Computers & Operations Research 40 (2013) 2260–2269

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Choquet optimal set in biobjective combinatorial optimization

Thibaut Lust [a,*], Antoine Rolland [b]

[a] LIP6, Université Pierre et Marie Curie, 4, place Jussieu, F-75252 Paris Cedex 05, France
[b] Laboratoire ERIC - Université Lumière Lyon 2, 5, avenue Pierre Mendes-France, F-69676 Bron Cedex, France

ARTICLE INFO

Available online 12 April 2013

Keywords:
Multiobjective optimization
Choquet integral
Algorithmic decision theory
Knapsack problem
Minimum spanning tree problem

ABSTRACT

We study in this paper the generation of the Choquet optimal solutions of biobjective combinatorial optimization problems. Choquet optimal solutions are solutions that optimize a Choquet integral. The Choquet integral is used as an aggregation function, presenting different parameters, and allowing to take into account the interactions between the objectives. We develop a new property that characterizes the Choquet optimal solutions. From this property, a general method to easily generate these solutions in the case of two objectives is defined. We apply the method to two classical biobjective optimization combinatorial optimization problems: the biobjective knapsack problem and the biobjective minimum spanning tree problem. We show that Choquet optimal solutions that are not weighted sum optimal solutions represent only a small proportion of the Choquet optimal solutions and are located in a specific area of the objective space, but are much harder to compute than weighted sum optimal solutions.

1. Introduction

Multiobjective combinatorial optimization (MOCO) problems deal with situations where a decision-maker has to optimize several objectives simultaneously. These situations often come from a problem with a combinatorial number of solutions, for example spanning tree, shortest path, knapsack, traveling salesman tour, etc. [6]. Many algorithms have been proposed to solve these problems in the case of a single objective to be optimized [33]. But these algorithms have to be adapted when several objectives have to be taken into consideration.

To solve a MOCO problem, three different approaches are usually followed. In the *a posteriori* approach, all the Pareto optimal solutions, that is the solutions for which it does not exist a solution better or equal on all the objectives and better on at least one objective, are first generated. Once this has been done, the decision-maker is free to choose among all solutions the one that corresponds the best to his/her preferences. For this specific task, since the number of Pareto optimal solutions can be very large, different methods coming from the multicriteria decision making field can help [4].

Another possibility, called the *a priori* approach, is to first ask the decision-maker what are his/her preferences among all the objectives and to compute an aggregation function [13], like a weighted sum or an ordered weighted operator [34], with specified parameters. The aggregation function is then optimized and at the end, only one solution is generally proposed to the decision-maker (even if more than one optimal solution can exist).

A last possibility is to *interact* with the decision-maker along the process of generation of the solutions [22]. In this *interactive* approach, we ask the decision-maker to establish his/her preferences among different solutions, in order to guide the search, and to finally obtain a solution that suits him/her.

The approach considered in this paper is quite new and is between the *a posteriori* approach and the *a priori* approach. This new approach consists in trying to find the set of solutions that are optimal for at least one set of parameters of a given aggregation function. This approach presents two advantages comparing to the preceding ones: the parameters of the aggregation function do not have to be determined and at the end, a set of solutions of smaller size comparing to the set of Pareto optimal solutions is proposed to the decision-maker.

We will thus study in this paper how to optimize a specific aggregation function in the context of multiobjective combinatorial optimization problems. As usually done previously in the literature [25,27,31,32], only biobjective combinatorial optimization problems will be considered in this paper (for example, once the number of objectives is higher than two, generating simple weighted sum optimal solutions is nontrivial [26]).

We will also measure the impact of using an aggregation function instead of using Pareto dominance, that is which Pareto optimal solutions will be not retained if an aggregation function is used to compare solutions.

We focus in this paper on the Choquet integral as a specific aggregation function [5,13]. A Choquet integral can be seen as an integral on a non-additive measure (or capacity). This integral is widely used in multicriteria aggregation problems [12] as it can

* Corresponding author. Tel.: +33 144272440.
E-mail addresses: thibaut.lust@lip6.fr (T. Lust), antoine.rolland@univ-lyon2.fr (A. Rolland).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.04.003

model many specific aggregation operators, including, but not limited to, the average, the minimum, the maximum and all the statistic quantiles, the ordered weighted averaging, the weighted ordered weighted averaging, etc.

Some papers already deal with the optimization of the Choquet integral of multiobjective combinatorial optimization problems [8–10] when a specific capacity is given by the decision-maker. But to our knowledge, the development of a method to generate the whole set of Choquet optimal solutions has not yet been studied. In this paper, we present a property that characterizes the Choquet optimal set of biobjective combinatorial optimization problems and develop a method based on this property to generate the Choquet optimal set, containing the solutions that are optimal solutions of Choquet integrals.

In the next sections, we first define popular aggregation operators (Section 2) and then the Choquet integral (Section 3). The Section 4 is devoted to the generation of the Choquet optimal set. We first expose a brief state-of-the-art (Section 4.1), then a property that characterizes the Choquet optimal set (Section 4.2) and finally a method to generate this set (Section 4.3). In Section 5, we experiment the method with two classic biobjective combinatorial optimization (BOCO) problems: the biobjective knapsack problem and the biobjective minimum spanning tree problem.

2. Aggregation operators

Before defining the Choquet integral, we first introduce the formalism of a MOCO problem, and then present other popular aggregation operators, that can be seen as particular cases of the Choquet integral.

2.1. Definitions and notations

A multiobjective (linear) combinatorial optimization (MOCO) problem is generally defined as follows:

$$\text{“}\max_x\text{”} \quad f(x) = Cx = (f_1(x), f_2(x), \ldots, f_p(x))$$

$$\text{subject to} \quad Ax \le b \quad x \in \{0,1\}^n$$

$x \in \{0,1\}^n \longrightarrow n$ variables, $\quad i = 1, \ldots, n$
$C \in \mathbb{R}^{p \times n} \longrightarrow p$ objective functions, $\quad k = 1, \ldots, p$
$A \in \mathbb{R}^{r \times n}$ and
$b \in \mathbb{R}^{r \times 1} \longrightarrow r$ constraints, $\quad j = 1, \ldots, r$

A feasible solution x is a binary vector of n variables, having to respect the r constraints of the problem. Therefore, the feasible set in decision space is given by $\mathcal{X} = \{x \in \{0,1\}^n : Ax \le b\}$. The image of the feasible set is given by $\mathcal{Y} = f(\mathcal{X}) = \{f(x) : x \in \mathcal{X}\} \subset \mathbb{N}^p$. An element of the set $\mathcal{Y}$ is called a cost-vector or a point.

Let us recall the concept of Pareto efficiency. We consider that all the objectives have to be maximized.

Definition 1. The Pareto dominance relation (P-dominance for short) is defined, for all $y^1, y^2 \in \mathbb{Z}^p$, by

$$y^1 \succ_P y^2 \Longleftrightarrow [\forall k \in \{1, \ldots, p\}, y^1_k \ge y^2_k \text{ and } y^1 \ne y^2]$$

Definition 2. The strict Pareto dominance relation (sP-dominance for short) is defined as follows:

$$y^1 \succ_{sP} y^2 \Longleftrightarrow [\forall k \in \{1, \ldots, p\}, y^1_k > y^2_k]$$

Within a feasible set $\mathcal{X}$, any element x^1 is said to be *P-dominated* when $f(x^2) \succ_P f(x^1)$ for some x^2 in $\mathcal{X}$, *P-optimal* (or *P-efficient*) if there is no $x^2 \in \mathcal{X}$ such that $f(x^2) \succ_P f(x^1)$ and *weakly P-optimal* if there is no $x^2 \in \mathcal{X}$ such that $f(x^2) \succ_{sP} f(x^1)$. The *P-optimal set* denoted by $\mathcal{X}_P$ contains all the P-optimal solutions. The image $f(x)$ in the objective space of a P-optimal solution x is called a P-non-dominated point. The image of the P-optimal set in $\mathcal{Y}$, equal to $f(\mathcal{X}_P)$, is called the Pareto front, and is denoted by $\mathcal{Y}_P$. We can also define the *weakly P-optimal set* denoted by $\mathcal{X}_{wP}$ that contains all the weakly P-optimal solutions.

Generating all the P-optimal solutions of MOCO problems is an arduous and challenging task: MOCO problems are often $\mathcal{NP}-$Hard, with a non-convex feasible set $\mathcal{X}$ and a huge number of optimal solutions. Moreover, many MOCO problems have been proved to be intractable, that is there are instances for which the number of P-non-dominated points is exponential in the size of the instance [6]. Consequently, exact methods can only be applied to solve small size biobjective or three-objective problems [15,25,32].

2.2. Weighted sum

The most popular aggregation operator is the weighted sum (WS), where positive importance weights λ_i $(i = 1, \ldots, p)$ are allocated to the objectives:

Definition 3. Given a vector $y \in \mathbb{Z}^p$ and a weight set $\lambda \in \mathbb{R}^p_+$, the WS $f^{ws}_\lambda(y)$ of y is equal to

$$f^{ws}_\lambda(y) = \sum_{i=1}^{p} \lambda_i y_i$$

The main drawback of the WS is that this operator does not favor “balanced” solutions.[1] For example, if the WS is used to establish the average grade of students and if we consider three students that have respectively obtained the following notes for two subjects: (9,16), (12,12) and (16,9), the second student will never be ranked first, regardless of the weight set used. It is impossible to have $f^{ws}_\lambda(12,12) \ge f^{ws}_\lambda(9,16)$ and $f^{ws}_\lambda(12,12) \ge f^{ws}_\lambda(16,9)$ since for the first inequality, we need to have $\lambda_1 \ge \frac{4}{7}$ and for the second, $\lambda_1 \le \frac{3}{7}$.

