Exact algorithms for OWA-optimization in multiobjective spanning tree problems

Lucie Galand, Olivier Spanjaard

Exact algorithms for OWA-optimization in multiobjective spanning tree problems

Lucie Galand[*,a], Olivier Spanjaard[b]

[a]*LAMSADE-CNRS, Université Paris-Dauphine, Place du Maréchal de Lattre de Tassigny, F-75775 Cedex 16 Paris, France*
[b]*LIP6-CNRS, Université Pierre et Marie Curie (UPMC), 4 Place Jussieu, F-75252 Paris Cedex 05, France*

Abstract

This paper deals with the multiobjective version of the optimal spanning tree problem. More precisely, we are interested in determining the optimal spanning tree according to an *Ordered Weighted Average* (OWA) of its objective values. We first show that the problem is weakly NP-hard. We then propose different mixed integer programming formulations, according to different subclasses of OWA functions. Furthermore, we provide various preprocessing procedures, the validity scopes of which depends again on the considered subclass of OWA functions. For designing such procedures, we propose generalized optimality conditions and efficiently computable bounds. These procedures enable to reduce the size of the instances before launching an off-the-shelf software for solving the mixed integer program. Their impact on the resolution time is evaluated on the basis of numerical experiments.

Key words: Multiobjective spanning tree problem, ordered weighted average, MIP formulation, optimality conditions, bounding procedure

1. Introduction

Multiobjective combinatorial optimization deals with problems involving multiple viewpoints [7]. More formally, the valuation structure of such problems is made of vectors (each component representing a specific viewpoint) instead of scalars. Most popular single objective optimization problems (e.g., valued graph problems, integer linear programming...) can be recasted in this setting, and solution algorithms must be proposed. Several types of approaches can be studied: either one aims at generating the whole set of Pareto solutions (i.e., solutions that cannot be improved on one objective without being depreciated on another one), also called the *P*areto set, or one focuses on a specific subset of the Pareto set (e.g., supported Pareto solutions, Lorenz optimal solutions [26]), or one looks for a best compromise solution according to a given aggregation function (e.g., max operator, Chebyshev's norm to a reference vector [48], ordered weighted average [49], Choquet integral [15]). Computing the Pareto set is natural when no preferential information is available or when the available information is unsufficient to elicit the parameters of the aggregation function: the solutions in the Pareto set are indeed the only ones likely to be selected by any rational decision maker. The interest in this approach has spawned a substantial literature (for a survey on the topic, the reader can refer to several quite recent papers [8, 9]). However, the number of Pareto solutions can grow exponentially with the size of the instance [e.g., 16, 17] and the number of objectives [39]. To overcome this difficulty, a first way to proceed could be to focus on a specific subset of the Pareto set, such as, for instance, the supported Pareto solutions. Those are the Pareto solutions that optimize linear scalarizations of the different objectives. When the single objective version of the studied problem is polynomially solvable, generating a supported solution can also be made in polynomial time. However, it is possible to exhibit instances for which even the number of supported solutions is exponential in the size of the instance [e.g., 16, 17]. Furthermore, focusing on supported Pareto solutions a priori excludes compromise solutions that could be preferred by the decision maker. For this reason, more involved decision criteria have been

[*]Corresponding author.
Email addresses: lucie.galand@dauphine.fr (Lucie Galand), olivier.spanjaard@lip6.fr (Olivier Spanjaard)

proposed in the field of multicriteria decision making [e.g., 23]. Consequently, for tractability and decision theoretic reasons, focusing on *o*ne particular compromise solution seems to be a better approach when an aggregation function is available. This enables indeed to considerably speed up the resolution procedure while preserving the decision maker's preferences. This is the approach we study in this paper.

More precisely, we investigate here the multiobjective version of the optimal spanning tree problem. This problem arises naturally in various contexts. For example, consider a broadcasting network where the values of the edges represent bandwidths. Assuming that the bandwidth of a chain is equal to the minimum bandwidth over its edges, it is well-known that, in a maximal spanning tree, the bandwidth between two nodes is the maximum possible. When there are several scenarios of traffic (impacting on the values of the bandwidths) or several opinions of experts on the values of the bandwidths, the problem becomes multiobjective. Previous works on the topic mainly deal with generating the whole Pareto set in the biobjective case [4, 16, 43, 44], or computing a min-max (regret) optimal solution when there are two or more objectives [3, 10, 16, 21, 47, 52]. To our knowledge, there is therefore no operational algorithmic tool for this problem when there are more than two objectives and when the min-max (regret) criterion is not really suitable. The present paper precisely aims at tackling this gap, by providing algorithms able to optimize a less conservative decision criterion for any number of objectives. Provided the required preferential information is available, and provided the objectives are commensurate (which is the case in the above example for instance), we propose to resort to an averaging operator to compare the vectorial values of the feasible solutions (spanning trees). According to the decision context, one may however want to put the emphasis on the best, the worst or the median evaluations of a solution. In other words, one needs to assign importance weights not to specific objectives (scenarios, experts), but rather to best and worst evaluations. The *Ordered Weighted Average* (OWA) precisely enables to model such concern. The optimization of this (non-linear) function has been widely studied in location problems under the name of *Discrete Ordered Median Problem* (DOMP) [e.g., 6, 24, 27, 28, 29]. A location problem typically involves a set of sites at which facilities can be located, and a set of demand points (clients) that request to be serviced from facilities. In DOMP, one aims at locating facilities so as to fairly satisfy the clients. In this concern, for evaluating a feasible solution, the OWA function (or ordered median function) is particularly suitable to aggregate the costs incurred by the different clients. Several mixed integer programming (MIP) formulations of the problem have been proposed in the literature [e.g., 6]. These formulations involve a linearization of the OWA function. In the case of decreasing weights (favouring well-balanced solutions), another way to linearize the OWA function has been studied by Ogryczak, Sliwinski and Tamir [31, 32, 33] in the field of continuous optimization under linear constraints.

