point separating oracle. Moreover, (Sep-y^*) has an infinite number of inequalities. To deal with this obstacle and solve (Sep-y^*), we use the ellipsoid method

in theory (Grötschel et al. 1993). To do so, we observe that the constraints (FEAS) can be separated by using a weighted-sum scalarization of the original MOMILP: Given a solution $(\widehat{w}, \widehat{\alpha})$, there exists a violated constraint (FEAS), if and only if $\min\left\{\widehat{w}^\mathsf{T} y : y \in \mathcal{Q}\right\} = \min\left\{\widehat{w}^\mathsf{T} C x : x \in \mathcal{X}\right\} < \widehat{\alpha}$. The ellipsoid method thus guarantees that (Sep-y^*) can be solved correctly in finite time.

Remark 13 While the ellipsoid method is a strong theoretical tool, it is not practically competitive. Hence for the practical implementation, we make use of the traditional cutting-plane approach. The ellipsoid method, however, can also give us a guarantee on the number of oracle calls, the cutting-plane approach cannot. See Sect. 5.4 for more details on the running time analysis and Sect. 6.2 for more details on the implementation.

Concluding this section, we arrive at the following theorem.

Theorem 14 *Computing an optimal solution to (Sep-y^*) using the ellipsoid method is a point separating oracle that provides us with facet supporting inequalities.*

5.3 A target cut-like separation oracle

Another separation oracle can be defined as follows. This oracle follows the *target-cut* paradigm introduced in Buchheim et al. (2008) for single-objective mixed-integer linear programming. In the description below, we assume that all points encountered in the separation process are ≥ 1 (component-wise). This is without loss of generality, as we can always add a large constant to all points. In Sect. 6.2 we describe how we do this in our implementation. The oracle is given by the following LP.

$$\begin{aligned} \text{(TSep-}y^*\text{)} \quad \min\ & (y^*)^\mathsf{T} w & & \\ s.t.\ & y^\mathsf{T} w \geq 1 & \forall y \in \mathcal{Q} \quad & \text{(TFEAS)} \\ & w_i \geq 0 & i = 1, \ldots, p \quad & \text{(TWSUM)} \end{aligned}$$

If an optimal solution $\widehat{w}$ of (TSep-y^*) has an objective value smaller than 1, $\widehat{w}^\mathsf{T} y \geq 1$ gives a separating hyperplane. Otherwise, we return *inside*. Similar to constraints (FEAS) of (Sep-y^*), we propose to solve (TSep-y^*) using a cutting-plane approach, where constraints (TFEAS) are separated by solving weighted-sum problems.

The reason why this algorithm is a point separating oracle is analogous to Lemma 11: We see that for any feasible solution $\widehat{w}$ with objective value $(y^*)^\mathsf{T}\widehat{w} < 1$, the hyperplane $\{x \in \mathbb{R}^p : w^\mathsf{T} x = 1\}$ again separates y^* from $\mathcal{Q}^+$, as long as $\mathcal{Q} \subseteq \mathbb{R}^p_{\geq 0}$. Moreover, in case the minimum value is at least 1, then there is no separating hyperplane, hence $y^* \in \mathcal{Q}^+$.

To prove that an optimal extreme point solution corresponds in fact to a facet supporting inequality, we observe that the feasible set of (TSep-y^*) is a polar dual polyhedron to $\mathcal{Q}^+$.

Theorem 15 *Computing an optimal solution to (TSep-y^*) using the ellipsoid method is a point separating oracle that provides us with facet supporting inequalities.*

5.4 Theoretical running time

Let us first analyze the most general version of the OA algorithm. Using the methodology of Bökler (2018), we observe the following corollary:

Corollary 16 *The OA algorithm with the point separating oracles from Sect. 4 produces the next facet in time $\mathcal{O}\left(k^{\lfloor \frac{p}{2} \rfloor}(T_O + k \log k)\right)$, where T_O is the running time of the point separating oracle and $k \in \mathbb{N}$ is the number of facets already computed.*

