BIOBJECTIVE OPTIMIZATION OVER THE EFFICIENT SET OF MULTIOBJECTIVE INTEGER PROGRAMMING PROBLEM

YASMIN CHERFAOUI
USTHB, Department of Operational Research
Bp 32 El Alia, 16111 Algiers, Algeria

MUSTAPHA MOUNI
USTHB, LaROMaD Laboratory,
Bp 32 El Alia, 16111 Algiers, Algeria

(Communicated by Kaisa Miettinen)

Abstract. In this article, an exact method is proposed to optimize two preference functions over the efficient set of a multiobjective integer linear program (MOILP). This kind of problems arises whenever two associated decision-makers have to optimize their respective preference functions over many efficient solutions. For this purpose, we develop a branch-and-cut algorithm based on linear programming, for finding efficient solutions in terms of both preference functions and MOILP problem, without explicitly enumerating all efficient solutions of MOILP problem. The branch and bound process, strengthened by efficient cuts and tests, allows us to prune a large number of nodes in the tree to avoid many solutions. An illustrative example and an experimental study are reported.

1. Introduction. In many real life situations decision-makers encounter with multiple conflicting objectives to optimize such as minimizing time, maximizing profitability etc. This is why that kind of problems have attracted attention of many authors [2, 5, 7, 10]. In general, multiobjective optimization aims to find a set of solutions called efficient solutions. The size of this set is proportional to the problem size and the computational burden of generating all the set grows rapidly with problem size [7]. Thus why is it not interesting to generate each element of this set? Instead decision-maker can give a preference function, which provides an appropriate solution. These problems are called optimizing a function over the efficient set and are treated by many authors such as [4, 6, 13, 3]. Their solution can be quite beneficial since it can reduce the search time and yields exactly the preferred solution. However, suppose that there exist two decision-makers or one decision-maker expressing two preference functions. The solution of this problem will provide a subset of the efficient set. Hence in this paper, we will consider the problem of optimizing two preference functions over the efficient set of MOILP.

In practice, the efficient set is infinite in the case where the feasible domain is a polyhedron. So in this article we will consider the case where the domain is discrete,
and the two preference functions are linear, so the domain is not convex and the
efficient set is finite.

The optimization of one linear preference function over a MOILP has been con-
sidered and solved by many authors [1, 9, 11]. In [12], the authors strengthen the
branch-and-bound procedure by adding an efficiency test and an efficient cut to
the branching process. Inspired by this technique, even though our method treats
a more difficult problem, it can provide the preferred solutions regarding the two
main functions among the efficient set without browsing all the efficient set in a
reasonable execution time.

This paper is organized as follow: In the following section, the problem is defined
and some definitions and notations concerning the MOILP are given. In Section
3, we outline the solution method and present the algorithm. Then an illustrative
example is solved in Section 4. Computational experiments are reported in Section
5. The last section summarizes concluding remarks.

2. Definitions and preliminaries. The multiobjective integer linear program-
ming problem (MOILP) can be defined as follows:

\[ \text{(MOILP)} \begin{cases} \max Z_i(x) = c^i x, i \in \{1, 2, \ldots, r\} \\ s.t. \quad x \in D = X \cap \mathbb{Z}^n \end{cases} \tag{1} \]

where the following notations and assumptions are employed: \( Z \) is the set of integers,
\( r \geq 2, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, (c^i) = C \in \mathbb{R}^{r \times n} \) are row vectors,
\( X = \{ x \in \mathbb{R}^n | Ax \leq b, x \geq 0 \} \).

The set \( X \) is a nonempty convex polytope and the set \( D \) is not empty. Several
authors propose a method to solve a (MOILP) \([1, 7, 8]\) using the concept of efficiency.
A feasible solution \( x^* \in D \) is called efficient solution for (MOILP) problem if and
only if there is no feasible solution \( x \in D \) such that \( Z_i(x) \geq Z_i(x^*) \) for all \( i \in \{1, \ldots, r\} \) and \( Z_k(x) > Z_k(x^*) \) for at least one \( k \in \{1, \ldots, r\} \). Otherwise, \( x^* \) is not
efficient and its criterion vector \( Z(x^*) \) is dominated by the criterion vector \( Z(x) \),
where \( Z(x^*) = (Z_i(x^*))_{i=1}^{r} \).