We focus here on the *OWA-optimal spanning tree problem*, i.e. finding an optimal spanning tree according to the OWA function in a multiobjective spanning tree problem. We propose MIP formulations of the problem, together with preprocessing procedures that enable to reduce the size of the formulations before launching an off-the-shelf software. The paper is organized as follows. After recalling some preliminary definitions and stating the problem, we give some insights into its computational complexity (Section 2). In the next section, we present two alternative MIP formulations, one in the case where the OWA function is endowed with decreasing weights, and the other one in the case of arbitrary weights (Section 3). We then propose various preprocessing procedures to reduce the size of the MIP formulations (Section 4), based on generalized optimality conditions and/or efficiently computable bounds. Finally, we provide numerical experiments to assess the operationality of the proposed methods (Section 5).

2. Preliminaries

2.1. Multiobjective compromise search problem

A *multiobjective compromise search problem* is a problem endowed with vectorial costs where one searches for a best compromise solution according to a given aggregation function. A generic multiobjective compromise search problem can be formulated as a mathematical program. We now introduce some notations for this purpose. We denote by $X \subseteq \{0,1\}^m$ the set of feasible solutions, $f : X \to \mathbb{R}^p$ a vector-valued function on X, and $\varphi : \mathbb{R}^p \to \mathbb{R}$ a multiobjective aggregation function. Within this setting, a multiobjective compromise search problem is written:

$$(P) \quad \begin{cases} \min \ \varphi(y) \\ \text{s.t. } y = f(x) \\ x \in X \end{cases}$$

Denoting by $Y = f(X) = \{f(x) : x \in X\}$ the *image set* of X in the *objective space* $\mathbb{R}^p$, problem P can be simply reformulated as $\min_{y \in Y} \varphi(y)$. The OWA-optimal spanning tree problem (introduced in the following), consisting in looking for an optimal spanning tree according to the OWA function, belongs to this class of problems. The preprocessing methods we propose hereafter for this problem are generic. Mathematical formulation P will therefore be convenient to describe these methods in the sequel. We now more specifically introduce the OWA operator and then the OWA-optimal spanning tree problem.

2.2. The OWA operator

Given a set $\{1, \ldots, p\}$ of objectives (to minimize), one can associate a vector in $\mathbb{R}^p$ to every feasible solution of a multiobjective problem. The comparison of solutions amounts then to comparing the corresponding vectors. Following several works in multiobjective optimization [e.g., 29, 30, 35, 36], we propose to compare the vectors on the basis of their OWA value [49] (to minimize), defined as follows:

Definition 1. *Given a vector y in $\mathbb{R}^p$, its ordered weighted average is OWA*$(y)=\sum_{i=1}^{p} w_i y_{(i)}$*, where $\sum_{i=1}^{p} w_i = 1$ and $y_{(1)} \geq .. \geq y_{(p)}$ are the components of y sorted in non-increasing order.*

According to the decisional context, one can distinguish within the class of OWA operators several interesting subclasses depending on the definition of the weights:

- *decreasing weights*, i.e. $w_1 \geq \ldots \geq w_p \geq 0$: the set of weights naturally belongs to this class when performing robust discrete optimization. In *robust optimization* [21], the cost of a solution depends on different possible scenarios (states of the world). The aim is to find a *robust solution* according to this multiobjective representation, i.e. a solution that remains suitable whatever scenario finally occurs. The use of the OWA criterion with decreasing weights in this setting is justified since it can be characterized by a set of axioms that are natural for modelling robustness [35]. Compared to the max criterion, frequently used in robustness, the OWA criterion is less conservative since it enables trade-offs between several scenarios. Note however that the OWA criterion includes the max criterion as a special case, where one sets $w_1 = 1$ and $w_2 = \ldots = w_p = 0$. Another interesting special case is obtained for "big-stepped weights", i.e. when the gaps between successive weights are huge ($w_1 \gg \ldots \gg w_p$). The OWA criterion reduces then to the leximax operator, which consists in comparing two vectors on the basis of their greatest component, their second greatest one in case of equality on the first one, and so on... This criterion refines thus the max criterion by discriminating between vectors with the same value on the greatest component. Finally, the k-centra criterion [5] is also a well-identified special case of OWA optimization, where $w_1 = \ldots = w_k = 1/k$ and $w_{k+1} = \ldots = w_p = 0$. It consists in evaluating a vector according to the sum of its k greatest components.

- *arbitrary weights*: one of the most famous decision criterion in decision under complete uncertainty (i.e., when several states of the world can occur, and no information is available about their plausibilities) is the Hurwicz criterion, that enables to model intermediate attitudes towards uncertainty (i.e., neither desperately pessimistic nor outrageously optimistic) by performing a linear combination of the maximum possible value of a solution under the different scenarios, and the minimum possible one. More formally, if y is the image of a solution in the objective space, then its value according to the Hurwicz criterion is: $\alpha \max_i y_i + (1 - \alpha) \min_i y_i$. The Hurwicz criterion is clearly a special case of the OWA criterion, obtained by setting $w_1 = \alpha$, $w_p = 1 - \alpha$ and $w_i = 0 \ \forall i \neq 1, p$. As soon as there are more than two scenarios and $\alpha \notin \{0, 1\}$, these sets of weights are neither non-increasing nor non-decreasing. Another natural decision context where the sequence of weights is arbitrary happens when every component represents the opinion of a particular expert, and one wants to dismiss the extreme opinions. For instance, one can set $w_1 = 0$, $w_p = 0$ and $w_i = 1/(p-2) \ \forall i \neq 1, p$. The k-trimmed mean, used in location problems, is a generalization of this kind of criterion, where one sets $w_1 = \ldots = w_k = 0$, $w_{k+1} = \ldots = w_{p-k} = 1/(p-2k)$ and $w_{p-k+1} = \ldots = w_p = 0$.