Both our new point separating oracles show that a point separating oracle can be implemented by solving a sequence of weighted-sum scalarizations. The major advantage of solving weighted-sum scalarizations over other known methods to implement the oracle model is that no new constraints are added to the oracle problem and thus the structural properties of the feasible sets remain untouched. It remains to be shown, however, that the number of weighted-sum scalarizations that need to be solved does not grow too fast. The problems (Sep-y^*) and (TSep-y^*) are linear programming problems and thus only weakly polynomial-time algorithms are known to solve them. Consequently, we have to take a closer look at the numbers in the definition of the feasible sets of (Sep-y^*) and (TSep-y^*). In both LPs, the only crucial numbers come from the constraints (FEAS) and (TFEAS). In both cases due to geometric or polar duality, the subset of necessary constraints come from extreme points of $\mathcal{Q}^+$ as they are exactly the constraints that define facets of the respective feasible sets.

For MOMILPs in general, the encoding length of the numbers defining extreme points of $\mathcal{Q}^+$ can be bounded by a polynomial in the input size: Let us consider the feasible set $\mathcal{X}$ of a given MOMILP instance. It is well known that the encoding length of extreme points of conv $\mathcal{X}$ are bounded by a polynomial in the size of the encoding of A and b (Conforti et al. 2014). The encoding length of the image of each such feasible extreme point under the objective function is thus also polynomially bounded in the encoding length of A, b, and C, since the image can be computed by a matrix–vector product. For a rigorous proof of the latter statement, see Bökler (2018). We can thus derive a general bound on the number of oracle calls in the ellipsoid method based on the encoding length of the input MOMILP.

Lemma 17 *Let a MOMILP be given. An optimal extreme point of (Sep-y^*) and (TSep-y^*) can be computed with a number of weighted-sum calls that grows at most polynomially in p and the encoding length of A, b, and C.*

Proof Using the methodology of Grötschel et al. (1993), we compute an optimal extreme point solution to the LPs (Sep-y^*) or (TSep-y^*) in polynomial time by using the ellipsoid method and employing a rounding mechanism to reach an optimal extreme point solution in decision space. The ellipsoid method has a running time polynomial in the number of variables ($p+1$ or p) and the encoding length of extreme points of the input polyhedron, i.e., the feasible set of (Sep-y^*) or (TSep-y^*). The encoding length of the extreme points of the feasible sets of (Sep-y^*) and (TSep-y^*) are polynomially

bounded in the encoding length of their facets. The facets of the feasible set of (Sep-y^*) and (TSep-y^*), in turn, correspond to the extreme points of $\mathcal{Q}^+$. With the above considerations on the extreme points of $\mathcal{Q}^+$, these encoding lengths are polynomial in the encoding length of A, b, and C.

The separation oracle in the invocation of the ellipsoid method is implemented by a solver for the weighted-sum problem. The weighted-sum solver is hence called only polynomially many times. □

For the special case of polynomial-time solvable weighted-sum scalarizations, we arrive at an even stronger result.

Theorem 18 *If the weighted-sum problem for a given MOMILP is polynomial-time solvable, then the facets of $\mathcal{Q}^+$ can be computed with incremental-polynomial delay.*

Proof This result follows directly, since the separation-oracle calls in the ellipsoid method are now polynomial-time solvable and thus the whole point separating oracle takes only polynomial time in the input encoding. □

This theorem complements the work by Bökler and Mutzel (2015), where the same result was shown for linear combinatorial optimization problems and extreme point of $\mathcal{Q}^+$ instead of facets. It is easy to see that their result can also be applied to MOMILPs.

6 Computational experiments

The proposed OA algorithms are implemented in Python using CPLEX 12.10 as LP/mixed integer linear programming-solver and the `parma polyhedral library (ppl)` (Bagnara et al. 2008) for vertex enumeration. In this section, we first describe the instances used in our computational experiments in Sect. 6.1, followed by a discussion of implementation details in Sect. 6.2 and results in Sect. 6.3. The runs were made on a single core of an Intel Xeon E5-2670v2 machine with 2.5 GHz and 3 GB of RAM, and we used a time limit of 600 s.