Let \( X_E \) denote the set of all efficient solutions for a MOILP problem. Another
type of problem that is well studied is the optimization of a linear function over the
efficient set, which is defined as follows. Let \( d \) be a row vector in \( \mathbb{R}^n \):

\[ \text{(P1)} \begin{cases} \max dx \\ s.t. \quad x \in X_E \end{cases} \tag{2} \]

Since Benson’s initiating article [4], optimization over the efficient set has evolved
sequentially [13], several strategies having been proposed to solve this problem.
Some strategies propose to explore the criterion space where upper and lower bound-
aries are generally used to progressively narrow the search [3]. While other strategies
explore the solutions space and trace adjacent efficient vertex [9], those methods
present generally the inconvenient of enumerating all efficient vertices.

In [12], the authors proposed a branch-and-cut procedure that combines the well-
known Ecker and Kouada efficiency test [8], and the efficient cut cited in [5]. In
fact this method allows to trace efficient vertices in the solutions space and move
from one efficient solution to another without having to enumerate the efficient set
and explores the domain by optimizing the objective function.
So if $\mathcal{X}_l$ is the subset of the feasible solutions at step $l$ where the optimal integer solution is $x^*(l)$ and $\mathcal{B}_l, \mathcal{N}_l$ are the index sets of basic and nonbasic variables respectively of the optimal solution $x^*(l)$:

$$\sum_{j \in \mathcal{H}_l} x_j \geq 1$$  \hspace{1cm} (3)

where

$$\mathcal{H}_l = \left\{ j \in \mathcal{N}_l | \exists i \in \{1, \ldots, r\} \text{ with } c_{ij} > 0 \right\} \cup \left\{ j \in \mathcal{N}_l | c_{ij} = 0, \forall i = 1, \ldots, r \right\}$$

The inequality (3) is called efficient cut because it takes into account the efficiency and has the property of removing inefficient solutions without having to enumerate them [5].

In this paper we propose a new type of problem that can be of use in at least two types of situations. The first type may occur when a decision-maker has two preference functions for evaluating the efficient solutions of a multiobjective problem. For the second type of situation, there are two decision makers. For instance, let consider the existence of two firms that have joined in a common project. This project requires optimizing several conflicting objectives, but these two firms have also individual profit functions to maximize since they do not participate in the project in the same way, the problem is to find solutions that optimize two preference functions over an efficient set of multiobjective problem.

Let $d^1, d^2$ be two row-vectors in $\mathbb{R}^n$ representing preference functions of decision makers. This problem can be written as:

$$\begin{align*}
(P^2) & \max d^1 x \\
& \max d^2 x \\
& s.t. \quad x \in \mathcal{X}_E
\end{align*}$$

3. **Methodology, algorithm and theoretical results.** In what follows, we propose a branch-and-cut procedure to solve the program $(P^2)$ where $\mathcal{X}_E$ is the efficient set of MOILP (1). The solution of $(P^2)$ provides a subset $\mathcal{X}_C \subset \mathcal{X}_E$ such as the elements in $\mathcal{X}_C$ are also efficient in terms of $d_1$ and $d_2$. The naïve method of solving the problem would be to generate the whole set $\mathcal{X}_E$ then to select the elements of $\mathcal{X}_E$ that can be put in $\mathcal{X}_C$, of course it would be computationally heavy.

3.1. **Methodology description and algorithm.** The proposed algorithm generates all elements of $\mathcal{X}_C$ without having to enumerate $\mathcal{X}_E$. It is based on branch-and-cut process. The procedure is strengthened by two types of cuts allowing to fathom nodes more quickly.

At each node $l$ the following linear program is solved using the simplex or the dual simplex method:

$$\begin{align*}
(LP_l) & \max d^1 x \\
& s.t. \quad x \in \mathcal{X}_l
\end{align*}$$

where $\mathcal{X}_0 = \mathcal{X}$ and $\mathcal{X}_l$ is a subset of the original set $\mathcal{X}$ to explore at node $l$ of the tree.