Not only do these subclasses have different decision theoretic meanings, but also different algorithmic properties. Consequently we distinguish these two cases in the sequel of the paper. For the convenience of the reader, we denote by $\text{OWA}^{\downarrow}$ the subclass of OWA operators endowed with decreasing weights.

2.3. Problem and complexity

The problem we study in this paper is the OWA-optimal spanning tree problem, that can be formulated as follows:

OWA-optimal spanning tree problem (OWA-ST)
Input: a finite connected graph $G = (V, E)$,
p integer valuations v_j^k for every edge $e^k \in E$ ($j = 1, \ldots, p$ and $k = 1, \ldots, m$),
a set of weights w_i ($i = 1, \ldots, p$) for the OWA criterion;
Goal: determine a spanning tree $T^* \in \arg\min_{T \in \mathcal{T}} \text{OWA}(f(T))$,
where $\mathcal{T}$ is the set of spanning trees in G and $f(T) = (\sum_{k=1}^m v_1^k, \ldots, \sum_{k=1}^m v_p^k)$.

Coming back to generic formulation P of a multiobjective compromise search problem, problem OWA-ST corresponds to the following specifications: a spanning tree T is characterized by a vector $x = (x^1, \ldots, x^m) \in X$ where $x^k = 1$ if edge e^k belongs to T, and its cost is $f(x) = (f_1(x), \ldots, f_p(x))$ where $f_j(x) = \sum_{k=1}^m v_j^k x^k$. The aggregation function is of course defined by $\varphi(y) = \text{OWA}(y)$. We now introduce a small instance of problem OWA-ST that will be used as a running example in the sequel of the paper.

Example 1. *Consider a complete graph G with 4 vertices, and assume that the edges have been evaluated according to 3 objectives (scenarios, experts...). The set of vertices is $V = \{1, 2, 3, 4\}$ and the valuations of the edges are the following: $v^{[1,2]} = (3, 2, 3)$, $v^{[1,3]} = (4, 3, 1)$, $v^{[1,4]} = (1, 2, 2)$, $v^{[2,3]} = (2, 4, 1)$, $v^{[2,4]} = (2, 6, 1)$, $v^{[3,4]} = (1, 5, 1)$, where the valuation of edge e is denoted by v^e. A minimum spanning tree is $T_1 = \{[1, 4], [2, 3], [3, 4]\}$ according to the first dimension and the arithmetic mean, $T_2 = \{[1, 2], [1, 3], [1, 4]\}$ according to the second dimension, and $T_3 = \{[1, 3], [2, 3], [3, 4]\}$ according to the third dimension. None of them is however completely satisfying: either it is too much unbalanced ($f(T_1) = (4, 11, 4)$ and $f(T_3) = (7, 12, 3)$), or it is too much conservative ($f(T_2) = (8, 7, 6)$). Spanning tree $T_4 = \{[1, 2], [1, 4], [2, 3]\}$ (with $f(T_4) = (6, 8, 6)$) presents none of the drawbacks of the previous trees, and is OWA-optimal when setting for instance $w_1 = 0.5$, $w_2 = 0.3$ and $w_3 = 0.2$. The overall ranking according to OWA on $T_1, \ldots, T_4$ is indeed: 1) T_4 with OWA$(6, 8, 6) = 7$, 2) T_2 with OWA$(8, 7, 6) = 7.3$, 3) T_1 with OWA$(4, 11, 4) = 7.5$, 4) T_3 with OWA$(7, 12, 3) = 8.7$.*

As already indicated above, when $w_1 = 1$ and all other weights w_i's are equal to zero, the OWA criterion reduces to the max criterion. It has been proved NP-hard to compute an optimal solution with respect to the max criterion in a multiobjective spanning tree problem [16, 52]. Consequently, problem OWA-ST is also NP-hard in the general case. Note however that it is polynomially solvable for some subclasses of instances:

- if $w_1 = w_2 = \ldots = w_p$ (arithmetic mean), an optimal solution can be obtained by valuing every edge e^k by $\sum_{j=1}^p v_j^k$, and then applying a standard minimum spanning tree algorithm (e.g., Prim's algorithm or Kruskal's algorithm);
- if $w_p = 1$ and $w_i = 0\ \forall i \neq p$ (min criterion), an optimal solution can be obtained by solving p standard minimum spanning tree problems (one for each objective) and then returning the optimal one among them according to OWA;
- if there exists a permutation π of objectives such that $v_{\pi(1)}^k \geq \ldots \geq v_{\pi(p)}^k$ for every edge e^k, an optimal solution can be obtained by valuing every edge e^k by $\sum_{i=1}^p w_i v_{\pi(i)}^k$, and then applying a standard minimum spanning tree algorithm.

Note that these types of polynomial instances are quite uncommon. Nevertheless, one can design a *Fully Polynomial Time Approximation Scheme* (FPTAS) that works for any instance of OWA-ST. For this purpose, in the way Aissi et al. [2] designed an FPTAS for the max criterion, one can enrich the FPTAS proposed by Papadimitriou and Yannakakis [34] for approximating the OWA-optimal spanning tree. The FPTAS of Papadimitriou and Yannakakis indeed returns a set $\mathcal{T}_\varepsilon$ of spanning trees, the cardinality of which is polynomial in the size of the instance. For every spanning tree T in the graph, there exists $T_\varepsilon \in \mathcal{T}_\varepsilon$ such that $f_j(T_\varepsilon) \leq (1 + \varepsilon) f_j(T)$ for $j = 1, \ldots, p$. Note that $[y_j \leq y'_j \quad \forall j] \Rightarrow \text{OWA}(y) \leq \text{OWA}(y')$ [11], and that $\text{OWA}((1 + \varepsilon)y) = (1 + \varepsilon)\text{OWA}(y)$. Thus, a spanning tree T_ε^* which is OWA-optimal among the trees of $\mathcal{T}_\varepsilon$ satisfies $\text{OWA}(f(T_\varepsilon^*)) \leq (1+\varepsilon)\text{OWA}(f(T^*))$, where T^* is an OWA-optimal spanning tree in $\mathcal{T}$. We have therefore an FPTAS for problem OWA-ST (the algorithm is indeed polynomial since the search for T_ε^* is performed in a set of polynomial cardinality). However, the efficiency of the FPTAS quickly decreases with the number of

objectives since its complexity is exponential in this parameter. Note that another FPTAS [1, 18] has been proposed, based on the matrix-tree theorem, but is specific to the biobjective case. To our knowledge, no other FPTAS can be found in the literature for this problem. In this paper, we propose *exact* and operational procedures (the complexity of which is exponential in the worst case) for problem OWA-ST, based on linearizations of the OWA function. The validity of these linearizations depends on the subclass of weights used in the OWA criterion: decreasing or arbitrary.