This property is well-established in multiobjective optimization.

Definition 4. Let $x \in \mathcal{X}$ and $y = f(x)$ its image in $\mathcal{Y}$. If $\exists \lambda \in \mathbb{R}^p$ $(\lambda_i > 0)$ such that $f^{ws}_\lambda(y) \ge f^{ws}_\lambda(y^2)$ $\forall y_2 \in \mathcal{Y}$ then x is a supported P-optimal solution, and its image y a supported P-non-dominated point.

On the other hand, the P-optimal solutions that are not supported (that is they do not optimize a WS, regardless of the weight set) are called *non-supported* P-optimal solutions. The images of the supported P-efficient solutions are located on the boundary of the convex hull of $\mathcal{Y}$.

Note that optimizing a WS with $\lambda_i \ge 0$ only leads to weakly P-optimal solutions [6].

2.3. Ordered weighted average

The ordered weighted average (OWA) operator has been introduced by Yager [34] and is defined as follows:

Definition 5. Given a vector $y \in \mathbb{Z}^p$ and a weight set $w \in \mathbb{R}^p_+$ (with $\sum_{i=1}^p w_i = 1$, $w_i \in [0,1]$), the ordered weighted average (OWA) of y is called $f^{owa}_w(y)$ and is equal to $\sum_{i=1}^p w_i y_{\lfloor i \rfloor}$, where $\lfloor \cdot \rfloor$ represents the permutation over $\{1, \ldots, p\}$ such that $y_{\lfloor 1 \rfloor} \ge \cdots \ge y_{\lfloor p \rfloor}$ are the components of y sorted in non-increasing order.

[1] This notion is not easy to define, but we will simply say that if $y_i < y_j$ for some cost-vector, slightly improving (here increasing) component y_i to the detriment of y_j while preserving the mean of the costs would produce a better distribution of costs, and consequently a more equitable solution [21].

In OWA, the weights are assigned to the ordered values rather than to the specific objectives. The OWA operator allows to model the maximum ($w_1=1, w_i=0, i=2,\dots,p$), the minimum ($w_p=1$, $w_i=0, i=1,\dots,(p-1)$), the arithmetic mean ($w_i=1/p, i=1,\dots,p$) and all the statistic quantiles. Also, if the weight set w is well-chosen, OWA favors balanced solutions. For example, if $w_1 \le \frac{3}{7}$, $f_w^{owa}(12,12) \ge f_w^{owa}(9,16)$ and $f_w^{owa}(12,12) \ge f_w^{owa}(16,9)$. With the OWA operator, non-supported P-non-dominated points can be thus generated.

2.4. Weighted ordered weighted average

The OWA operator is a powerful operator often used to favor balanced solutions. However a simple weighted mean cannot be expressed with this operator. Therefore, Torra [30] has introduced a more general aggregation operator, the weighted ordered weighted average (WOWA), defined as follows:

Definition 6. Given a vector $y \in \mathbb{Z}^p$, a weight set $\lambda \in \mathbb{R}_+^p$ (with $\sum_{i=1}^p \lambda_i = 1$, $\lambda_i \in [0,1]$) which allows to define the importance accorded to the objectives and a weight set $w \in \mathbb{R}_+^p$ (with $\sum_{i=1}^p w_i = 1$, $w_i \in [0,1]$) which allows to express the importance attached to the order of the components, the weighted ordered weighted average (WOWA) $f_{w,\lambda}^{wowa}(y)$ of y is defined by

$$f_{w,\lambda}^{wowa}(y) = \sum_{i=1}^{p} \omega_i y_{\lfloor i \rfloor} \quad \text{with } \omega_i = \omega^*\left(\sum_{k \le i} \lambda_{\lfloor k \rfloor}\right) - \omega^*\left(\sum_{k < i} \lambda_{(\lfloor k \rfloor)}\right)$$

where ω^* is an increasing function interpolating the points $(i/p, \sum_{k \le i} w_k)$ together with the point (0,0) and $\lfloor \cdot \rfloor$ represents the permutation over $\{1,\dots,p\}$ such that $y_{\lfloor 1 \rfloor} \ge \cdots \ge y_{\lfloor p \rfloor}$ are the components of y sorted in non-increasing order.

We will consider piecewise linear interpolation function ω^* which is the simplest form of the required interpolation. The WOWA operator becomes the weighted mean if the preference weights are all equal (weight w) and it is reduced to the standard OWA operator if the importance weights are all equal (weight λ).

3. Choquet integral

The Choquet integral [5] generalizes all the aggregation operators defined before (WS, OWA and WOWA):

3.1. Definition

We first define the notion of capacity, on which the Choquet integral is based. We design by $\mathcal{P}$ the set of objectives $\{1,\dots,p\}$.

Definition 7. A capacity is a set function $v : 2^{\mathcal{P}} \to [0,1]$ such that:

- $v(\varnothing) = 0$
- $v(\mathcal{P}) = 1$
- $\forall \mathcal{A}, \mathcal{B} \in 2^{\mathcal{P}}$ such that $\mathcal{A} \subseteq \mathcal{B}, v(\mathcal{A}) \le v(\mathcal{B})$

Therefore, for each subset of objectives $\mathcal{A} \subseteq \mathcal{P}$, $v(\mathcal{A})$ represents the importance of the set $\mathcal{A}$.

Definition 8. The Choquet integral of a vector $y \in \mathbb{Z}^p$ for a capacity v is defined by

$$f_v^C(y) = \sum_{i=1}^{p} (v(Y_{\lceil i \rceil}) - v(Y_{\lceil i+1 \rceil}))y_{\lceil i \rceil}$$

$$= \sum_{i=1}^{p} (y_{\lceil i \rceil} - y_{\lceil i-1 \rceil})v(Y_{\lceil i \rceil}) \quad (1)$$

where $\lceil \cdot \rceil$ represents the permutation over $\{1,\dots,p\}$ such that $0 = y_{\lceil 0 \rceil} \le y_{\lceil 1 \rceil} \le \cdots \le y_{\lceil p \rceil}$, $Y_{\lceil i \rceil} = \{j \in \{1,\dots,p\}, y_j \ge y_{\lceil i \rceil}\} = \{\lceil i \rceil, \lceil i+1 \rceil, \dots, \lceil p \rceil\}$ for $i \le p$ and $Y_{\lceil p+1 \rceil} = \varnothing$.

An example of the isopreference lines of this operator is represented in Fig. 1, for two objectives. Contrary to the OWA operator, the isopreference lines are not symmetric according to the bisector of equation ($f_1(x) = f_2(x)$) separating the objective space.

We show below how the different aggregation operators (WS, OWA and WOWA) can be obtained from a Choquet integral:

- *WS*: If the capacity v is *additive*, that is $\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{P}$, $v(\mathcal{A} \cup \mathcal{B}) = v(\mathcal{A}) + v(\mathcal{B})$, then the Choquet integral of a vector $y \in \mathbb{Z}^p$ is equal to

$$f_v^C(y) = \sum_{i=1}^{p} (v(Y_{\lceil i \rceil}) - v(Y_{\lceil i+1 \rceil}))y_{\lceil i \rceil} = \sum_{i=1}^{p} v(\{i\})y_i$$

- *OWA*: If the capacity v is symmetric, that is $\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{P}$ such that $|\mathcal{A}| = |\mathcal{B}|, v(\mathcal{A}) = v(\mathcal{B})$, then the Choquet integral of a vector $y \in \mathbb{Z}^p$ is equal to

$$f_v^C(y) = \sum_{i=1}^{p} (v(Y_{\lceil i \rceil}) - v(Y_{\lceil i+1 \rceil}))y_{\lceil i \rceil} = \sum_{i=1}^{p} w_i y_{\lfloor i \rfloor}$$

  for a weight set w defined by: $\forall i \in \mathcal{P}$, $w_i = v(\mathcal{A}) - v(\mathcal{B})$ when $i = |\mathcal{A}| = |\mathcal{B}| + 1$.
- *WOWA*: If the capacity is defined from the function ω^*, then the Choquet integral of a vector $y \in \mathbb{Z}^p$ is equal to

$$f_v^C(y) = \sum_{i=1}^{p} (v(Y_{\lceil i \rceil}) - v(Y_{\lceil i+1 \rceil}))y_{\lceil i \rceil} = \sum_{i=1}^{p} \left(\omega^*\left(\sum_{j \in Y_{\lceil i \rceil}} \lambda_{\lfloor j \rfloor}\right) - \omega^*\left(\sum_{j \in Y_{\lceil i+1 \rceil}} \lambda_{\lfloor j \rfloor}\right)\right) y_{\lceil i \rceil}$$

In [29], Torra shows that all WOWA operators are Choquet integrals, but that the reversal is not always true, as shown in an example with three objectives. If only two objectives are considered, we can wonder whether there is a difference between the Choquet integral and the WOWA operator.