If this program has no solution, the node is fathomed, otherwise, there are two cases; the first case is that the optimal solution $x^*(l)$ is not integer: Then a normal branching process is followed. The other case is that the solution $x^*(l)$ is integer:
Then two sets $H_l$ and $H'_l$ are constructed and the efficient cuts (5) and (6) are eventually added to the successor nodes of $l$.

$$\sum_{j \in H_l} x_j \geq 1 \quad (5)$$

$$H_l = \{ j \in N_l | \exists \in \{1, \ldots, r \}; c^*_j > 0 \} \cup \{ j \in N_l | c^*_j = 0, \forall i = 1, \ldots, r \}$$

$$\sum_{j \in H'_l} x_j \geq 1 \quad (6)$$

$$H'_l = \{ j \in N_l | d^*_j > 0 \} \cup \{ j \in N_l | d^*_j = 0 \}$$

Thus corresponding domain of node $l_1$, the successor of $l$, is obtained by applying (5) and (6) to $X_l$

$$X_{l_1} = X^1_l \cap X^2_l. \quad (7)$$

After that two efficiency tests are preformed by solving the following programs :

$$(T^1_{x^*(i)}) \begin{cases} 
\max \sum_{i=1}^r w_i \\
\text{s.t.} \\
c^*x - w_i = c^*x^{*(i)}, \ i = 1, \ldots, r \\
x \in D_l = X^1_l \cap \mathbb{Z}^n, \\
w_i \geq 0, \ i = 1, \ldots, r \\
\max v_1 + v_2 \\
\text{s.t.} \\
d^*x - v_i = d^*x^{*(i)}, \ i = 1, 2 \\
x \in D_l = X^1_l \cap \mathbb{Z}^n \\
v_i \geq 0, \ i = 1, 2
\end{cases}$$

$$(T^2_{x^*(i)}) \begin{cases} 
\max \sum_{i=1}^r w_i \\
\text{s.t.} \\
c^*x - w_i = c^*x^{*(i)}, \ i = 1, \ldots, r \\
x \in D_l = X^1_l \cap \mathbb{Z}^n, \\
w_i \geq 0, \ i = 1, \ldots, r \\
\max v_1 + v_2 \\
\text{s.t.} \\
d^*x - v_i = d^*x^{*(i)}, \ i = 1, 2 \\
x \in D_l = X^1_l \cap \mathbb{Z}^n \\
v_i \geq 0, \ i = 1, 2
\end{cases}$$

A well-known result, see [8], is that the solution $x^{*(i)}$ is efficient for $MOILP$ if and only if $(T^1_{x^*(i)})$ has a maximum value of zero. Hence if both of $(T^1_{x^*(i)})$ and $(T^2_{x^*(i)})$ have a maximum value of zero then $x^{*(i)}$ belongs to $X_C$.

The algorithm is summarized as follows:

3.2. Theoretical results. The following theoretical tools show that the algorithm yields the set of solutions for the program (4) in a finite number of iterations.

**Theorem 1.** Assume that $H_l \neq \emptyset$ and $H'_l \neq \emptyset$ at the current integer solution $x^{*(i)}$. If $x \neq x^{*(i)}$ is an efficient solution of program (P2) in domain $X_l$, then $x \in X_{l_1}$ ($l_1$ is the successor of $l$).

**Proof.** Let $x \neq x^{*(i)}$ be an integer solution in domain $X_l$ such that $x \notin X_{l_1}$, two cases can occur:

- $x \notin X^1_l$ implies $x \in \{ x \in X_l | \sum_{j \in N_l \setminus H_l} x_j \geq 1 \}$.

Therefore, the components of vector $x$ satisfy the inequalities

$$\sum_{j \in H_l} x_j < 1$$

$$\sum_{j \in N_l \setminus H_l} x_j \geq 1,$$
Algorithm 1: Biobjective optimization over the integer efficient set

Result: $\mathcal{X}_C$, the solution set of $(P_2)$

initialization $l = 0$, $\mathcal{X}_l = \{x \in \mathbb{R}^n | Ax \leq b, x \geq 0\}$ and $\mathcal{X}_C = \emptyset$;

while there is a non-fathomed node $l$ do
    solve $(LP_l)$ using simplex or dual simplex method;
    if $(LP_l)$ has an optimal solution $x^{*(l)}$ then
        if $x^{*(l)}$ is integer then
            Solve $(T_1^l x^{*(l)})$;
            if the optimal value of the objective function of $(T_1^l x^{*(l)})$ is 0 then
                Solve $(T_2^l x^{*(l)})$;
                if the optimal value of the objective function of $(T_2^l x^{*(l)})$ is 0 then
                    $\mathcal{X}_C = \mathcal{X}_C \cup \{x^{*(l)}\}$;
                end
            end
        end
        Construct the sets $H_l$ and $H'_l$;
        if $H_l = \emptyset$ or $H'_l = \emptyset$ then
            Fathom the node $l$
        else
            Add the cuts (5) and (6) to the successors of $l$;
        end
    else
        Choose an index $k$ such as $x_k^{*(l)}$ is fractional. Then, split the program $(LP_l)$ into two sub programs, by adding respectively the constraints $x_k \leq x_k^{*(l)}$ and $x_k \geq \lfloor x_k^{*(l)} \rfloor + 1$ to obtain $(LP_{l_1})$ and $(LP_{l_2})$ ($l_1 \geq l + 1$, $l_2 > l + 1$ and $l_1 \neq l_2$);
    end
    else
        Fathom the node $l$;
    end
end