6.1 Instances

We use instance sets from the literature. The sets are as follows.

- `AP-MOO`: These are multi-objective assignment instances from the multi-objective optimization library (`moolibrary`),[1] with three objectives. The number of agents and tasks is identical and ranges from 5 to 40, in increments of 5. The objective function coefficients are random integers in the range [1, 20]. There are 100 instances in the set. They were proposed in Kirlik and Sayın (2014); similar instances were also used in experiments for IA algorithms in Özpeynirci and Köksalan (2010), Przybylski et al. (2010, 2019).
- `KP-MOO`: These are multi-objective knapsack instances from the `moolibrary` with three to five objectives. Both profits and weights are random integers from

[1] Available at http://home.ku.edu.tr/~moolibrary/.

the interval [1, 1000]. The budget is calculated as half the total weight of all items, rounded up to the next integer. There are 160 instances in the set: 100 instances with three objectives (10–100 items), 40 with four (10–40 items), and 20 with five (10–20 items). Like the set `AP-MOO`, these instances are proposed in Kirlik and Sayın (2014) and similar instances are also used in experiments for IA algorithms in Özpeynirci and Köksalan (2010), Przybylski et al. (2010, 2019).

6.2 Implementation details

6.2.1 Instance transformation and offset calculation

We first transform the instances to minimization form by flipping the objective coefficients for all objectives which are maximization. We then compute y^I by solving $\min_{y \in \mathcal{Q}} e_i^T y$ for unit-vector e_i. If the value of y^I is negative for an objective j, we add the constant $-y_j^I + 1$ to the respective objective to ensure that all points encountered in the separation process are component-wise ≥ 1 as needed for (TSep-y^*). Moreover, instead of using the value of one on the right-hand-side of constraints (TWSUM), we fix the value to the sum of the components of the ideal point (after adding the constant $-y_j^I + 1$ to objectives j with negative value). This proved to be more numerically stable in preliminary computations.

6.2.2 Separation

For the initial approximation $\mathcal{S}$, the ideal point y^I can be obtained by solving $\min_{y \in \mathcal{Q}} e_i^T y$ for unit-vector e_i. In our implementation of the oracles, we use a cutting-plane approach which is illustrated in Algorithm 2 for the first oracle, the second oracle is implemented in a similar fashion.

We initialize the separation LPs by adding constraints (FEAS), resp., (TFEAS) induced by the solutions obtained for the p problems used for calculating the ideal point. When separating constraints (FEAS), resp., (TFEAS), we use tolerance ε for checking violation. Once a constraint (FEAS), resp., (TFEAS) is added to the separation LP, we leave it there for the remainder of the algorithm (i.e., for all subsequent separation oracle calls). In the separation LPs, the `numerical emphasis` parameter of CPLEX is turned on, and the feasiblity and optimality tolerances are set to 1e-9, i.e., the most accurate value possible. When checking the result of the separation oracle (i.e., the objective value of the separation LP against zero, resp., one), we also use tolerance ε.

We use `ppl` with integer numbers as input, as using fractional numbers leads to (more) numerical instabilities. In order to do so, we scale each (w, α) obtained from the separation oracle by 10^9 and take the integer part of the obtained number. When checking if a point $y^* \in extremePoints(\mathcal{S}_i)$ is in $\mathcal{Q}^+$, we proceed as follows: When a point y^* is within tolerance value ε to a point in $y' \in extremePoints(\mathcal{Q}^+)$ for each coordinate, we consider $y^* = y'$ and do not call the separation oracle for it. For each $\mathcal{S}_i$, we check the points in the order in which they are made available by `ppl`. Moreover, when updating the outer approximations, we use a slightly different strategy