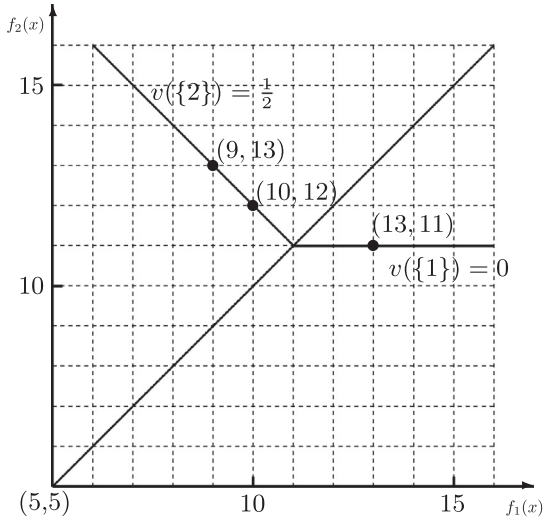

Fig. 1. Isopreference lines of the Choquet integral, for the point (10,12), with the capacity function $v(\{1\}) = 0$ and $v(\{2\}) = \frac{1}{2}$. For this capacity, the points (10,12), (13,11) and (9,13) have the same Choquet integral evaluations. We can also remark that the point (10,12) can never be better or equal than the points (13,11) and (9,13) if the OWA operator is used.

Property 1. *With two objectives, all Choquet integrals are WOWA operators.*

Proof. If the case of two objectives, from the equation above, we have

$$f_v^C(y) = \left(\omega^*\left(\sum_{j\in Y_{[1]}} \lambda_{[j]}\right) - \omega^*\left(\sum_{j\in Y_{[2]}} \lambda_{[j]}\right)\right) y_{[1]} + \left(\omega^*\left(\sum_{j\in Y_{[2]}} \lambda_{([j])}\right)\right) y_{[2]}$$

That is, if $y_1 \geq y_2$:

$$\begin{aligned} f_v^C(y) &= (\omega^*(\lambda_1+\lambda_2) - \omega^*(\lambda_1))y_2 + \omega^*(\lambda_1)y_1 \\ &= (1-\omega^*(\lambda_1))y_2 + \omega^*(\lambda_1)y_1 \\ &= y_2 + \omega^*(\lambda_1)(y_1 - y_2) \\ &= y_2 + v(\{1\})(y_1 - y_2) \end{aligned}$$

And if $y_1 < y_2$:

$$\begin{aligned} f_v^C(y) &= y_1 + \omega^*(\lambda_2)(y_2 - y_1) \\ &= y_1 + v(\{2\})(y_2 - y_1) \end{aligned}$$

We have to show that it is possible to generate $v(\{1\})$ and $v(\{2\})$, that are independent and included between [0,1], from $\omega^*(\lambda_1)$ and $\omega^*(\lambda_2)$, given that $\lambda_1 + \lambda_2 = 1$, and $w_1 + w_2 = 1$.

Let us suppose that $\lambda_1 \leq \lambda_2$ (that is $\lambda_1 \leq 0.5$):

$$\omega^*(\lambda_1) = 2w_1\lambda_1$$

$$\begin{aligned} \omega^*(\lambda_2) &= w_1 + 2(1-w_1)(\lambda_2 - 0.5) \\ &= w_1 + 2(1-w_1)(0.5-\lambda_1) \\ &= 1 - 2\lambda_1 + 2w_1\lambda_1 = 1 - 2\lambda_1 + \omega^*(\lambda_1) \end{aligned}$$

And if $\lambda_1 > \lambda_2$ (that is $\lambda_1 > 0.5$):

$$\omega^*(\lambda_2) = 2w_1\lambda_2 = 2w_1(1-\lambda_1)$$

$$\begin{aligned} \omega^*(\lambda_1) &= w_1 + 2(1-w_1)(\lambda_1 - 0.5) \\ &= 2\lambda_1 - 1 + \omega^*(\lambda_2) \end{aligned}$$

Therefore, the WOWA operator equivalent to the Choquet integral is obtained with:

If $v(\{1\}) \leq v(\{2\})$:

$$\lambda_1 = \frac{1 + v(\{1\}) - v(\{2\})}{2}, \quad w_1 = \frac{v(\{1\})}{1 + v(\{1\}) - v(\{2\})}$$

and if $v(\{1\}) \geq v(\{2\})$:

$$\lambda_1 = \frac{1 + v(\{1\}) - v(\{2\})}{2}, \quad w_1 = \frac{v(\{2\})}{1 + v(\{2\}) - v(\{1\})}$$

$\forall v(\{1\}) \in [0,1], v(\{2\}) \in [0,1]$, we have a corresponding $\lambda_1 \in [0,1]$ and a corresponding $w_1 \in [0,1]$, therefore all the capacities $v(\{1\})$ and $v(\{2\})$ can be obtained, and all Choquet integrals are WOWA operators, in the case of two objectives. □

For example, the Choquet integral obtained with $v(\{1\}) = \frac{3}{4}$ and $v(\{2\}) = \frac{1}{2}$ is equivalent to the WOWA operator obtained with $\lambda = (\frac{5}{8}, \frac{3}{8})$ and $w = (\frac{2}{3}, \frac{1}{3})$.

We have represented in Fig. 2 the different operators for two objectives in the capacity space (x-axis is $v(\{1\})$ and y-axis is $v(\{2\})$).

3.2. Example

In Table 1, we compare the points represented in Fig. 3. They are all P-optimal, but only some of them are WS, OWA, WOWA or C optimal.

We see that the C-optimal set is a subset of the P-optimal set. All the supported P-optimal solutions belong to the C-optimal set, but not all the non-supported P-optimal solutions. The WS optimal set and the OWA optimal set are a subset of the C-optimal set. The WOWA optimal set is equal to the C-optimal set in biobjective optimization. In this example, the point (10,12) is the only point which is not WS optimal or OWA optimal but C-optimal.

4. Generation of the C-optimal set

4.1. State-of-the-art

Generating the C-optimal solutions of MOCO problems is a recent research topic. We have only listed three previous works on this topic, none of them before 2009 [8–10]. Unlike the algorithm presented in this paper, these works only focus on the generation of one C-optimal solution given a capacity. Moreover, the problem of generating all the capacities that allow to generate the C-optimal set is not studied. On the other hand, the previous works are not limited to two objectives, as considered here.

In [10], Galand et al. search for C-optimal solutions of multiobjective spanning trees problems and multiobjective knapsack problems. They present a condition (named preference for interior points) that characterizes preferences favoring well-balanced solutions: the capacity v has to be *supermodular* (in the case of maximization of the objectives), that is $\forall \mathcal{A}, \mathcal{B} \in 2^{\mathcal{P}}$, $v(\mathcal{A}\cup\mathcal{B}) + v(\mathcal{A}\cap\mathcal{B}) \geq v(\mathcal{A}) + v(\mathcal{B})$. They investigate the generation of C-optimal solutions under this assumption. For both problems

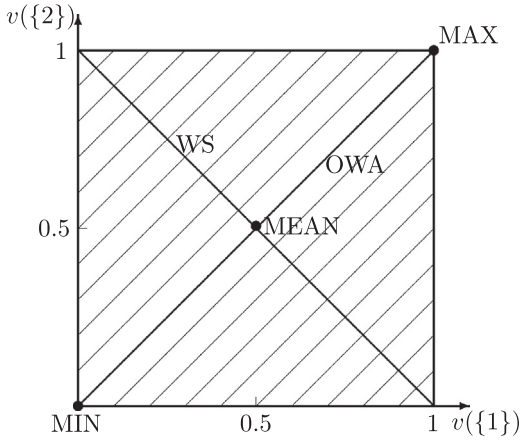

Fig. 2. Representation of the different operators, for two objectives, in the capacity space (that is, x-axis is $v(\{1\})$ and y-axis is $v(\{2\})$).

Table 1
Comparisons of the points of Fig. 3 according to the P-dominance and the different operators (WS, OWA, WOWA, Choquet).

$(f_1(x), f_2(x))$	P	WS	OWA	WOWA	C
(29,0)	1	1	1	1	1
(28,1)	1	0	0	0	0
(27,4)	1	1	1	1	1
(24,5)	1	0	0	0	0
(23,6)	1	0	0	0	0
(21,8)	1	1	1	1	1
(17,9)	1	0	0	0	0
(15,10)	1	0	0	0	0
(13,11)	1	0	1	1	1
(10,12)	1	0	0	1	1
(9,13)	1	1	0	1	1
(0,14)	1	1	0	1	1

studied, they introduce a linear bound of the Choquet integral and propose a branch and bound algorithm using this bound. They consider instances with a number of objectives equal to 3, 5 or 8.

In [9], Galand et al. are looking for *C*-optimal solutions, under the same assumption. Multiobjective spanning tree and shortest path problems are investigated. They use a branch and bound algorithm and an enumeration algorithm, based on a ranking approach, to solve the problems.

In [8], Fouchal et al. study the same problem: generating a *C*-optimal solution given a capacity. They apply their algorithm to the multiobjective shortest path problem. To find a *C*-optimal solution of the multiobjective shortest path problem, they introduce Choquet dominance rules that they integrate within the label setting algorithm of Martins [20], based on dynamic programming.