this means that $x_j = 0$ for all $j \in H_l$, and $x_j \geq 1$ for at least one index $j \in \mathcal{N}_l \setminus H_l$. Using the simplex table in $x^{*(l)}$, the following equality is supported for all criterion $i \in \{1, \ldots, r\}$:

$$Z_i(x) = c_i^t x = \sum_{j \in B_l} c_{i,j}^l x_j + \sum_{j \in N_l} c_{i,j}^l x_j, \quad \text{where} \quad \sum_{j \in B_l} c_{i,j}^l x_j = Z_i(x^{*(l)}).$$

Then, we can write

$$Z_i(x) - Z_i(x^{*(l)}) = \sum_{j \in N_l} c_{i,j}^l x_j$$

$$= \sum_{j \in H_l} c_{i,j}^l x_j + \sum_{j \in N_l \setminus H_l} c_{i,j}^l x_j$$

$$= \sum_{j \in N_l \setminus H_l} c_{i,j}^l x_j.$$ 

Thus, $Z_i(x) \leq Z_i(x^{*(l)})$ for all criterion $i \in \{1, \ldots, r\}$, with $Z_i(x) < Z_i(x^{*(l)})$ for at least one criterion since $c_{i,j}^l \leq 0$ for all $j \in N_l \setminus H_l$. 

Proof. Without loss of any elements in $X$, Theorem 3. The algorithm terminates in a finite number of iterations and the set $X_C$ contains all the solutions of (P2).

Proposition 2. Suppose that $H_t = \emptyset$ or $H'_t = \emptyset$ at the current integer solution $x^{*}(l)$ then there is no solution in the remaining domain that is not dominated by $x^{*}(l)$.

Proof. Assume $H_t = \emptyset$, then $\forall i \in \{1, \ldots, r\}$, $\forall j \in N_i$, we have $\overline{c}_j \leq 0$ and $\exists i_0 \in \{1, \ldots, r\}$ such that $\overline{c}_{i_0} < 0, \forall j \in N_i$. So, $x^{*}(l)$ dominates all points $x, x \neq x^{*}(l)$ of domain $D_l$.

Now assume that $H'_t = \emptyset$, then $\forall j \in N_i$, $\overline{d}_j^l < 0$ or $\overline{d}_j^l = 0$ and $\overline{d}_j^l < 0$, adding to that $\overline{d}_j^l < 0, \forall j \in N_i$ since it is an optimal solution for $(LP_t)$, $x^{*}(l)$ becomes the most preferred solution in the domain $D_l$.

Theorem 3. The algorithm terminates in a finite number of iterations and the set $X_C$ contains all the solutions of (P2).

Proof. Let $D$, the set of integer feasible solutions of MOILP problem, be a finite bounded set contained in $X$. The cardinality of efficient sets $X_C$ and $X_E$ is a finite number. It contains a finite number of integer solutions. So the search tree would have a finite number of branches. Thus the algorithm terminates in a finite number of steps.

In order $X_C$ to contain all the solutions of (P2), the fathoming rules are used without loss of any elements in $X_C$. At each step $l$ of the algorithm 1, if an integer solution $x^{*}(l)$ is found, the cuts eliminate $x^{*}(l)$ and all dominated solutions from search (see proposition 2). So the first fathoming rule is when the set $H_t$ or $H'_t$ is empty. In this case the current node can be pruned since the rest of the domain contains only dominated solutions either in terms of the multiobjective program or in terms of the two preference functions. The second rule is the trivial case when the reduced domain becomes infeasible, whether it is because of previous cuts or the branching.
4. **Illustrative example.** Let consider two decision makers with the following preference functions. We solve the following problem using the proposed branch-and-cut algorithm as follows:

\[
\begin{align*}
\max & \quad \frac{1}{2}x_1 + 5x_2 - \frac{1}{2}x_3 \\
\max & \quad x_3 \\
\text{s.t.} & \quad (x_1, x_2, x_3) \in \mathcal{X}_E
\end{align*}
\]

where \( \mathcal{X}_E \) is the efficient set of the following (MOILP) problem:

\[
\begin{align*}
\max & \quad x_1 - 2x_2 - x_3 \\
\max & \quad x_2 - \frac{1}{2}x_3 \\
\max & \quad x_1 - 8x_3 \\
\text{s.t.} & \quad \frac{1}{2}x_1 - x_2 - x_3 \leq \frac{7}{2} \\
& \quad \frac{1}{2}x_1 + \frac{3}{2}x_2 \leq 8 \\
& \quad x_2 + \frac{5}{2}x_3 \leq 6 \\
& \quad x_1, x_2, x_3 \geq 0 \text{ and integers.}
\end{align*}
\]