3. MIP formulations of OWA-ST

We present below two MIP formulations of OWA-ST, one taking advantage of the decreasingness of the weights in the $\text{OWA}^{\downarrow}$ function, the other one remaining valid whatever are the weights used. In the way of Yaman et al. [51] for the robust spanning tree problem with interval data, we start from a compact mixed integer programming (MIP) formulation of the minimum spanning tree problem proposed by Magnanti and Wolsey [22]. We recall this formulation in the following section.

3.1. Non-linear formulation of OWA-ST

In the formulation proposed by Magnanti and Wolsey [22], the minimum spanning tree problem is considered as a special version of a flow problem. Every edge $[i, j]$ is replaced by two opposite arcs (i, j) and (j, i). The set of such arcs is denoted by A in the sequel. Some vertex of the graph — say 1 — is selected as a source, and $n-1$ units of flow are incoming into it (where $V = \{1, \ldots, n\}$). Furthermore, 1 unit of flow is outgoing of every vertex $i \neq 1$. To each feasible flow in the directed graph, one can associate a connected partial graph by selecting edge $e^k = [i, j]$ as soon as there is at least one unit of flow on (i, j) or (j, i). By imposing that the number of selected edges is $n-1$, one obtains a spanning tree. Let φ_{ij} denote the flow on arc (i, j), and x^k denote the boolean variable taking value 1 if $\varphi_{ij} > 0$ or $\varphi_{ji} > 0$ (for $e^k = [i, j]$). The constraints defining the set X_{ST} of feasible spanning trees are the following:

$$
\begin{aligned}
X_{\text{ST}} = \Bigg\{ x \in \{0,1\}^m \ : \quad & \sum_{(i,j)\in A} \varphi_{ij} - \sum_{(j,i)\in A} \varphi_{ji} = \begin{cases} n-1 & \text{if } i = 1, \\ -1 & \forall i \in V \setminus \{1\}, \end{cases} \\
& \varphi_{ij} \leq (n-1)x^k \quad \forall e^k = [i, j], \\
& \varphi_{ji} \leq (n-1)x^k \quad \forall e^k = [i, j], \\
& \sum_{k=1}^{m} x^k = n-1, \\
& \varphi_{ij} \geq 0 \text{ and } \varphi_{ji} \geq 0 \quad \forall [i, j] \in E \Bigg\},
\end{aligned}
$$

where $x = (x^1, \ldots, x^m)$ is the characterizing vector of a spanning tree, as defined in Section 2.3. In order to obtain a mathematical programming formulation of OWA-ST, it remains to define the objective function. For this purpose, one introduces variables $y_j = \sum_{k=1}^{m} v_j^k x^k$ ($j \in \{1, \ldots, p\}$) corresponding to the value of spanning tree x on the j^{th} component. It yields the program:

$$
(NLP_{\text{OWA}}) \begin{cases} \min \ \text{OWA}(y) \\ \text{s.t. } y = f(x) \\ x \in X_{\text{ST}} \end{cases}
$$

Nevertheless, this formulation is not linear due to the non-linearity of the objective function.

3.2. Linearization with decreasing weights

The decreasingness of the weights implies the convexity of the $\text{OWA}^{\downarrow}$ function. It is well-known that convexity of the objective function is a suitable property in minimization problems. In particular, this property makes it possible to linearize the $\text{OWA}^{\downarrow}$ function without introducing new binary variables. Such a linearization, specifically dedicated to decreasing weights, has been studied by Ogryczak, Sliwinski and Tamir [32, 33]. The key idea of this linearization is to use the Lorenz vector of y, the i^{th} component of which is defined, in a minimization setting, by $L_i(y) = \sum_{j=1}^{i} y_{(j)}$. This notion has been introduced in economics for inequality comparisons (in a maximization setting), where y is seen as an income distribution [e.g., 26]. By noting that $\text{OWA}^{\downarrow}(y) = \sum_{i=1}^{p} (w_i - w_{i+1}) L_i(y)$, where $w_{p+1} = 0$, it appears that

function $\text{OWA}^{\downarrow}$ is linear in the components of the Lorenz vector. Furthermore, component $L_i(y)$ is the solution of linear program P_i^y defined below. However, when y is a variable, program P_i^y is not linear anymore. To overcome this difficulty, it is worth considering the dual version D_i^y, where d_j^i (resp. r_i) are the dual variables for the inequality (resp. equality) constraints:

$$(P_i^y)\begin{cases} \max & \sum_{j=1}^{p} \alpha_j^i y_j \\ \text{s.t.} & \sum_{j=1}^{p} \alpha_j^i = i, \\ & 0 \le \alpha_j^i \le 1 \quad \forall j \in \{1,\dots,p\}. \end{cases} \qquad (D_i^y)\begin{cases} \min & ir_i + \sum_{j=1}^{p} d_j^i \\ \text{s.t.} & r_i + d_j^i \ge y_j \quad \forall j \in \{1,\dots,p\}, \\ & d_j^i \ge 0 \quad \forall j \in \{1,\dots,p\}. \end{cases}$$