4.2. Characterization of the C-optimal solutions

The aim is to generate a *complete* *C*-optimal set called $\mathcal{X}_C$, containing at least one solution $x\in\mathcal{X}$ optimal for each possible Choquet integral, that is $\forall v\in\mathcal{V}, \exists x_c\in\mathcal{X}_C | f_c^v(f(x_c))\geq f_c^v(f(x))\ \forall x\in\mathcal{X}$, where $\mathcal{V}$ represents the set of capacity functions defined over two objectives (that is $v(\emptyset)=0$, $v(\{1\})=\alpha$, $v(\{2\})=\beta$ and $v(\{1,2\})=1$, with $\alpha\in[0,1]$ and $\beta\in[0,1]$).

Therefore, we do not guarantee to generate all the *C*-optimal solutions, but at least one solution optimal for each possible Choquet integral (for computational reasons, as in single-objective optimization where only one optimal solution is usually sought).

First note that a solution *C*-optimal is only weakly *P*-optimal:

Property 2. *A solution C-optimal is weakly P-optimal.*

$x_c\in\mathcal{X}_C \Rightarrow x_c\in\mathcal{X}_{wP}$

Proof. If $x_c\notin\mathcal{X}_{wP}$, there is $x\in\mathcal{X}$ such that $f(x)\succ_{sP} f(x_c)$, which implies that $f_c^v(f(x))>f_c^v(f(x_c))$, that is $x_c\notin\mathcal{X}_C$. Therefore $x_c\in\mathcal{X}_{wP}$. □

However, the case where *C*-optimal solutions are weakly *P*-optimal but not *P*-optimal only occurs if a Choquet integral with $(\alpha,\beta)=(0,1)$ or $(\alpha,\beta)=(1,0)$ is used. Indeed, in this case, only one objective is optimized and if x_1 and x_2 represent two different optimal solutions, we can have $f_c^v(f(x_1))=f_c^v(f(x_2))$ even if $f(x_2)\succ_P f(x_1)$. In the algorithm proposed to generate $\mathcal{X}_C$, this case will be avoided by always generating a *P*-optimal solution corresponding to an optimal solution of the Choquet integral with $(\alpha,\beta)=(0,1)$ or $(\alpha,\beta)=(1,0)$. Therefore, all the *C*-optimal solutions generated will be *P*-optimal.

We now propose a property that characterizes the *C*-optimal set.

Let $\mathcal{X}_{ws}$ be the set containing at least one solution $x_{ws}\in\mathcal{X}$ optimal for each WS defined through a weight vector λ $(\forall\lambda\in[0,1]^2\ \exists x_{ws}\in\mathcal{X}_{ws} |\ f_{ws}(f(x_{ws}))\geq f_{ws}(f(x))\ \forall x\in\mathcal{X})$.

Let us separate the decision space into two parts: the subspace $\mathcal{X}^+$ composed of the solutions $x\in\mathcal{X}$ such that $f_2(x)\geq f_1(x)$ and the subspace $\mathcal{X}^-$ composed of the solutions $x\in\mathcal{X}$ such that $f_2(x)<f_1(x)$. In order to avoid trivial cases, we will suppose in the following that both $\mathcal{X}^+$ and $\mathcal{X}^-$ are not empty.

We define f_1^+ and f_2^- the two values of $\mathbb{Z}$ such that $f_1^+=\max_{x\in\mathcal{X}^+}(f_1(x))$ and $f_2^+=\max_{x\in\mathcal{X}^-}(f_2(x))$. Let us denote m the (fictitious) solution in the decision space such that $f_1(m)=f_2(m)=\max(f_1^+,f_2^+)=\max_{x\in\mathcal{X}}\min(f_1(x),f_2(x))$.

We note $\mathcal{M}^+$ the set $\mathcal{X}^+\cup\{m\}$ and we note $\mathcal{M}^-$ the set $\mathcal{X}^-\cup\{m\}$.

Finally, let $\mathcal{M}_{ws}^+$ be the set containing at least one solution $x_{ws^+}\in\mathcal{X}^+$ optimal, among the search space $\mathcal{M}^+$, for each WS defined through a weight vector λ $(\forall\lambda\in[0,1]^2\exists x_{ws^+}\in\mathcal{M}_{ws}^+|\ f_{ws}(f(x_{ws^+}))\geq f_{ws}(f(x))\ \forall x\in\mathcal{M}^+)$. Let also $\mathcal{M}_{ws}^-$ be the set containing at least one solution $x_{ws^-}\in\mathcal{X}^-$ optimal, among the search space $\mathcal{M}^-$, for each WS defined through a weight vector λ $(\forall\lambda\in[0,1]^2\exists x_{ws^-}\in\mathcal{M}_{ws}^-|\ f_{ws}(f(x_{ws^-}))\geq f_{ws}(f(x))\ \forall x\in\mathcal{M}^-)$.

We have represented an example of these sets in Fig. 4.

We have then the following property which characterize $\mathcal{X}_C$:

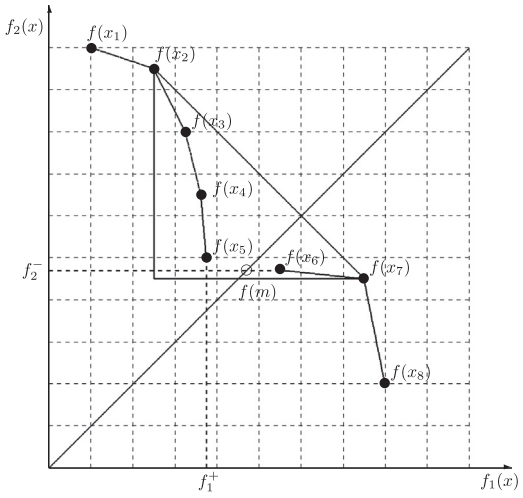

Fig. 4. $\mathcal{X}^+=\{x_1,x_2,x_3,x_4,x_5\}, \mathcal{X}^-=\{x_6,x_7,x_8\}$, $\mathcal{X}_{ws}=\{x_1,x_2,x_7,x_8\}, \mathcal{M}_{ws}^+=\{x_1,x_2,x_3\}, \mathcal{M}_{ws}^-=\{x_6,x_7,x_8\}$.

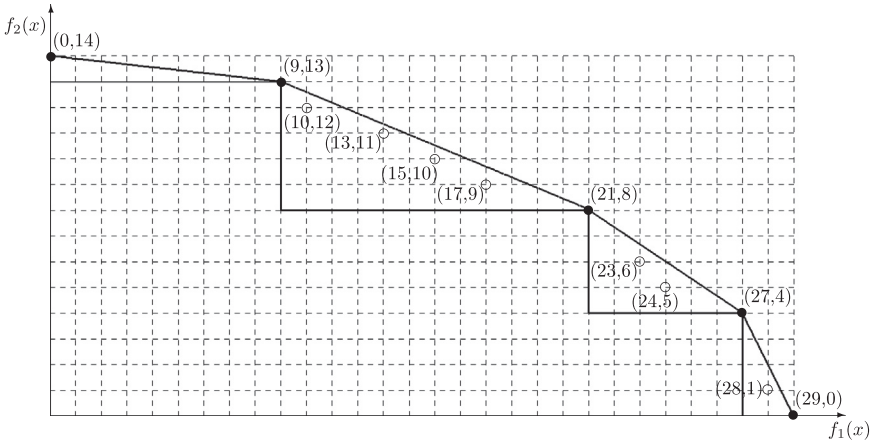

Fig. 3. Supported (represented by •) and non-supported (represented by ○) *P*-non-dominated points.

Property 3.

$\mathcal{X}_C = \mathcal{M}^+_{ws} \cup \mathcal{M}^-_{ws}$

Proof. In order to prove Property 3, we have to prove the following implications:

1. $x \in \mathcal{M}^+_{ws} \Rightarrow x \in \mathcal{X}_C$;
2. $x \in \mathcal{M}^-_{ws} \Rightarrow x \in \mathcal{X}_C$;
3. $x \in \{\mathcal{X}^+ \cap \mathcal{X}_C\} \Rightarrow x \in \mathcal{M}^+_{ws}$;
4. $x \in \{\mathcal{X}^- \cap \mathcal{X}_C\} \Rightarrow x \in \mathcal{M}^-_{ws}$.

Due to the symmetry of the situations, we present below only the proofs of points 1 and 3.

1. $\boldsymbol{x}^* \in M^+_{\boldsymbol{ws}} \Rightarrow \boldsymbol{x}^* \in X_{\boldsymbol{C}}$
   As $x^* \in \mathcal{M}^+_{ws}$, we have:

$$\exists \beta \in [0,1] | (1-\beta) f_1(x^*) + \beta f_2(x^*) \geq (1-\beta) f_1(x) + \beta f_2(x) \ \forall x \in \mathcal{M}^+ \quad (2)$$

   - We first show that $\forall x \in \mathcal{X}^+ \exists v \in \mathcal{V} | f^v_C(f(x^*)) \geq f^v_C(f(x))$. If we take $v(\{2\}) = \beta$, we have $f^v_C(f(x^*)) = (1-\beta) f_1(x^*) + \beta f_2(x^*)$ and $f^v_C(f(x)) = (1-\beta) f_1(x) + \beta f_2(x)$. Therefore, following (1), for this capacity v, $f^v_C(f(x^*)) \geq f^v_C(f(x)) \ \forall x \in \mathcal{X}^+$ ($\mathcal{X}^+ \subseteq \mathcal{M}^+$).
   - We show now that $\forall x \in \mathcal{X}^- \exists v \in \mathcal{V} | f^v_C(f(x^*)) \geq f^v_C(f(x))$. If we take $v(\{1\}) = \alpha = 0$, we have $f^v_C(f(x)) = \alpha f_1(x) + (1-\alpha) f_2(x) = f_2(x)$. And if we take $v(\{2\}) = \beta$, following (1), we have $f^v_C(f(x^*)) = (1-\beta) f_1(x^*) + \beta f_2(x^*) \geq (1-\beta) f_1(m) + \beta f_2(m) = f_2(m)$ as $f_1(m) = f_2(m)$. And by definition of m, we have $\forall x \in \mathcal{X}^- f_2(x) \leq f_2(m)$, therefore $f^v_C(f(x^*)) \geq f_2(m) \geq f_2(x) = f^v_C(f(x))$, $\forall x \in \mathcal{X}^-$.