**Initialization:** We put \( l = 0, \mathcal{X}_C = \emptyset \) and \( \mathcal{X}_0 = \mathcal{X} \).

**Node 0:** solving \((LP_0)\) gives Table 1:

| \( B_1 \) | \( x_1 \) | \( x_3 \) | \( x_5 \) | RHS |
|---|---|---|---|---|
| \( x_4 \) | 5/6 | -1 | 2/3 | 53/6 |
| \( x_2 \) | 1/3 | 0 | 2/3 | 16/3 |
| \( x_6 \) | 1/3 | 5/2 | -2/3 | 2/3 |
| \( d^3 \) | -7/6 | -1/2 | -10/3 | 80/3 |

Since the optimal solution \((0, \frac{16}{3}, 0)\) is not integer, the problem is divided into two subproblems by adding constraints \( x_2 \leq \left\lfloor \frac{16}{3} \right\rfloor \) and \( x_2 \geq \left\lfloor \frac{16}{3} \right\rfloor + 1 \) to the problem \((LP_0)\) as follows

\[
\begin{align*}
\text{(LP}_1\text{)} & \quad \begin{cases}
\max & \frac{1}{2}x_1 + 5x_2 - \frac{1}{2}x_3 \\
\text{s.t.} & \quad x \in \mathcal{X}_0 \\
x_2 \leq 5
\end{cases} \\
\text{(LP}_2\text{)} & \quad \begin{cases}
\max & \frac{1}{2}x_1 + 5x_2 - \frac{1}{2}x_3 \\
\text{s.t.} & \quad x \in \mathcal{X}_0 \\
x_2 \geq 6
\end{cases}
\end{align*}
\]

**Node 1:** Solving the problem \((LP_1)\) gives the following Table 2:
Table 2. Optimal simplex table for node 1

| \( B \) | \( x_3 \) | \( x_5 \) | \( x_7 \) | RHS |
|---|---|---|---|---|
| \( x_4 \) | -1 | -1 | \( \frac{5}{2} \) | 8 |
| \( x_1 \) | 0 | 2 | -3 | 1 |
| \( x_6 \) | \( \frac{5}{2} \) | 0 | -1 | 1 |
| \( x_2 \) | 0 | 0 | 1 | 5 |
| \( \bar{d}_1 \) | \( \frac{1}{2} \) | -1 | \( -\frac{7}{2} \) | \( \frac{51}{2} \) |
| \( \bar{d}_2 \) | 1 | 0 | 0 | 0 |
| \( \bar{c}_1 \) | -1 | -2 | 5 | -9 |
| \( \bar{c}_2 \) | \( \frac{1}{2} \) | 0 | -2 | 10 |
| \( \bar{c}_3 \) | -1 | -2 | 3 | 1 |

The optimal solution found \((1, 5, 0)\) is integer and so: \( X_C = \{(1, 5, 0)\} \).

From Table 2, \( H_1 = \{3\} \) and \( H'_1 = \{7\} \) and the node 3 is generated.

**Node 2:** The program \((LP_2)\) is infeasible.

**Node 3:** After adding the cuts \( x_3 \geq 1 \) and \( x_7 \geq 1 \) to Table 2 and solving the problem, it gives Table 3:

Table 3. Optimal simplex table for node 3

| \( B \) | \( x_5 \) | \( x_6 \) | \( x_9 \) | RHS |
|---|---|---|---|---|
| \( x_4 \) | 1 | \( \frac{5}{2} \) | \( \frac{21}{4} \) | \( \frac{21}{4} \) |
| \( x_1 \) | 2 | -3 | \( \frac{15}{2} \) | \( \frac{11}{2} \) |
| \( x_3 \) | 0 | 0 | -1 | 1 |
| \( x_2 \) | 0 | 1 | \( \frac{5}{2} \) | \( \frac{7}{2} \) |
| \( x_7 \) | 0 | -1 | \( -\frac{5}{2} \) | \( \frac{3}{2} \) |
| \( x_8 \) | 0 | -1 | \( -\frac{5}{2} \) | \( \frac{1}{2} \) |
| \( \bar{d}_1 \) | -1 | \( -\frac{1}{2} \) | \( -\frac{21}{4} \) | \( \frac{79}{4} \) |