Algorithm 2 Cutting-plane algorithm for the first point separating oracle

input: MOMILP instance $inst$, point $y^* \in \mathbb{R}^p$
output: $(status, H)$
1: $LP \leftarrow$ (Sep-y^*) without constraints (FEAS) ▷ add some constraints (FEAS) to avoid unboundedness of LP in the first iteration of the cutting-plane approach
2: $LP \leftarrow LP \cup$ constraints (FEAS) for the solutions y obtained when calculating the ideal point y^I
3: $violated \leftarrow true$
4: **do**
5: $violated \leftarrow false$
6: $(\widehat{w}, \widehat{\alpha}) \leftarrow$ optimal solution obtained when solving LP
7: $\widehat{y} \leftarrow$ optimal solution obtained when solving $\min \left\{ \widehat{w}^\mathsf{T} Cx : x \in \mathcal{X} \right\}$
8: **if** $\widehat{w}^\mathsf{T} \widehat{y} < \widehat{\alpha}$ **then**
9: $violated \leftarrow true$
10: $LP \leftarrow LP \cup \left\{ \widehat{y}^\mathsf{T} w \geq \alpha \right\}$
11: **end if**
12: **while** $violated$
13: **if** $\widehat{w}^\mathsf{T} y^* - \widehat{\alpha} \geq 0$ **then**
14: **return** $(inside, \emptyset)$
15: **else**
16: **return** $(outside, \widehat{w}^\mathsf{T} y^* = \widehat{\alpha})$
17: **end if**

compared to the outline in Algorithm 1: We do not immediately calculate $\mathcal{S}_i \cap H$ and move to $\mathcal{S}_{i+1}$ once we discovered a $y^* \notin extremePoints(\mathcal{Q}^+)$; instead, we check all $y^* \in extremePoints(\mathcal{S}_i)$, collect the obtained separating hyperplanes, and then add them all at the same time to obtain the next outer approximation. This approach turned out to be faster and more numerically stable in preliminary computations. Finally, we also add every point found when separating (FEAS), resp., (TFEAS) to $extremePoints(\mathcal{Q}^+)$, as these points are obtained from the solution of a weighted-sum problem. This means that these points are supported non-dominated points and thus they can be extreme points of $\mathcal{Q}^+$. Note that by doing this it can happen that $extremePoints(\mathcal{Q}^+)$ contains some points which are not extreme points, but this does not matter for the correctness of the algorithm.

6.3 Results

6.3.1 Numerical accuracy of our approach and the influence of ε

The focus in the implementation of our algorithms is to find a good balance between numerical accuracy and speed. We note that IA algorithms can suffer from numerical issues, for example in Przybylski et al. (2019) different versions of the same IA algorithm are tested, and some versions cannot find the complete set of extremes points for some instances. In our computational study, we use the open-source IA solver `PolySCIP` (Borndörfer et al. 2016; Maher et al. 2017) for comparison. As experiments show (see below), there are some instances were `PolySCIP` does not find all extreme points; moreover, we observed in `PolySCIP`'s output that in some cases it also reports some weakly-dominated points as part of the set of extremes points. In OA, numerical issues can lead to missing cut-off points, which do not belong to $\mathcal{Q}^+$.

However, even if we miss some of these points, the obtained solution is still a valid outer approximation.

To investigate the influence of different values of ε on the performance and accuracy of our approach, we first consider the instance set AP-MOO. Since the assignment problem has a totally unimodular constraint matrix, it can be solved as a linear programming problem and thus we can obtain the extreme points of $\mathcal{Q}^+$ by using the multi-objective LP solver bensolve. Table 1 shows the number of extreme points of $\mathcal{Q}^+$ obtained by our OA with both oracles (columns $|\tilde{\mathcal{Q}}^+|$), and the number of extreme points when these points are rounded to the next integer (columns $|\mathcal{Q}^+|$, recall that AP-MOO has integer coefficients, thus all extreme points have integer values). We considered $\varepsilon = 1e-3$ and $\varepsilon = 1e-5$ for both oracles. We also report the number of extreme points obtained by bensolve and PolySCIP and the run time of the methods (column $t[s]$). The results are aggregated by instance size. To allow for a meaningful comparison, we only report results for instance sizes were all approaches terminate within the time limit for all instances of this size.