   So $\forall x^* \in \mathcal{M}^+_{ws}$ there is a capacity $v : \{v(\varnothing) = 0,\ v(\{1\}) = 0,\ v(\{2\}) = \beta, v(\{1,2\}) = 1)$ such that $f^v_C(f(x^*)) \geq f^v_C(f(x))$, $\forall x \in \mathcal{X}^+$ and $\forall x \in \mathcal{X}^-$. Therefore $x^* \in \mathcal{X}_C$.
3. $\mathbf{x}^* \in \{X^+ \cap X_{\boldsymbol{C}}\} \Rightarrow \mathbf{x}^* \in M^+_{\boldsymbol{ws}}$.
   As $x^* \in \{\mathcal{X}^+ \cap \mathcal{X}_C\}$, there is a capacity $v : \{v(\varnothing) = 0, v(\{1\}) = \alpha, v(\{2\}) = \beta, v(\{1,2\}) = 1)$ such that $f^v_C(f(x^*)) \geq f^v_C(f(x)) \ \forall x \in \mathcal{X}$. We have to show that there exists $\beta^* \in [0,1]$ such that $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(x) + \beta^* f_2(x) \ \forall x \in \mathcal{M}^+$.
   - We first show that $\exists\, \beta^* \in [0,1] | (1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(x) + \beta^* f_2(x) \ \ \forall x \in \mathcal{X}^+$. As $f^v_C(f(x)) = (1-\beta) f_1(x) + \beta f_2(x)$ and $f^v_C(f(x^*)) = (1-\beta) f_1(x^*) + \beta f_2(x^*)$, and as $f^v_C(f(x^*)) \geq f^v_C(f(x))$, it is enough to take $\beta^* = \beta$ to have $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(x) + \beta^* f_2(x) \ \forall x \in \mathcal{X}^+$.
   - We show now that $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(m) + \beta^* f_2(m)$
     - Suppose that $f_1(m) = f_2(m) = f^+_1$. It means that there is $x' \in \mathcal{X}^+$ such that $f_1(x') = f_1(m)$ and $f_2(x') \geq f_2(m)$. As $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(x) + \beta^* f_2(x) \ \ \forall x \in \mathcal{X}^+$, we have $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(x') + \beta^* f_2(x')$ and as $(1-\beta^*) f_1(x') + \beta^* f_2(x') \geq (1-\beta^*) f_1(m) + \beta^* f_2(m)$, we obtain by transitivity $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(m) + \beta^* f_2(m)$.
     - Suppose that $f_1(m) = f_2(m) = f^-_2$. It means that there is $x' \in \mathcal{X}^-$ such that $f_2(x') = f_2(m)$ and $f_1(x') \geq f_1(m)$. As $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\alpha) f_2(x) + \alpha f_1(x) \ \ \forall x \in \mathcal{X}^-$, we have $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\alpha) f_2(x') + \alpha f_1(x')$ and as $(1-\alpha) f_2(x') + \alpha f_1(x') \geq (1-\alpha) f_2(m) + \alpha f_1(m)$, we obtain by transitivity $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\alpha) f_2(m) + \alpha f_1(m)$. Moreover, we have

$$(1-\alpha) f_2(m) + \alpha f_1(m) = f_2(m) \ (\text{as } f_1(m) = f_2(m)) = (1-\beta^*) f_1(m) + \beta^* f_2(m)$$

     Therefore $(1-\beta^*) f_1(x^*) + \beta^* f_2(x^*) \geq (1-\beta^*) f_1(m) + \beta^* f_2(m)$.

We have thus shown that if $x^* \in \mathcal{X}_C$, there is a $\beta^* \in [0,1]$ such that:
(a) $(1-\beta^*) f_1(x^*) + \beta f_2(x^*) \geq (1-\beta^*) f_1(x) + \beta^* f_2(x) \ \forall x \in \mathcal{X}^+$.
(b) $(1-\beta^*) f_1(x^*) + \beta f_2(x^*) \geq (1-\beta^*) f_1(m) + \beta^* f_2(m)$

which means that $x^* \in \mathcal{M}^+_{ws}$. □

4.3. Method to generate a C-optimal set

We want now to generate a C-optimal set $\mathcal{X}_C$. In order to do so, we need to generate the sets $\mathcal{M}^+_{ws}$ and $\mathcal{M}^-_{ws}$. For computational reasons, we first generate a set $\mathcal{X}_{ws}$. All the solutions of $\mathcal{X}_{ws}$ are C-optimal (as they optimize a WS) and included in $\mathcal{M}^+_{ws} \cup \mathcal{M}^-_{ws}$. We then generate the remaining C-optimal solutions.

4.3.1. Generation of $\mathcal{X}_{ws}$

The set $\mathcal{X}_{ws}$ can easily be obtained with the dichotomic method of Aneja and Nair [2]. This method consists in generating all the weight sets which make it possible to obtain a set of WS optimal solutions of a BOCO problem. Note that as the aim is to generate a set $\mathcal{X}_C$ containing at least one solution $x \in \mathcal{X}$ optimal for each possible Choquet integral, we do not need to generate all the WS optimal solutions but only a set $\mathcal{X}_{ws}$ containing at least one optimal solution for each possible WS (as a consequence, not all the non-extreme supported P-optimal solutions (located along the facets) will be generated).

We recall this method in Algorithm 1, containing Algorithm 2.

Algorithm 1. Generation of $\mathcal{X}_C$.

Parameters ↓: a BOCO problem, with $f_1(x)$ and $f_2(x)$ both objectives.
Parameters ↑: the set $\mathcal{X}_C$.
- -| Computation of one lexicographically optimal solution for $(f_1(x), f_2(x))$
`Solve` $\text{lexmax}_{x \in \mathcal{X}}(f_1(x), f_2(x))$ (1)
Let $\hat{x}_1$ one optimal solution of (1)
- -| Computation of one lexicographically optimal solution for $(f_2(x), f_1(x))$
`Solve` $\text{lexmax}_{x \in \mathcal{X}}(f_2(x), f_1(x))$ (2)
Let $\hat{x}_2$ one optimal solution of (2)
- -| Computation of $\mathcal{X}_{ws}$
$\mathcal{X}_{ws} = \{\hat{x}_1, \hat{x}_2\}$
`SolveRecursion`$(f(\hat{x}_1)\downarrow, f(\hat{x}_2)\downarrow, \mathcal{X}\downarrow, \mathcal{X}_{ws}\updownarrow)$
$\mathcal{X}_C = \mathcal{X}_{ws}$
- -| Computation of $f(m)$
`Solve` $f^+_1 = \max_{x \in \mathcal{X}^+} f_1(x)$
`Solve` $f^+_2 = \max_{x \in \mathcal{X}^-} f_2(x)$
$f_1(m) = f_2(m) = \max(f^+_1, f^+_2)$
- -| Computation of $\hat{x}_{ws^+}$ and $\hat{x}_{ws^-}$
`Solve` $\max_{x \in \{\mathcal{X}_{ws} \cap \mathcal{X}^+\}} f_1(x)$ (3)
Let $\hat{x}_{ws^+}$ one optimal solution of (3)
`Solve` $\max_{x \in \{\mathcal{X}_{ws} \cap \mathcal{X}^-\}} f_2(x)$ (4)
Let $\hat{x}_{ws^-}$ one optimal solution of (4)
- -| Computation of $\mathcal{X}_C$
`SolveRecursion`$(f(\hat{x}_{ws^+})\downarrow, f(m)\downarrow, \mathcal{X}^+\downarrow, \mathcal{X}_C\updownarrow)$
`SolveRecursion`$(f(m)\downarrow, f(\hat{x}_{ws^-})\downarrow, \mathcal{X}^-\downarrow, \mathcal{X}_C\updownarrow)$

First, the set $\mathcal{X}_{ws}$ is initialized with both lexicographic optimal solutions $\hat{x}_1$ and $\hat{x}_2$ corresponding respectively to $\text{lexmax}_{x \in \mathcal{X}}(f_1(x), f_2(x))$ and $\text{lexmax}_{x \in \mathcal{X}}(f_2(x), f_1(x))$. We use the method proposed by Przybylski et al. [25] to compute a lexicographic optimal solution. We first solve two BOCO problems: one by only considering the first objective; the weight set is this equal

to (1,0) (the optimal solution of this problem is called x_1) and one by only considering the second objective; the weight set is this equal to (0,1) (the optimal solution of this problem is called x_2). Then, for computing the lexicographic optimal solution $\hat{x}_1$ corresponding to $\text{lexmax}_{x\in\mathcal{X}}(f_1(x),f_2(x))$, we improve the second objective without degrading the first objective by solving the single-objective problem with a weight set defined by the normal to the line through $(f_1(x_1),f_2(x_1))$ and $(f_1(x_1)-1,\max f_2(x_2)+1)$, that is $\lambda=(\max f_2(x_2)+1-f_2(x_1),1)$, since we suppose that $\mathcal{Y}\subset\mathbb{N}^2$. The optimal solution $\hat{x}_1$ obtained is an optimal solution of $\text{lexmax}_{x\in\mathcal{X}}(f_1(x),f_2(x))$.