The optimal solution \((11, 7, 1)\) is not integer so the constraints \( x_1 \leq \lfloor \frac{11}{2} \rfloor \) and \( x_1 \geq \lfloor \frac{11}{2} \rfloor + 1 \) are added respectively to Table 3 and the nodes 4 and 5 are generated.

**Node 4:** Solving the problem gives Table 4:
Table 4. Optimal simplex table for node 4

| B₄   | x₆  | x₉  | x₁₀ | RHS |
|------|-----|-----|-----|-----|
| x₄   | 1   | 3/2 | 1/2 | 11/2|
| x₂   | 1   | 5/2 | 0   | 7/2 |
| x₃   | 0   | -1  | 0   | 1   |
| x₁   | 0   | 0   | 1   | 5   |
| x₅   | 3/2 | -15/4 | 1/2 | 1/2 |
| x₇   | -1  | 5/2 | 0   | 3/2 |
| x₈   | -1  | 5/2 | 0   | 1/2 |
| d₁   | -5  | -13 | -1/2| 39/2|

The optimal solution found (5, 7/2, 1) is not integer so the constraints \(x₂ \leq \left\lfloor \frac{7}{2} \right\rfloor\) and \(x₂ \geq \left\lfloor \frac{7}{2} \right\rfloor + 1\) are added respectively to Table 4 and node 6 and 7 are generated.

**Node 5:** By adding the constraint \(x₂ \leq \left\lfloor \frac{5}{3} \right\rfloor\) to Table 3 and solving, we obtain Table 5.

Table 5. Optimal simplex table for node 5

| B₅   | x₅  | x₉  | x₁₀ | RHS |
|------|-----|-----|-----|-----|
| x₄   | 2/3 | -1  | 5/6 | 29/6|
| x₁   | 0   | 0   | -1  | 6   |
| x₃   | 0   | -1  | 0   | 1   |
| x₂   | 2/3 | 0   | 1/3 | 10/3|
| x₇   | -2/3| 0   | -1/3| 2/3 |
| x₈   | -2/3| 0   | -1/3| 2/3 |
| x₆   | -2/3| 5/2 | -1  | 1/2 |
| d₁   | -10/3| -1/2| 7/6 | 115/6|

The optimal solution (6, 10/3, 1) is not integer so the constraints \(x₂ \leq \left\lfloor \frac{10}{3} \right\rfloor\) and \(x₂ \geq \left\lfloor \frac{10}{3} \right\rfloor + 1\) are added respectively to Table 5 to generate node 8 and 9.

**Node 6:** Adding the constraint \(x₂ \leq \left\lfloor \frac{7}{2} \right\rfloor\) to Table 4 produces Table 6.
Table 6. Optimal simplex table for node 6

| $B_6$ | $x_9$ | $x_{10}$ | $x_{11}$ | RHS |
|-------|-------|----------|----------|-----|
| $x_4$ | -1    | $-\frac{1}{2}$ | 1        | 5   |
| $x_1$ | 0     | 1         | 0        | 5   |
| $x_3$ | -1    | 0         | 0        | 1   |
| $x_2$ | 0     | 0         | 0        | 3   |
| $x_5$ | 0     | $-\frac{1}{2}$ | $-\frac{3}{2}$ | 1 |
| $x_8$ | 0     | 0         | -1       | 1   |
| $x_6$ | 5     | 0         | -1       | $\frac{1}{7}$ |
| $x_7$ | 0     | 0         | -1       | 2   |

$\bar{d}^1$ | $-\frac{1}{2}$ | $-\frac{1}{2}$ | -5 | 17 |
$\bar{d}^2$ | 1 | 0 | 0 | 1 |
$c^1$ | -1 | -1 | 2 | -2 |
$c^2$ | $-\frac{1}{2}$ | 0 | -2 | $\frac{11}{2}$ |
$c^3$ | -1 | -1 | 0 | 4 |

The optimal solution found (5, 3, 1) is integer but not efficient since the solution of $T^1_{(5,3,1)}$ gives a nonzero at optimality, $H_6 = \{11\}$ and $H'_6 = \{9\}$ and the node 10 is generated.