The results show that for instances with size up to 30, all methods give a consistent number of extreme points. For instances of size 35, the size of $|\mathcal{Q}^+|$ obtained by the OA is consistent with bensolve considering both oracles when $\varepsilon = 1e-5$. For instances of size 40, the number of extreme points obtained by the OA and by bensolve becomes inconsistent, but only by a small amount. This is not surprising as with larger instances size, the size of $|\mathcal{Q}^+|$ also grows, and thus the potential for numerical inaccuracies increases as well. We observe that the number of non-rounded extreme points ($|\tilde{\mathcal{Q}}^+|$) also stays fairly consistent for smaller-sized instances, and for all sizes is about three times as large as the number of points after rounding. Regarding the run time, using $\varepsilon = 1e{-}3$ is about twice as fast as using $\varepsilon = 1e{-}5$.

6.3.2 Results for instance set KP-MOO

Table 2 shows the results for instance set KP-MOO obtained by our algorithms and by PolySCIP. Our algorithms are run with $\varepsilon = 1e{-}5$. In this table, we give the number of rounded extreme points (columns $|\mathcal{Q}^+|$), the number of facets (columns #fac) and the runtime (columns t [s]). We see that the number of points is quite similar for all three methods, however, as the instances become larger for $p = 3$ and for $p = 4, 5$ in general, the numbers differ. This is not unexpected, as both larger instances and more objectives could have a negative effect on numerical accuracy. In particular, as for a larger number of objectives, the number of facets is growing considerably. We observe that for the number of facets, the difference is larger compared to the number of points. This could be explained by the fact that $|\mathcal{Q}^+|$ is the rounded set of points, which exploits that the instance coefficients are integer. Thus, numerical inaccuracies are corrected by the rounding, e.g., multiple slightly fractional points are likely to be rounded to the same integer point. The runtime-performance of both separation oracles is quite similar.

Table 1 Results for instances AP-MOO aggregated by size

Size	(Sep-y^*)						(TSep-y^*)						bensolve		PolySCIP	
	$\varepsilon = 1e-3$			$\varepsilon = 1e-5$			$\varepsilon = 1e-3$			$\varepsilon = 1e-5$						
	$\lvert\tilde{\mathcal{Q}}^+\rvert$	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\tilde{\mathcal{Q}}^+\rvert$	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\tilde{\mathcal{Q}}^+\rvert$	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\tilde{\mathcal{Q}}^+\rvert$	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\mathcal{Q}^+\rvert$	t [s]
5	19.0	7.5	0.7	19.0	7.5	0.6	18.7	7.5	0.1	18.7	7.5	0.1	7.5	0.0	7.5	1.0
10	117.4	39.0	2.0	117.4	39.0	1.8	117.4	39.0	0.8	117.4	39.0	0.9	39.0	0.5	38.1	2.3
15	258.7	83.1	3.5	258.7	83.1	3.9	258.7	83.1	3.0	258.7	83.1	3.6	83.1	0.4	78.5	6.3
20	495.3	161.3	10.8	495.3	161.3	12.6	495.3	161.3	10.2	495.3	161.3	13.6	161.3	0.4	153.5	11.1
25	809.1	253.1	26.9	809.1	253.1	36.6	808.7	253.1	27.2	809.1	253.1	39.2	253.1	0.8	223.3	17.6
30	1183.0	379.4	62.6	1183.2	379.4	95.7	1182.4	379.5	63.1	1183.2	379.4	101.6	379.4	1.7	348.1	35.5
35	1579.7	501.2	121.7	1581.1	501.4	202.0	1576.9	500.9	123.1	1581.1	501.4	213.3	501.4	3.2	458.0	65.9
40	2170.0	698.9	244.8	2160.4	700.8	437.2	2167.8	698.4	247.5	2164.6	700.5	451.9	699.1	5.8	631.9	122.2

Table 2 Results for instances KP−MOO with $p = 3, 4, 5$ (separated by horizontal lines) aggregated by number of items