The second lexicographic optimal solution $\hat{x}_2$ corresponding to $\text{lexmax}_{x\in\mathcal{X}}(f_2(x),f_1(x))$ is found in the same way by using x_2, solution of the BOCO problem with only the second objective considered, and solving an additional single-objective problem with $\lambda=(1,\max f_1(x_1)+1-f_1(x_2))$.

Once both lexicographic optimal solutions have been generated, the dichotomic scheme starts and Algorithm 2, called `SolveRecursion`, is run.

Algorithm 2. `SolveRecursion`.

Parameters $\downarrow$: $f(x_r),f(x_s)$ (with $f_1(x_r)<f_1(x_s)$), $\mathcal{X}_r$
Parameters $\updownarrow$: $\mathcal{X}_{sr}$.
- -| Creation of a weight set λ normal to the line segment connecting $f(x_r)$ and $f(x_s)$
$\lambda_1=f_2(x_r)-f_2(x_s)$, $\lambda_2=f_1(x_s)-f_1(x_r)$
- -| Solve the WS single-objective problem
`Solve` $\max_{x\in\mathcal{X}_r}(\lambda_1 f_1(x)+\lambda_2 f_2(x))$ (3)
Let x_t one optimal solution of (3)
- -| Update of $\mathcal{X}_{sr}$
$\mathcal{X}_{sr}=\mathcal{X}_{sr}\cup\{x_t\}$
- -| Search for new optimal solutions
if $f(x_t)\cap\overline{f(x_r)f(x_s)}=\varnothing$ **then**
`SolveRecursion` $(f(x_r)\downarrow,f(x_t)\downarrow,\mathcal{X}_r\downarrow,\mathcal{X}_{sr}\updownarrow)$
`SolveRecursion` $(f(x_t)\downarrow,f(x_s)\downarrow,\mathcal{X}_r\downarrow,\mathcal{X}_{sr}\updownarrow)$
end if

The dichotomic scheme is initialized with both lexicographic points $f(\hat{x}_1)$ and $f(\hat{x}_2)$ ($f(x_r)=f(\hat{x}_1)$ and $f(x_s)=f(\hat{x}_2)$). We also specify the search space $\mathcal{X}_r$, in this case, $\mathcal{X}_r=\mathcal{X}$. Then, a single-objective problem with a weight set defined by the normal to the line through $f(x_r)$ and $f(x_s)$ is solved. The corresponding weight set is equal to $(f_2(x_r)-f_2(x_s),f_1(x_s)-f_1(x_r))$. The solution of this problem is called x_t.

Let us note the line segment joining two points y_r and y_s in $\mathbb{R}^2$, by $\overline{y_r y_s}$.

If $f(x_t)\cap\overline{f(x_r)f(x_s)}=\varnothing$, that is if $f(x_t)$ is not on the line segment joining both points $f(x_r)$ and $f(x_s)$, we start again the dichotomic scheme two times: with x_r and x_t, and with x_t and x_s as starting solutions of the `SolveRecursion` algorithm.

When the dichotomic scheme stops, the set $\mathcal{X}_{ws}$ has been obtained and the other C-optimal solutions have to be generated.

4.3.2. Generation of $\{\mathcal{M}^+_{ws}\cup\mathcal{M}^-_{ws}\}\backslash\mathcal{X}_{ws}$

We want now to generate the rest of the C-optimal solutions, that is the C-optimal solutions that do not optimize a WS. We have first to generate the fictitious point $f(m)$, with $f_1(m)=f_2(m)=\max(\max_{x\in\mathcal{X}^+}f_1(x),\max_{x\in\mathcal{X}^-}f_2(x))=\max_{x\in\mathcal{X}}\min(f_1(x),f_2(x))$.

To compute $f_1(m)$ and $f_2(m)$, we simply solve two single-objective problems: $f_1^+=\max_{\mathcal{X}^+}f_1(x)$ and $f_2^+=\max_{\mathcal{X}^-}f_2(x)$. We have then $f_1(m)=f_2(m)=\max(f_1^+,f_2^+)$.

Afterwards, we use the dichotomic scheme two times: firstly, from the point $f(\hat{x}_{ws^+})$, optimal for $\max_{x\in\{\mathcal{X}_{ws}\cup\mathcal{X}^+\}}f_1(x)$ (easily obtained as $\mathcal{X}_{ws}$ has been previously generated) and $f(m)$, and secondly, from $f(m)$ and the point $f(\hat{x}_{ws^-})$, optimal for $\max_{x\in\{\mathcal{X}_{ws}\cup\mathcal{X}^-\}}f_2(x)$. In the first case, the search space $\mathcal{X}_r$ is limited to $\mathcal{X}^+$ and in the second case to $\mathcal{X}^-$.

If we come back to Fig. 4, we will apply the dichotomic scheme from $f(x_2)$ and $f(m)$ (x_3 will be generated), and from $f(m)$ and $f(x_7)$ (x_6 will be generated).

5. Results

As examples, we experiment the algorithm on the biobjective knapsack problem, and then on the biobjective minimum spanning tree.

The computer used for the experiments has a Intel Core i7-950 with 3.07 GHz and 11.8 GB of RAM.

5.1. Biobjective knapsack problem

We have tested the algorithm on the classic version of the knapsack problem, with two objectives (BOKP). This problem has been widely studied in multiobjective optimization [18] (but essentially the generation of the P-optimal solutions). The BOKP is formulated as follows.

Given n items ($i=1,\ldots,n$) having one characteristic w^i—typically the weight—and two profits $c_k{}^i$ ($k=1,2$), some items should be selected to maximize the two total profits while not exceeding the knapsack capacity W.

The BOKP problem is formulated as follows:

$$\text{“max”}\quad f_k(x)=\sum_{i=1}^{n}c_k^i x_i \quad k=1,2$$

$$\text{subject to}\quad \sum_{i=1}^{n}w^i x_i\le W \quad x_i\in\{0,1\},\quad i=1,\ldots,n$$

where $x_i=1$ means that the item i is selected to be in the knapsack. It is assumed that all coefficients $c_k{}^i$, w^i and W are nonnegative.

We have used the instances of the BOKP generated by Bazgan et al. [3]. Four types of instances have been defined: random (type A: the profits and the weights are randomly generated), correlated (type B: the two profits are correlated and the weights are randomly generated), uncorrelated (type C: the two profits are uncorrelated and the weights are randomly generated) and uncorrelated with correlated weights (type D: the two profits are uncorrelated and the weights are correlated with the sum of the profits). For each type of instances and for each number of items, ten different instances are considered.

To solve the WS problems, we have used the `minknap` algorithm developed by Pisinger [23]. The constrained knapsack problems (that is with $f_1(x)\ge f_2(x)$ or with $f_2(x)\ge f_1(x)$) have been solved with CPLEX since we have not found particular methods to solve these problems.

The results are given in Table 2 for the four types of instances and for different number of items. Ten different instances are considered for each type of instances and for each number of items. We indicate the following values: the mean number of WS optimal solutions generated, the mean number of C-optimal solutions generated and the mean number of P-optimal solutions (obtained by Bazgan et al. [3]). We also indicate the CPU time, in seconds, needed to generate the WS optimal solutions, the C-optimal solutions (included the WS optimal solutions), and the P-optimal solutions (in this case, the CPU time corresponds to the CPU time of the method proposed by Bazgan et al. [3], on a 3.4 GHz computer with 3 GB of RAM).

We can observe from Table 2:

- The number of C-optimal solutions is only lightly higher than the number of WS optimal solutions. The difference (mean) is included between one and five. There are not significant

Table 2
Results for the biobjective knapsack problem.

Instances		#			CPU (s)		
Type	n	WS	Choquet	Pareto	WS	Choquet	Pareto
A	400	68.4	71.3	1713.3	0.007	0.739	307.10
	700	115.0	118.7	4814.8	0.014	1.537	5447.92
B	2000	38.8	39.8	477.7	0.016	0.483	251.06
	4000	76.4	78.4	1542.3	0.046	2.484	6773.26
C	300	66.5	69.9	2893.6	0.011	1.015	373.1
	500	111.9	115.9	7112.1	0.019	9.045	4547.98
D	100	35.5	38.7	1765.4	0.009	1.335	40.87
	200	63.8	68.0	5464.0	0.018	3.328	1145.92

differences between the four types of instances. However, the higher the number of items is, the higher the difference is. This small difference between the number of *C*-optimal solutions and WS optimal solutions can be explained by the fact that the solutions that are *C*-optimal but not WS optimal are only located in the small area of the objective space defined by the two consecutive supported *P*-non-dominated points located on each side of the bisector.