Replacing $L_i(y)$ by program D_i^y for $i = 1,\dots,p$, program NLP_{OWA} becomes linear:

$$(P_{\text{OWA}^{\downarrow}})\begin{cases} \min & \displaystyle\sum_{i=1}^{p}(w_i - w_{i+1})(ir_i + \sum_{j=1}^{p} d_j^i) \\ \text{s.t.} & r_i + d_j^i \ge y_j \quad \forall i,j \in \{1,\dots,p\}, \\ & y_j = \sum_{k=1}^{m} v_j^k x^k \ \forall i \in \{1,\dots,p\}, \\ & d_i^j \ge 0 \quad \forall i,j \in \{1,\dots,p\} \text{ and } r_i \text{ unrestricted}, \\ & x \in X_{\text{ST}}. \end{cases}$$

Note that this formulation is valid if $w_i - w_{i+1} \ge 0$ for $i = 1,\dots,p$, which is the case here since the w_i's are assumed to be decreasing. With m edges in the graph, the program involves $p^2+n+2m+1$ constraints and p^2+p+3m variables (variables y_i's can be omitted in the implementation): its size is therefore linear in the size of the input for a fixed number of objectives.

3.3. Linearization with arbitrary weights

In the case of arbitrary weights, we linearize the OWA operator in the way of Boland et al. [6] for the discrete ordered median problem (DOMP). We transpose the linearization of DOMP in our setting by using the following variables:

$$s_{ij} = \begin{cases} 1 & \text{if objective } j \text{ is the } i^{th} \text{ highest one in spanning tree } x, \\ 0 & \text{otherwise} \end{cases}$$

for $i = \{1,\dots,p\}$ and $j = \{1,\dots,p\}$. We also introduce variables $z_{ijk} = s_{ij}x^k$, so $z_{ijk} = 1$ if objective j is the i^{th} highest one and edge e^k belongs to spanning tree x, and 0 otherwise. The linearization of OWA-ST can then be formulated as follows:

$$(P_{\text{LOM}})\begin{cases} \min \displaystyle\sum_{i=1}^{p} w_i (\sum_{j=1}^{p}\sum_{k=1}^{m} v_j^k z_{ijk}) & \\ \text{s.t.} \displaystyle\sum_{i=1}^{p} s_{ij} = 1 \qquad \forall j \in \{1,\dots,p\}, & (1a) \\ \displaystyle\sum_{j=1}^{p} s_{ij} = 1 \qquad \forall i \in \{1,\dots,p\}, & (1b) \\ \displaystyle\sum_{j=1}^{p}\sum_{k=1}^{m} v_j^k z_{ijk} \ge \sum_{j=1}^{p}\sum_{k=1}^{m} v_j^k z_{(i+1)jk} \qquad \forall i \in \{1,\dots,p-1\}, & (1c) \\ \displaystyle\sum_{i=1}^{p} z_{ijk} = x^k \qquad \forall j \in \{1,\dots,p\}, \forall k \in \{1,\dots,m\}, & (1d) \\ \displaystyle\sum_{k=1}^{m} z_{ijk} = (n-1)s_{ij} \qquad \forall i \in \{1,\dots,p\}, \forall j \in \{1,\dots,p\}, & (1e) \\ s_{ij} \in \{0,1\},\ z_{ijk} \ge 0 \qquad \forall i \in \{1,\dots,p\}, \forall j \in \{1,\dots,p\}, \forall k \in \{1,\dots,m\}, & \\ x \in X_{\text{ST}}. & \end{cases}$$

Every instanciation of variables s_{ij} compatible with constraints 1a and 1b defines some permutation of the objectives. For intance, assume that $p = 2$ and objective 2 is higher than objective 1 for spanning tree x, then the values of variables s_{ij}'s should be: $s_{11} = s_{22} = 0$ and $s_{12} = s_{21} = 1$. Constraint 1c ensures that variables s_{ij} characterize the ranking of the objectives in spanning tree x. Constraints 1d and 1e make it possible to linearize contraints $z_{ijk} = s_{ij}x^k$. Note that if edges e^k and $e^{k'}$ belong to spanning tree x then $z_{ijk} = z_{ijk'}$ for all i and j. Consequently program P_{LOM} can be directly improved by defining variables $y_{ij} = \sum_{k=1}^{m} v_j^k z_{ijk}$. These variables y_{ij} denote the value of spanning tree x on objective j if objective j is the i^{th} highest one, and are equal to 0 otherwise. The formulation with y variables instead of z variables is thus as follows:

$$
(P_{\text{OWA}}) \begin{cases}
\min \displaystyle\sum_{i=1}^{p} w_i \Big(\sum_{j=1}^{p} y_{ij}\Big) & \\
\text{s.t.} \displaystyle\sum_{i=1}^{p} s_{ij} = 1 \qquad \forall j \in \{1,\dots,p\}, & \text{(2a)} \\
\displaystyle\sum_{j=1}^{p} s_{ij} = 1 \qquad \forall i \in \{1,\dots,p\}, & \text{(2b)} \\
\displaystyle\sum_{j=1}^{p} y_{ij} \geq \sum_{j=1}^{p} y_{(i+1)j} \qquad \forall i \in \{1,\dots,p-1\}, & \text{(2c)} \\
y_{ij} \leq M s_{ij} \qquad \forall i \in \{1,\dots,p\}, \forall j \in \{1,\dots,p\}, & \text{(2d)} \\
\displaystyle\sum_{i=1}^{p} y_{ij} = \sum_{k=1}^{m} v_j^k x^k \qquad \forall j \in \{1,\dots,p\}, & \text{(2e)} \\
s_{ij} \in \{0,1\},\ y_{ij} \geq 0 \qquad \forall i \in \{1,\dots,p\}, \forall j \in \{1,\dots,p\}, & \\
x \in X_{\text{ST}}. &
\end{cases}
$$

where constant M denotes a value greater than all possible values of the objective functions. Constraint 2d ensures that, for a given objective j, one and only one variable y_{ij} can be strictly positive, and constraint 2e ensures that the value of this latter variable is precisely the value of spanning tree x on objective j. With n vertices and m edges in the graph, program P_{OWA} involves $p^2 + m$ binary variables, $p^2 + 2m$ real variables, and $p^2 + 4p + n + 2m$ constraints.