**Node 7**: Solving the problem generated after adding the constraint $x_2 \geq \lceil \frac{7}{2} \rceil + 1$ to Table 4 gives no solution and the node is fathomed.

**Node 8**: Solving the problem generated after adding the constraint $x_2 \leq \lceil \frac{10}{3} \rceil$ to Table 5 gives the optimal simplex Table 7.

Table 7. Optimal simplex table for node 8

| $B_8$ | $x_5$ | $x_9$ | $x_{11}$ | RHS |
|-------|-------|-------|----------|-----|
| $x_4$ | -1    | -1    | $\frac{5}{2}$ | 4   |
| $x_2$ | 0     | 0     | 1         | 3   |
| $x_3$ | 0     | -1    | 0         | 1   |
| $x_1$ | 2     | 0     | -3        | 7   |
| $x_6$ | 0     | $\frac{5}{2}$ | -1       | $\frac{1}{2}$ |
| $x_7$ | 0     | 0     | -1        | 2   |
| $x_8$ | 0     | 0     | -1        | 1   |
| $x_{10}$ | 2 | 0 | -3 | 1 |

$\bar{d}^1$ | -1 | $-\frac{1}{2}$ | $-\frac{7}{2}$ | 18 |
$\bar{d}^2$ | 0 | 1 | 0 | 1 |
$c^1$ | -2 | -1 | 5 | 0 |
$c^2$ | 0 | $-\frac{1}{2}$ | -2 | $\frac{11}{2}$ |
$c^3$ | -2 | -1 | 3 | 6 |
the optimal solution found (7, 3, 1) is not efficient, the node 11 is generated with $H_8 = \{11\}$ and $H'_8 = \{9\}$.

**Node 9:** The problem became infeasible after adding the constraint $x_2 \geq \lceil \frac{10}{x} \rceil + 1$ to Table 5, the node is fathomed.

**Node 10:** solving the problem gives Table 8.

### Table 8. Optimal simplex table for the node 10

| $x_6$  | $x_9$  | $x_{10}$ | $x_{13}$ | RHS |
|--------|--------|----------|----------|-----|
| $x_5$  | $-\frac{3}{2}$ | $-\frac{1}{2}$ | $-\frac{15}{4}$ | 4   |
| $x_4$  | 1      | $-\frac{1}{2}$ | $\frac{3}{2}$  | 4   |
| $x_3$  | 0      | 0         | $-1$       | 2   |
| $x_7$  | -1     | 0         | $-\frac{5}{2}$| 4   |
| $x_8$  | -1     | 1         | $-\frac{5}{2}$| 3   |
| $x_9$  | 0      | 0         | $-1$       | 1   |
| $x_2$  | 1      | 0         | $\frac{5}{2}$ | 1   |
| $x_1$  | 0      | 1         | 0          | 5   |

The optimal solution found (5, 1, 2) is a rejected integer solution since $(T_{10}^{1}(5, 1, 2))$ admits a nonzero maximum value. So $H_{10} = \{6, 13\}$ and $H'_{10} = \{13\}$ and the node 12 is generated.

**Node 11:** After adding the cuts $x_9 \geq 1$ and $x_{11} \geq 1$ and solving the problem, Table 9 is obtained.

The optimal solution found (13, 1, 2) is integer so $X_C = \{(1, 5, 0), (13, 1, 2)\}$, $H_{11} = \{13\}$ and $H'_{11} = \{13\}$, the node 13 is generated.

**Node 12:** After adding the cuts $x_6 + x_{13} \geq 1$ and $x_{13} \geq 1$ the problem became infeasible and the node is fathomed.

**Node 13:** After adding the cut $x_{13} \geq 1$ the problem became infeasible, the node is fathomed.

The final set found is $X_C = \{(13, 1, 2), (1, 5, 0)\}$ while $X_E = \{(13, 1, 2), (11, 1, 1), (10, 2, 0), (9, 1, 0), (7, 3, 0), (4, 4, 0), (1, 5, 0)\}$. The search tree is presented in Figure 1.

5. **Computational results.** The proposed branch-and-cut algorithm is implemented in Visual Studio 2010 environment. The linear and integer linear programs are solved using the library IBM CPLEX 12.6 for C++ programs. To perform tests we have used a computer with an Intel Pentium 2.53 GHz processor and 8GB of memory. The method is tested on randomly generated MOILP with two randomly generated preferences functions. The objective functions and constraints coefficients are uncorrelated uniformly distributed. Each component of the vector $b$ and the
The problems were grouped according to the number of variables, constraints and objective functions into six categories. In each category the number of objective functions $r$ changes from 3 to 8. For each category of problems, 50 instances were solved.