- The *C*-optimal solutions represent only a small part of the *P*-optimal solutions (between 1% and 9%).
- The CPU time to generate the *C*-optimal solutions is much less than the CPU time needed to generate all the *P*-optimal solutions (from 30 to 3500 times faster). But if we compare the CPU times to generate the *C*-optimal solutions and to generate the WS optimal solutions, we see that is much faster to generate only the WS optimal solutions (from 30 to 480 times faster). This can be explained by the fact that a particular solver has been used to solve the WS problems, while to obtain the additional *C*-optimal solutions, CPLEX is needed to solve the constrained knapsack problems (harder to solve than the WS problems).

We have seen through the BOKP that this new approach to generate the *C*-optimal solutions is efficient since the solutions are rapidly obtained comparing to the generation of the whole set of *P*-optimal solutions.

However, we will see in the following section, that is not true for all BOCO problems.

5.2. Biobjective minimum spanning tree problem

We apply now the algorithm to the biobjective spanning tree problem (BOMSTP). The BOMSTP is formulated as follows. We have a connected graph $G=(V,E)$, with $n=|V|$ vertices and $m=|E|$ edges, and the set $\mathcal{X}$ of feasible solutions is the set of spanning trees (a connected subgraph of G which contains no cycle and all vertices of G). Each edge is evaluated by two integer costs, c_1^e and c_2^e. The value of a spanning tree in the objective space is $f(x)=(f_1(x),f_2(x))$ where $f_k(x)=x_i c_k^{e_i}$ $(k=1,2)$, with $x_i=1$ if the edge e_i belongs to the tree and $x_i=0$ otherwise.

The single-objective MSTP can be easily solved in polynomial time with Prim's [24] or Kruskal [17] greedy algorithm. However, generating the *P*-optimal solutions of the BOMSTP has been proved $\mathcal{NP}$-Hard by Emelichev and Perepelitsa [7].

Also, to generate the C-optimal solutions that are not WS-optimal solutions, we need to solve the MSTP with an additional constraint ($f_1(x)\geq f_2(x)$ or $f_2(x)\geq f_1(x)$), that we will further call "constrained MSTP", which is $\mathcal{NP}$-Hard [1].

We have used three types of instances:

- Random: the costs c_1^e and c_2^e are randomly generated between 1 and 100. The graphs used are cliques (complete graphs).
- Grid: the costs c_1^e and c_2^e are also randomly generated between 1 and 100, but the graphs are not complete. The graphs form a grid, with $n=a^2$ vertices and with edges joining the neighboring vertices horizontally and vertically.
- Hard: these instances are particular instances introduced by Knowles et al. [16]. These instances are considered as hard because there are many *P*-non-dominated points located far away from any supported *P*-non-dominated points (in the middle of the Pareto front, see Fig. 5). To create these hard instances, three special vertices (labeled 0,1 and 2) are considered. We restrict the costs to lie in [0,100]. The costs used are the following: $c^{e(0,1)}=(\xi,\xi)$, $c^{e(0,2)}=(0,100-\xi)$, $c^{e(1,2)}=(100-\xi,0)$. The costs of all the other edges are randomly generated between ξ and η if $i,j>3$ and between $(100-\xi)$ and 100 if (i xor j)≤ 3, with $i,j\in V$, ξ a small positive value of the order of $1/n$ and $\xi<\eta\ll 100-\xi$. We have used $\xi=5$ and $\eta=20$.

To solve the MSTP, Prim's algorithm has been implemented. To solve the constrained MSTP, we have considered a single commodity flow formulation [19] of the minimum spanning tree problem, solved with the CPLEX solver (with the network algorithm of CPLEX). To generate the *P*-optimal solutions, we have used the ϵ-constraint method [14] (the related constrained problems are solved with the CPLEX solver from the single commodity flow formulation), for its simplicity, even if more efficient approaches have been developed [28]. However, with this method, we were not able to solve the random instances with 50 and 100 nodes, the grid instances with 100 nodes and the hard instances with 30 nodes (CPLEX ran out of memory while trying to solve the constrained MSTP).

The results are given in Table 3, for the three types of instances. We have used three different numbers of vertices for each instance. For each type of instances and for each number of items, ten different instances are considered.

For the random and grid instances, we can draw the same conclusion that with the BOKP: the number of *C*-optimal solutions is only slightly higher than the number of WS optimal solutions, but these additional solutions are much harder to compute.

For the hard instances, we see that the number of *C*-optimal solutions can be quite higher than the number of WS optimal solutions. But these additional solutions are harder to generate. For example, for the hard instances with 30 nodes, we only need 3 ms to generate all the WS optimal solutions, while we need 1494 s to generate all the *C*-optimal solutions (that is 498 000 times more, on average). We see thus that it will be necessary to develop specific approaches to solve the constrained MSTP to reduce the CPU time.

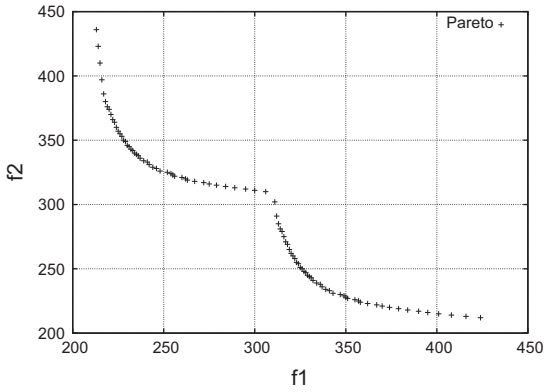

Fig. 5. Pareto front of a hard instance with 25 vertices.

We observe however through Fig. 6 the interest to generate the *C*-optimal solutions comparing to the generation of the WS optimal solutions: with the WS, no solution in the middle of the Pareto front is generated.

5.3. Additional remark

The optimization of a Choquet integral is an arduous task. Moreover, the elicitation of the parameters of a Choquet integral in order to represent a real-life preference relation is also difficult [11].

Therefore, it is worth to investigate whether the obtained optimal solutions for a given capacity could be obtained through a more simple optimization problem. In order to do so, we have investigated the whole space of parameters of a biobjective Choquet integral and we determine if the optimal solution for a given set of parameters of a capacity can also be simply obtained through a WS optimization or not.

We have represented the results in Fig. 7 for one instance of the BOKP (type C, 300 items) and for one instance of the BOMSTP (hard type, 25 items). The filled squares (■) represent capacities for which no solution is optimal for the Choquet integral and the WS, at the same time, while the empty squares (□) represent capacities for which there is a solution optimal for the Choquet integral and the WS.

We observe for both problems that the optimization of about 75% of the possible Choquet integrals leads to a solution which is also optimal for a WS. Therefore, the interest to try to elicitate a Choquet integral comparing to the use of the WS is not high if only two objectives are considered. In theory, the WS can be obtained from the Choquet integral if the capacity v is additive, which only represents a small area in the capacity space, as exposed in Fig. 2. But in practice, with two objectives, it seems that solutions that are WS optimal are also optimal for Choquet integral with capacities that are not additive.

We have only exposed the results of two instances, but similar results have been obtained for the other instances.

Table 3
Results for the biobjective minimum spanning tree problem.

Instances		#			CPU (s)		
Type	n	WS	Choquet	Pareto	WS	Choquet	Pareto
Random	25	40.8	42.8	219.7	0.006	7.18	460.87
	50	100.9	102.3	–	0.027	5.18	–
	100	216.5	218.5	–	0.2	303.67	–
Grid	25	12.2	13.7	43.4	0.002	0.72	5.54
	49	24.5	25.9	158.75	0.008	3.58	169.82
	100	56.4	58.6	–	0.049	21929.81	–
Hard	20	18.6	29.0	94.7	0.001	9.71	86.86
	25	27.2	37.6	112.4	0.001	91.97	271.51
	30	29.7	39.5	–	0.003	1494.22	–

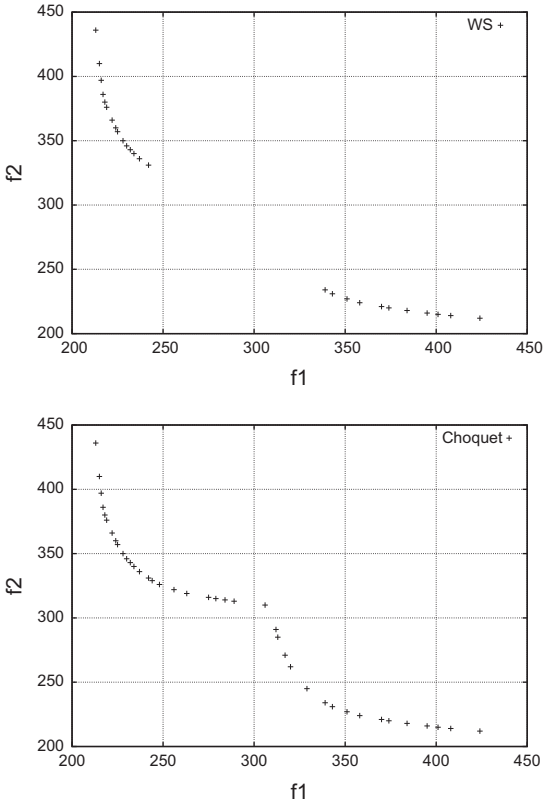

Fig. 6. Representation of the WS and *C*-optimal solutions of a hard instance with 25 vertices.