In order to further reduce the number of variables and constraints involved in $P_{\text{OWA}^\downarrow}$ and P_{OWA}, we now provide preprocessing methods.

4. Preprocessing the formulations

In this section, we first propose generalized optimality conditions that only hold for the OWA$^\downarrow$ subclass. We then propose efficiently computable bounds that allow us to identify some edges as mandatory or forbidden by applying a procedure called *shaving* (detailled in the following). These preprocessing procedures make it possible to significantly reduce the density of the graph prior to the resolution.

As often done when presenting algorithms for constructing spanning trees [e.g., 45], we describe our preprocessing method as an edge coloring process. Initially all edges are uncolored. We color one edge at a time either blue (mandatory) or red (forbidden). At each step of the preprocessing method, we have therefore a partial coloring $c(\cdot)$ of the edges of G. For a partial coloring c, we denote by $\mathcal{T}(c)$ the set of trees including all blue edges, some of the uncolored ones (possibly none), and none of the red ones. The aim of the preprocessing method is to color as many edges as possible, so as to get a maximal coloring c such that $\min_{T\in\mathcal{T}(c)} \text{OWA}(f(T)) = \min_{T\in\mathcal{T}} \text{OWA}(f(T))$.

4.1. Generalized optimality conditions for OWA$^\downarrow$

We now give conditions under which an edge can be colored blue or red, which are adaptations of the well-known *cut optimality condition* and *cycle optimality condition* to our problem.

Before introducing the optimality conditions, we recall some definitions from graph theory. A *cut* in a graph is a partition of the vertices into two disjoint sets and a *crossing edge* (with respect to a cut) is one edge that connects a vertex in one set to a vertex in the other one. When there is no ambiguity, the term *cut* will also be used to refer to the set of crossing edges defined by the partition of the vertices. Thanks to these definitions, we can now formulate the generalized optimality conditions (we recall that v^k denotes the cost vector of edge e^k):

Proposition 1 (optimality conditions)**.** *Let G be a connected graph with coloring c of the edges. The following properties hold:*
(i) Cut optimality condition. *Let us consider a cut C in G with no blue edge. If there is some uncolored edge $e^k \in C$ such that $OWA^{\downarrow}(v^k - v^{k'}) \leq 0$ for all uncolored edges $e^{k'} \in C$, then e^k belongs to an $OWA^{\downarrow}$-optimal tree in $\mathcal{T}(c)$.*
(ii) Cycle optimality condition. *Let us consider a cycle C in G containing no red edge. If there is some uncolored edge $e^k \in C$ such that $OWA^{\downarrow}(v^{k'} - v^k) \leq 0$ for all uncolored edges $e^{k'} \in C$, then e^k can be colored red without changing value* $\min_{T \in \mathcal{T}(c)} OWA^{\downarrow}(f(T))$.

Proof. The proof relies on the following property of function $\mathrm{OWA}^{\downarrow}$ with non-increasing weights: $\mathrm{OWA}^{\downarrow}(y - y') \geq \mathrm{OWA}^{\downarrow}(y) - \mathrm{OWA}^{\downarrow}(y')$. To prove this property, one uses the convexity of function $\mathrm{OWA}^{\downarrow}$, which follows from the fact that $\mathrm{OWA}^{\downarrow}(y) = \max_{\pi \in \Pi} \sum_i w_i y_{\pi(i)}$ where Π is the set of all possible permutations of $(1, \ldots, p)$. Combining convexity of $\mathrm{OWA}^{\downarrow}$ and equality $\mathrm{OWA}^{\downarrow}(\lambda y) = \lambda \mathrm{OWA}^{\downarrow}(y)$, one deduces thus the required inequality: $\mathrm{OWA}^{\downarrow}(y-y') + \mathrm{OWA}^{\downarrow}(y') = 2(\frac{1}{2}\mathrm{OWA}^{\downarrow}(y - y') + \frac{1}{2}\mathrm{OWA}^{\downarrow}(y')) \geq 2\mathrm{OWA}^{\downarrow}(\frac{1}{2}(y - y') + \frac{1}{2}y') = \mathrm{OWA}^{\downarrow}(y)$.

Proof of (i). Suppose there exists a cut C and an uncolored crossing edge e^k that satisfies the cut optimality condition. Let T be an $\mathrm{OWA}^{\downarrow}$-optimal spanning tree of $\mathcal{T}(c)$ that does not contain e^k. Now consider the graph formed by adding e^k to T. This graph has a cycle that contains e^k, and this cycle must contain at least one other uncolored crossing edge — say $e^{k'}$ — such that $e^{k'} \in C$, and therefore we have $\mathrm{OWA}^{\downarrow}(v^k - v^{k'}) \leq 0$. We can get a new spanning tree $T' \in \mathcal{T}(c)$ by deleting $e^{k'}$ from T and adding e^k. We claim that $\mathrm{OWA}^{\downarrow}(f(T')) \leq \mathrm{OWA}^{\downarrow}(f(T))$. Cost $f(T')$ is indeed equal to $f(T) - v^{k'} + v^k$. By the property indicated above, we have $\mathrm{OWA}^{\downarrow}(f(T) - v^{k'} + v^k) - \mathrm{OWA}^{\downarrow}(f(T)) \leq \mathrm{OWA}^{\downarrow}(f(T) - v^{k'} + v^k - f(T)) = \mathrm{OWA}^{\downarrow}(v^k - v^{k'})$. Consequently, since $\mathrm{OWA}^{\downarrow}(v^k - v^{k'}) \leq 0$, we deduce $\mathrm{OWA}^{\downarrow}(f(T')) = \mathrm{OWA}^{\downarrow}(f(T) - v^{k'} + v^k) \leq \mathrm{OWA}^{\downarrow}(f(T))$, and T' is therefore an $\mathrm{OWA}^{\downarrow}$-optimal spanning tree in $\mathcal{T}(c)$ (that does contain edge e^k).