Size	(Sep-y^*)			(TSep-y^*)			PolySCIP	
	#fac	$\lvert\mathcal{Q}^+\rvert$	t [s]	#fac	$\lvert\mathcal{Q}^+\rvert$	t [s]	$\lvert\mathcal{Q}^+\rvert$	t [s]
10	11.1	5.0	0.2	11.1	5.0	0.1	5.0	7.8
20	30.2	14.1	0.4	29.9	14.1	0.3	14.1	4.4
30	56.4	26.2	0.9	56.6	26.2	0.8	26.1	3.5
40	77.6	36.3	1.5	78.0	36.3	1.4	35.9	3.6
50	105.8	48.4	2.2	106.1	48.6	2.0	48.7	2.5
60	150.9	69.5	3.7	150.9	69.5	3.2	69.7	3.0
70	201.4	92.5	5.7	203.2	92.7	5.0	92.5	2.4
80	252.4	114.4	8.1	249.7	114.5	6.8	114.3	3.1
90	311.2	141.9	11.4	309.7	141.6	9.1	141.8	3.5
100	388.9	177.7	16.2	391.6	177.7	12.9	176.7	7.4
10	27.2	6.6	0.2	26.5	6.6	0.2	6.6	0.7
20	169.7	30.2	2.2	170.0	30.2	2.0	30.0	1.1
30	378.7	61.2	7.2	375.6	61.1	5.9	60.2	1.4
40	908.3	136.7	25.5	915.1	136.9	19.2	133.5	2.5
10	108.2	10.2	0.9	109.3	10.1	0.9	10.0	0.5
20	661.4	38.9	15.1	650.7	38.9	11.6	38.1	1.0

7 Conclusion

We present the first outer-approximation algorithm to compute the extreme points and facets for multi-objective mixed-integer linear programming problems (MOMILPs). The algorithm is based on an extension of the concept of geometric duality for MOLPs to MOMILPs. We show that the number of weighted-sum mixed-integer oracles needed to compute these extreme points and facets can be bounded polynomially by the size of the output and the input for the MOMILPs. Moreover, for MOMILPs with polynomial-time computable weighted-sum scalarizations, the facets of the Edgeworth–Pareto hull can be computed with incremental polynomial delay. We provide a computational study on instances from the literature. In this study, we investigate the numerical accuracy of our method and also include a comparison with PolySCIP, which is an inner approximation algorithm for MOMILPs. We conjecture that existing and future multi-objective B&B methods can benefit from partial bound set computation which now is possible with our proposed method. Future work should address general strategies for dealing with numerical instabilities and define appropriate threshold values especially for the general mixed integer case. It should also investigate the efficacy of this approach in practical B&B algorithms. Investigating possible extensions of our approach to non-linear settings could also be an interesting avenue for further research. Finally, studying potential consequences and

implications of our theoretical results for related areas such as parametric integer linear programming could also be an interesting topic.

Funding Open access funding provided by Johannes Kepler University Linz. This research was funded in whole, or in part, by the Austrian Science Fund (FWF) [P 31366, 35160-N]. The research was also supported by the Linz Institute of Technology (Project LIT-2019-7-YOU-211) and the JKU Business School. For the purpose of open access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

Data availability The datasets generated during and/or analyzed during the current study are available in the web pages http://home.ku.edu.tr/~moolibrary/.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

Adelgren N, Gupte A (2022) Branch-and-bound for biobjective mixed-integer linear programming. INFORMS J Comput 34(2):909–933

Aneja YP, Nair KP (1979) Bicriteria transportation problem. Manag Sci 25(1):73–78

Applegate D, Bixby R, Chvátal V, Cook W (2001) TSP cuts which do not conform to the template paradigm. In: Computational combinatorial optimization. Springer, pp 261–303

Avella P, Boccia M, Vasilyev I (2010) A computational study of exact knapsack separation for the generalized assignment problem. Comput Optim Appl 45:543–555

Bagnara R, Hill PM, Zaffanella E (2008) The Parma Polyhedra Library: toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. Sci Comput Program 72(1–2):3–21

Benson HP (1998) An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. J Glob Optim 13(1):1–24

Bökler F (2018) Output-sensitive complexity for multiobjective combinatorial optimization with an application to the multiobjective shortest path problem. Ph.D. thesis, TU Dortmund University

Bökler F, Ehrgott M, Morris C, Mutzel P (2017) Output-sensitive complexity for multiobjective combinatorial optimization. J Multi-Criteria Decis Anal 24(1–2):25–36