6. Conclusion and perspectives

We have introduced in this paper the first method to obtain all the *C*-optimal solutions of biobjective combinatorial problems. The method is based on a general property, and can easily be applied to any BOCO problems.

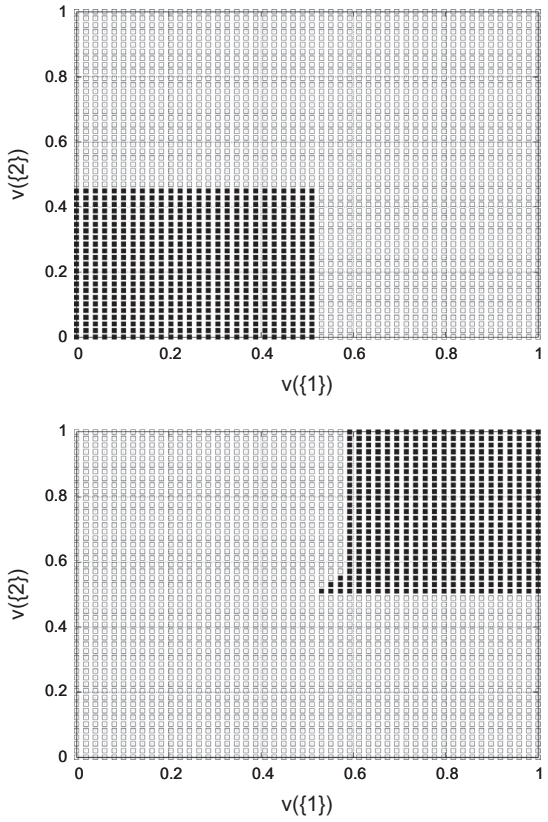

Fig. 7. Results obtained with one instance of the BOKP of type C, 300 items (above) and with one instance of the BOMSTP of hard type, with 25 nodes (below). The filled squares (■) represent capacities for which no solution is optimal for the Choquet integral and the WS, at the same time, while the empty squares (□) represent capacities for which there is a solution optimal for the Choquet integral and the WS.

Through this property, we have shown that the *C*-optimal solutions that are not optimal solutions of WS problems are located in a specific area of the objective space: between the two consecutive supported *P*-non-dominated points located on each side of the bisector.

The method has been applied to two BOCO problems: the biobjective knapsack problem and the biobjective minimum spanning tree problem. We have shown through these two problems that the number of *C*-optimal solutions is relatively low comparing to the number of *P*-optimal solutions. Also, computing the *C*-optimal solutions that are not WS optimal solutions is harder than computing WS optimal solutions since a single-objective problem with additional constraints has to be solved.

This first work about generating all the *C*-optimal solutions opens many perspectives:

- The property will have to be generalized to problems with more than two objectives. We think that the basis of the property will be the same, that is *C*-optimal solutions that are not WS optimal solutions will be located between the consecutive supported *P*-non-dominated points along the different bisectors of the objective space. The issue will be how to define the consecutive supported *P*-non-dominated points and the bisectors in a high-dimensional objective space.
- It will be also interesting to study and to define what brings exactly and concretely the *C*-optimal solutions that are not WS optimal solutions, given that they are harder to compute.
- We also have seen for different problems and instances that the optimization of the Choquet integral with about 75% of the possible values of the capacities leads to solutions which can also be obtained by simply optimizing a WS. It will be interesting to study how this percentage evolves with the number of objectives.
- Finally, through the biobjective spanning tree problem, we have shown that it can be very time-consuming to apply the general method developed in this paper. Therefore, specific methods to compute all the *C*-optimal solutions (branch and bound methods) or specific methods to optimize the single-objective problems with additional constraints could be studied.

References

[1] Aggarwal V, Aneja YP, Nair KPK. Minimal spanning tree subject to a side constraint. Computer & Operations Research 1982;4:287–96.
[2] Aneja YP, Nair KPK. Bicriteria transportation problem. Management Science 1979;25:73–8.
[3] Bazgan C, Hugot H, Vanderpooten D. Solving efficiently the 0–1 multi-objective knapsack problem. Computers & Operations Research 2009;36 (1):260–79.
[4] Bouyssou D, Dubois D, Pirlot M, Prade H, editors. Decision-making process concepts and methods. ISTE Wiley; 2009.
[5] Choquet G. Theory of capacities. Annales de l'Institut Fourier 1953(5):131–295.
[6] Ehrgott M. Multicriteria optimization. 2nd ed. Berlin: Springer; 2005.
[7] Emelichev VA, Perepelitsa VA. Multiobjective problems on the spanning trees of a graph. Soviet Mathematics Doklady, 1988;37:114–7.
[8] Fouchal H, Gandibleux X, Lehuédé F. Preferred solutions computed with a label setting algorithm based on Choquet integral for multi-objective shortest paths. In: 2011 IEEE symposium on computational intelligence in multicriteria decision-making (MDCM). Paris, France: IEEE; 2011. p. 143–50.
[9] Galand L, Perny P, Spanjaard O. Choquet-based optimisation in multiobjective shortest path and spanning tree problems. European Journal of Operational Research 2010;204(2):303–15.
[10] Galand L, Perny P, Spanjaard O. A branch and bound algorithm for Choquet optimization in multicriteria problems. In: Proceedings lecture notes in economics and mathematical systems, vol. 634. 2011. p. 355–65.
[11] Grabisch M, Kojadinovic I, Meyer P. A review of methods for capacity identification in Choquet integral based multi-attribute utility theory: applications of the kappalab R package. European Journal of Operational Research 2008;186(2):766–85.
[12] Grabisch M, Labreuche C. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. Annals OR 2010;175(1):247–86.
[13] Grabisch M, Marichal J-C, Mesiar R, Pap E. Aggregation functions. Encyclopedia of mathematics and its applications, vol. 127. Cambridge, UK: Cambridge University Press; 2009.
[14] Haimes Y, Lasdon L, Wismer D. On a bicriterion formulation of the problems of integrated system identification and system optimization. IEEE Transactions on Systems, Man, and Cybernetics 1971;1:296–7.
[15] Jorge J, Gandibleux X, Wiecek M. A priori reduction of the size of the binary multiobjective knapsack problem. In: Proceedings of the 8th international conference devoted to multi-objective programming and goal programming, Portsmouth, September 2008.
[16] Knowles JD, Watson RA, Corne DW. Reducing local optima in single-objective problems by multi-objectivization. In: Zitzler E, Deb K, Thiele L, Coello Coello CA, Corne D, editors. First international conference on evolutionary multi-criterion optimization, lecture notes in computer science, vol. 1993. Springer-Verlag; 2001. p. 268–82.
[17] Kruskal JB. On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical Society 1956;7 (February (1)):48–50.
[18] Lust T, Teghem J. The multiobjective multidimensional knapsack problem: a survey and a new approach. International Transactions in Operational Research 2012;19(4):495–520.
[19] Magnanti TL, Wolsey LA. Optimal Trees, 1994.
[20] Martins EQV. On a multicriteria shortest path problem. European Journal of Operational Research 1984;16(2):236–45.
[21] Ogryczak W. Inequality measures and equitable approaches to location problems. European Journal of Operational Research 2000;122(2):374–91.
[22] Ojalehto V, Miettinen K, Mäkelä MM. Interactive software for multiobjective optimization: IND-NIMBUS. WSEAS Transactions on Computers 2007;6:87–94.
[23] Pisinger D. A minimal algorithm for the 0–1 knapsack problem. Operations Research 1994;45:758–67.
[24] Prim RC. Shortest connection networks and some generalizations. Bell System Technology Journal 1957;36:1389–401.
[25] Przybylski A, Gandibleux X, Ehrgott M. Two-phase algorithms for the biobjective assignment problem. European Journal of Operational Research 2008;185(2):509–33.
[26] Przybylski A, Gandibleux X, Ehrgott M. A recursive algorithm for finding all nondominated extreme points in the outcome set of a multiobjective integer programme. INFORMS Journal on Computing 2010;22(3):371–86.
[27] Raith A, Ehrgott M. A two-phase algorithm for the biobjective integer minimum cost flow problem. Computers & Operations Research 2009;36 (6):1945–54.
[28] Sourd F, Spanjaard O, Perny P. Multi-objective branch and bound. Application to the bi-objective spanning tree problem. In: Proceedings of the multi-objective programming and goal programming conference (MOPGP'06), Tours, 2006.
[29] Torra V. On some relationships between the wowa operator and the Choquet integral. In: Proceedings of the seventh conference on information processing and management of uncertainty in knowledge-based systems, Paris, France, 1988. p. 818–24.
[30] Torra V. The weighted OWA operator. International Journal of Intelligent Systems 1997(12):153–66.
[31] Ulungu EL, Teghem J. The two-phases method: an efficient procedure to solve biobjective combinatorial optimization problems. Foundation of Computing and Decision Science 1995;20:149–56.
[32] Visée M, Teghem J, Pirlot M, Ulungu EL. Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem. Journal of Global Optimization 1998;12:139–55.
[33] Wolsey LA, Nemhauser GL. Integer and combinatorial optimization. Wiley-Interscience; November 1999.
[34] Yager RR. On ordered weighted averaging aggregation operators in multi-criteria decision making. IEEE Transactions on Systems, Man, and Cybernetics 1998;18:183–90.