Proof of (ii). Suppose there exists a cycle C containing no red edge with an uncolored edge $e^k \in C$ such that $\mathrm{OWA}^{\downarrow}(v^{k'} - v^k) \leq 0$ for all uncolored edges $e^{k'} \in C$. Let T be an $\mathrm{OWA}^{\downarrow}$-optimal spanning tree of $\mathcal{T}(c)$ that contains e^k. Now consider the graph formed by removing e^k from T. This graph is compounded of two connected components. The induced cut contains at least one other uncolored crossing edge — say $e^{k'}$ — such that $e^{k'} \in C$, and therefore $\mathrm{OWA}^{\downarrow}(v^{k'} - v^k) \leq 0$. We can get a new spanning tree $T' \in \mathcal{T}(c)$ by deleting e^k from T and adding $e^{k'}$. We claim that $\mathrm{OWA}^{\downarrow}(f(T')) \leq \mathrm{OWA}^{\downarrow}(f(T))$. Cost $f(T')$ is indeed equal to $f(T) - v^k + v^{k'}$. By the property indicated above, we have $\mathrm{OWA}^{\downarrow}(f(T) - v^k + v^{k'}) - \mathrm{OWA}^{\downarrow}(f(T)) \leq \mathrm{OWA}^{\downarrow}(v^{k'} - v^k)$. Consequently, since $\mathrm{OWA}^{\downarrow}(v^{k'} - v^k) \leq 0$, we deduce $\mathrm{OWA}^{\downarrow}(f(T')) \leq \mathrm{OWA}^{\downarrow}(f(T))$, and T' is therefore an $\mathrm{OWA}^{\downarrow}$-optimal spanning tree in $\mathcal{T}(c)$ (that does not contain edge e^k). □

Let us come back to the instance of Example 1 to illustrate how to color edges according to these conditions.

Example 2. *In the graph of Example 1, by considering cut $\{[1,4],[2,4],[3,4]\}$, edge $[1,4]$ can be colored blue since $OWA^{\downarrow}(v^{[1,4]} - v^{[2,4]}) = OWA^{\downarrow}(-1,-4,1) = -0.6$ (with $w_1 = 0.5$, $w_2 = 0.3$ and $w_3 = 0.2$) and $OWA^{\downarrow}(v^{[1,4]} - v^{[3,4]}) = OWA^{\downarrow}(0,-3,1) = -0.1$, where the valuation of edge e is denoted by v^e. Furthermore, by considering cycle $\{[2,4],[2,3],[3,4]\}$, edge $[2,4]$ can be colored red since $OWA^{\downarrow}(v^{[2,3]} - v^{[2,4]}) = OWA^{\downarrow}(0,-2,0) = -0.4$ and $OWA^{\downarrow}(v^{[3,4]} - v^{[2,4]}) = OWA^{\downarrow}(-1,-1,0) = -0.5$.*

Note that Pareto-dominance of edge $e^{k'}$ over edge e^k implies $\mathrm{OWA}^{\downarrow}(v^{k'} - v^k) \leq 0$ since all components of $v^{k'} - v^k$ are negative in such a case (see edge $[2,4]$ in the previous example). Our optimality conditions are thus an enrichment of the multiobjective optimality conditions provided by Sourd and Spanjaard [43], which are sufficient conditions for an edge to be mandatory or forbidden when generating the whole Pareto set. These conditions are indeed exclusively based on Pareto dominance between edges. We take here advantage of the knowledge of the $\mathrm{OWA}^{\downarrow}$ operator to optimize so as to be able to color a higher number of edges.

The single objective versions of these conditions make it possible to design a generic greedy method [e.g., 45], from which Kruskal's and Prim's algorithms can be derived. For problem $\mathrm{OWA}^{\downarrow}$-ST, this method can be adapted so

Algorithm PreprocessingByGoc(G,c)
Input : A graph G with coloring c of edges
Output : A maximal coloring c
for each edge e of E **do**
  if the cut optimality condition holds for e **then**
    set $c(e)$ = `blue`
  else if the cycle optimality condition holds for e **then**
    set $c(e)$ = `red`
return c

Figure 1: Preprocessing by applying generalized optimality conditions (*goc*) in the case of decreasing weights.

as to preprocess the graph prior to the resolution of the MIP formulation by a solver. The corresponding algorithm is indicated in Figure 1. Obviously, and contrarily to the single objective case, this preprocessing method does not yield a spanning tree since binary relation $\text{OWA}^{\downarrow}(v^k - v^{k'}) \leq 0$ does not induce a complete ordering of the edges in E. It enables however an important reduction of the number of variables (see numerical experiments), and therefore of the resolution time.

The complexity of the preprocessing method strongly depends on the complexity of detecting uncolored edges satisfying an optimality condition. In practice, to determine whether an uncolored edge $e^k = [i, j]$ satisfies the cut optimality condition, one performs a depth first search from i in the partial graph $G_k^{\text{cut}} = (V, E_k^{\text{cut}})$ where $E_k^{\text{cut}} = \{e^{k'} \in E \,|\, \text{OWA}^{\downarrow}(v^k - v^{k'}) > 0\} \cup \{e^{k'} \in E \,|\, c(e^{k'}) = \texttt{blue}\}$. If j does not belong to the set of visited vertices, then the partition between visited and non-visited vertices constitutes a cut for which e^k satisfies the cut optimality condition. Similarly, to determine whether an uncolored edge $e^k = [i, j]$ satisfies the cycle optimality condition, one performs a depth first search from i in the partial graph $G_k^{\text{cyc}} = (V, E_k^{\text{cyc}})$ where $E_k^{\text{cyc}} = \{e^{k'} \in E \,|\, \text{OWA}^{\downarrow}(v^{k'} - v^k) \leq 0\} \setminus \{e^k\} \cup \{e^{k'} \in E \,|\, c(e^{k'}) = \texttt{blue}\}$. If j is visited, then the chain from i to j in the search tree, completed with $[i, j]$, constitutes a cycle for which e^k satisfies the cycle optimality condition. Since the number of edges in the graph is m and the complexity of a depth first search is within $O(m)$ in a connected graph, the complexity of the overall preprocessing is $O(m^2)$.