Bökler F, Mutzel P (2015) Output-sensitive algorithms for enumerating the extreme nondominated points of multiobjective combinatorial optimization problems. In: Algorithms-ESA 2015. Springer, pp 288–299

Boland N, Charkhgard H, Savelsbergh M (2015) A criterion space search algorithm for biobjective mixed integer programming: the triangle splitting method. INFORMS J Comput 27(4):597–618

Borndörfer R, Schenker S, Skutella M, Strunk T (2016) PolySCIP. In: International congress on mathematical software. Springer, pp 259–264

Boyd EA (1993) Generating Fenchel cutting planes for knapsack polyhedra. SIAM J Optim 3(4):734–750

Boyd EA (1994) Fenchel cutting planes for integer programs. Oper Res 42(1):53–64

Braekers K, Hartl RF, Parragh SN, Tricoire F (2016) A bi-objective home care scheduling problem: analyzing the trade-off between costs and client inconvenience. Eur J Oper Res 248:428–443

Buchheim C, Liers F, Oswald M (2008) Local cuts revisited. Oper Res Lett 36(4):430–433
Chankong V, Haimes YY (2008) Multiobjective decision making: theory and methodology. Courier Dover Publications
Chazelle B (1993) An optimal convex hull algorithm in any fixed dimension. Discrete Comput Geom 10:377–409
Chinchuluun A, Pardalos PM (2007) A survey of recent developments in multiobjective optimization. Ann Oper Res 154(1):29–50
Chvátal V, Cook W, Espinoza D (2013) Local cuts for mixed-integer programming. Math Program Comput 5(2):171–200
Cohon JL (1978) Multiobjective programming and planning, vol 140. Courier Corporation
Conforti M, Cornuéjols G, Zambelli G et al (2014) Integer programming, vol 271. Springer
Csirmaz L (2021) Inner approximation algorithm for solving linear multiobjective optimization problems. Optimization pp 1487–1511
De Santis M, Eichfelder G, Niebling J, Rocktäschel S (2020) Solving multiobjective mixed integer convex optimization problems. SIAM J Optim 30(4):3122–3145
Demir E, Bektaş T, Laporte G (2014) The bi-objective pollution-routing problem. Eur J Oper Res 232(3):464–478
Dial RB (1979) A model and algorithm for multicriteria route-mode choice. Transp Res 13B:311–316
Dörfler D, Löhne A (2018) Geometric duality and parametric duality for multiple objective linear programs are equivalent. J Nonlinear Convex Anal 19(7):1181–1188
Ehrgott M (2005) Multicriteria optimization, vol 491. Springer
Ehrgott M, Gandibleux X (2007) Bound sets for biobjective combinatorial optimization problems. Comput Oper Res 34(9):2674–2694
Ehrgott M, Shao L, Schöbel A (2011) An approximation algorithm for convex multi-objective programming problems. J Global Optim 50(3):397–416
Ehrgott M, Löhne A, Shao L (2012) A dual variant of Benson's "outer approximation algorithm" for multiple objective linear programming. J Glob Optim 52(4):757–778
Ehrgott M, Gandibleux X, Przybylski A (2016) Exact methods for multi-objective combinatorial optimisation. In: Greco S, Ehrgott M, Figueira JR (eds) Multiple criteria decision analysis: state of the art surveys. Springer, New York, pp 817–850
Eichfelder G, Stein O, Warnow L (2023) A solver for multiobjective mixed-integer convex and nonconvex optimization. J Optim Theory Appl 874:1–31
Eskandarpour M, Dejax P, Péton O (2021) Multi-directional local search for sustainable supply chain network design. Int J Prod Res 59(2):412–428
Forget N, Gadegaard SL, Klamroth K, Nielsen LR, Przybylski A (2022) Branch-and-bound and objective branching with three or more objectives. Comput Oper Res 148:106012
Forget N, Gadegaard SL, Nielsen LR (2022) Warm-starting lower bound set computations for branch-and-bound algorithms for multi objective integer linear programs. Eur J Oper Res 302(3):909–924
Forget N, Parragh SN (2023) Enhancing branch-and-bound for multi-objective 0-1 programming. INFORMS J Comput
Gadegaard SL, Nielsen LR, Ehrgott M (2019) Bi-objective branch-and-cut algorithms based on LP relaxation and bound sets. INFORMS J Comput 31(4):790–804
Geoffrion AM, Nauss R (1977) Exceptional paper-parametric and postoptimality analysis in integer linear programming. Manag Sci 23(5):453–466
Grötschel M, Lovasz L, Schrijver A (1993) Geometric algorithms and combinatorial optimization. Springer
Halffmann P, Dietz T, Przybylski A, Ruzika S (2020) An inner approximation method to compute the weight set decomposition of a triobjective mixed-integer problem. J Glob Optim 77(4):715–742
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. J Multi-Criteria Decis Anal 29:341–363
Heyde F, Löhne A (2008) Geometric duality in multiple objective linear programming. SIAM J Optim 19(2):836–845
Kaparis K, Letchford AN (2010) Separation algorithms for 0–1 knapsack polytopes. Math Program 124:69–91
Kirlik G, Sayın S (2014) A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. Eur J Oper Res 232(3):479–488
Löhne A (2011) Vector optimization with infimum and supremum. Vector optimization. Springer