The computational results obtained are summarized in Table 10. The statistics about the CPU time (in seconds) are reported. The two last columns $\mu$ and $\rho$ refer to the average of $|X_E|$ and the average of $|X_C|/|X_E|$, respectively. The elements and the cardinality of $X_E$ are calculated using the algorithm presented in [5].

From the computational experiments shown in Table 10, it appears that $\rho$ does not exceed a maximum of 0.17 and has a decreasing trend compared to the size of the problem. These random experiments illustrate the usefulness of calculating $X_C$ since the cardinality of $X_C$ is much smaller than that of $X_E$ which explains the reasonability of the execution times.

6. Conclusion. In this paper we present an exact method to solve a new type of problem, where two decision makers are available in a multiobjective problem. The solution of the problem has two main advantages: The first is that, unlike the method for optimizing a single function over the efficient set, the optimization of two functions offers to decision-makers a set of solutions, in a finite number of iterations. The second advantage is that the solution is obtained without browsing.
| n x n | CPU time (second) | Mean | Min. | Max. |
|-------|------------------|------|------|------|
| 10 x 10 | 0.03 | 0.03 | 0.03 | 0.03 |
| 20 x 20 | 0.13 | 0.13 | 0.13 | 0.13 |
| 30 x 30 | 0.23 | 0.23 | 0.23 | 0.23 |
| 40 x 40 | 0.33 | 0.33 | 0.33 | 0.33 |

**Table 10. Random instances execution times**
all the efficient set so it requires less computationally effort than solving a MOILP problem.

The proposed solution method is a branch-and-cut algorithm using efficient cuts that reduce the search domain of the MOILP problem iteratively. Moreover, this algorithm can directly be extended to multiple decision makers by including the associated preferences in the efficient cut.

For future works, we are intended to include the nonlinear form in the multi-objective model and to study this problem with nonlinear preference functions of decision makers.

REFERENCES

[1] M. Abbas and D. Chaabane, Optimizing a linear function over an integer efficient set, European J. Oper. Res., 174 (2006), 1140–1161.
[2] P. Belotti, B. Soylu and M. Wiecek, Fathoming rules for biobjective mixed integer linear programs: Review and extensions, Discrete Optim., 22 (2016), 341–363.
[3] N. Boland, H. Charkhgard, and M. Savelsbergh, A new method for optimizing a linear function over the efficient set of a multiobjective integer program, European J. Oper. Res., 260 (2017), 904–919.

[4] H. P. Benson, Optimization over the efficient set, J. Math. Anal. Appl., 98 (1984), 562–580.

[5] M. E-A Chergui and M. Moulaï, An exact method for a discrete multiobjective linear fractional optimization, J. Appl. Math. Decis. Sci., (2008), 12pp.

[6] W. Drici, F-Z Ouail and M. Moulaï, Optimizing a linear fractional function over the integer efficient set, Ann. Oper. Res., 267 (2018), 135–151.

[7] J. G. Ecker, N. S. Hegner and I. A. Kouada, Generating all maximal efficient faces for multiple objective linear programs, J. Optim. Theory Appl., 30 (1980), 353–381.

[8] J. G. Ecker and I. A. Kouada, Finding all efficient extreme points for multiple objective linear programs, Math. Programming, 14 (1978), 249–261.

[9] J. G. Ecker and J. H. Song, Optimizing a linear function over an efficient set, J. Optim. Theory Appl., 83 (1994), 541–563.

[10] J. P. Evans and R. E. Steuer, A revised simplex method for linear multiple objective programs, Math. Programming, 5 (1973), 54–72.

[11] J. M. Jorge, An algorithm for optimizing a linear function over an integer efficient set, European J. Oper. Res., 195 (2009), 98–103.

[12] F-Z Ouail, M. E-A. Chergui and M. Moulaï, An exact method for optimizing a linear function over an integer efficient set, WSEAS Transactions on Circuits and Systems, 16(3) (2017), 141–148.

[13] Y. Yamamoto, Optimization over the efficient set: Overview, J. Global Optim., 22 (2002), 285–317.

Received May 2018; 1st revision December 2018; 2nd revision April 2019.

E-mail address: cheryasmine@gmail.com
E-mail address: mmoulai@usthb.dz