Löhne A, Weißing B (2017) The vector linear program solver Bensolve-notes on theoretical background. Eur J Oper Res 260(3):807–813
Löhne A, Rudloff B, Ulus F (2014) Primal and dual approximation algorithms for convex vector optimization problems. J Global Optim 60(4):713–736
Lotov AV, Bushenkov VA, Kamenev GK (2004) Interactive decision maps: approximation and visualization of Pareto frontier, vol 89. Springer
Luc DT (2011) On duality in multiple objective linear programming. Eur J Oper Res 210(2):158–168
Maher SJ, Fischer T, Gally T, Gamrath G, Gleixner A, Gottwald RL, Hendel G, Koch T, Lübbecke ME, Miltenberger M et al (2017) The SCIP optimization suite 4.0
Özpeynirci Ö, Köksalan M (2010) An exact algorithm for finding extreme supported nondominated points of multiobjective mixed integer programs. Manag Sci 56(12):2302–2315
Parragh SN, Tricoire F (2019) Branch-and-bound for bi-objective integer programming. INFORMS J Comput 31(4):805–822
Parragh SN, Tricoire F, Gutjahr WJ (2021) A branch-and-benders-cut algorithm for a bi-objective stochastic facility location problem. OR Spectr 25:1–41
Perini T, Boland N, Pecin D, Savelsbergh M (2020) A criterion space method for biobjective mixed integer programming: the boxed line method. INFORMS J Comput 32(1):16–39
Przybylski A, Gandibleux X, Ehrgott M (2010) A recursive algorithm for finding all nondominated extreme points in the outcome set of a multiobjective integer programme. INFORMS J Comput 22(3):371–386
Przybylski A, Klamroth K, Lacour R (2019) A simple and efficient dichotomic search algorithm for multiobjective mixed integer linear programs. arXiv preprint arXiv:1911.08937
Ramos TRP, Gomes MI, Barbosa-Póvoa AP (2014) Planning a sustainable reverse logistics system: balancing costs with environmental and social concerns. Omega 48:60–74
Rasmi SAB, Türkay M (2019) GoNDEF: an exact method to generate all non-dominated points of multiobjective mixed-integer linear programs. Optim Eng 20(1):89–117
Ruhe G (1988) Complexity results for multicrierial and parametric network flows using a pathological graph of Zadeh. Z Oper Res 32(1):9–27
Soylu B (2018) The search-and-remove algorithm for biobjective mixed-integer linear programming problems. Eur J Oper Res 268(1):281–299
Stidsen T, Andersen KA, Dammann B (2014) A branch and bound algorithm for a class of biobjective mixed integer programs. Manag Sci 60(4):1009–1032
Ziegler GM (2012) Lectures on polytopes, vol 152. Springer