4.2. Shaving procedures

The term "shaving" was introduced by Martin and Shmoys [25] for the job-shop scheduling problem. It aims at reducing the size of the instance before running the main algorithm. The shaving procedure can be specified as follows for OWA-ST. Assuming at least a feasible solution is known, for each edge e, we build a subproblem in which e is made mandatory. If the computation of a lower bound proves that the subproblem cannot improve the current best known solution, then it means that e can be made definitively forbidden (colored red). Conversely when e has not been colored red by the previous procedure, we test similarly whether e can be made definitively mandatory (colored blue). Note that, when computing the lower bounds, one checks whether a newly detected spanning tree improves the current best known solution (according to OWA). Of course, the shaving procedure is all the more efficient as a good feasible solution is initially known. For this purpose, we generate k feasible solutions by running a k-best ranking algorithm (i.e., returning the k best solutions) for the minimum spanning tree problem on the instance valued by the arithmetic mean of the vectors. The choice of k depends on the size of the instance.

Example 3. *Let us come back to the graph of Example 1. Assuming that the k-best ranking algorithm is run for* $k = 2$*, the current best known solution is then spanning tree* $T_4 = \{[1, 2], [1, 4], [2, 3]\}$*. Let us now simulate the progress of the shaving procedure. Making edge* $[1, 2]$ *forbidden, assume that the optimal OWA-value is lower bounded by 7.4. Since it is strictly greater than the OWA-value of* T_4 *(*$\text{OWA}(f(T_4)) = 7$*), edge* $[1, 2]$ *is colored blue. Then, making edge* $[1, 3]$ *mandatory, the optimal OWA-value is lower bounded by 7.3. Since it is strictly greater than the OWA-value of* T_4 *(*$\text{OWA}(f(T_4)) = 7$*), edge* $[1, 3]$ *is colored red. The procedure continues until all the edges have been examined.*

For such procedure, the efficient computation of bounds on the optimal value of problem OWA-ST is therefore worth investigating. To obtain such bounds, we present here two alternative relaxation methods for a multiobjective compromise search problem P: $\min \varphi(y)$ s.t. $y = f(x)$, $x \in X$, as defined in Section 2.1.

- *Relaxation of the image set:* it consists in defining a superset $Y' \supseteq Y$ of the image set (we recall that $Y = f(X)$) and solving problem $P_{Y'}$ defined by:

$$(P_{Y'}) \quad \begin{cases} \min & \varphi(y) \\ \text{s.t.} & y \in Y' \end{cases}$$

- *Relaxation of the objective function:* it consists in defining a function $\varphi' : \mathbb{R}^p \to \mathbb{R}$ such that for all images y in Y, $\varphi'(y) \leq \varphi(y)$ and solving problem $P_{\varphi'}$ defined by:

$$(P_{\varphi'}) \quad \begin{cases} \min & \varphi'(y) \\ \text{s.t.} & y = f(x) \\ & x \in X \end{cases}$$

The main point is then to define Y' (resp. φ') such that the optimal value of problem $P_{Y'}$ (resp. $P_{\varphi'}$) can be efficiently computed and provide a tight bound. In this concern, it seems opportune to consider some continuous relaxations of Y (resp. linear aggregation function φ'). In the following, we show more precisely how these relaxations can be performed when the aggregation function is an ordered weighted average. Note that, in both relaxations presented in the following, the computational efficiency of the procedure (to solve $P_{Y'}$ or $P_{\varphi'}$) only depends on the ability to quickly solve the single objective version of the problem. Consequently, these procedures can be applied to a broad spectrum of problems for which the single objective version can be efficiently solved.

4.2.1. Defining Y' and solving problem $P_{Y'}$

We now present a first bound on the value of an optimal solution to P, obtained by relaxation of the image set. For this purpose, let us define relaxed space Y' in program $P_{Y'}$ as follows:

$$Y' = \{y \in \mathbb{R}^p : y_i \geq b_i \quad \forall i = 0, \ldots, p\}$$

where $b_i = \min\{f_i(x) : x \in X\}$ denotes the value of an optimal solution according to f_i, and $f_0 = \sum_{i=1}^{p} f_i$. The values b_i's ($i = 1, \ldots, p$) are obtained by n runs of a standard algorithm for the single objective version of the problem (provided it can be efficiently solved) for valuations f_i's successively ($i = 1, \ldots, p$). Concerning the computation of b_0, for simplicity purpose, we only detail the case of multiobjective spanning tree problems. Value b_0 is obtained by applying a standard minimal spanning tree algorithm on the graph where each edge e^k is valued by $v_0^k = \sum_{i=1}^{p} v_i^k$: for any spanning tree x, we have indeed $f_0(x) = \sum_{i=1}^{p} f_i(x) = \sum_{i=1}^{p}(\sum_{k=1}^{m} v_i^k x^k) = \sum_{k=1}^{m}(\sum_{i=1}^{p} v_i^k)x^k = \sum_{k=1}^{m} v_0^k x^k$. Note that this technique to compute b_0 extends to many multiobjective versions of classical combinatorial optimization problems (actually, as soon as the value of a feasible solution is additively decomposable over its elements).

As indicated above, in order for the relaxation to be interesting in practice, one needs to provide an efficient procedure to compute the optimal value of program $P_{Y'}$. When the objective function is OWA, one clearly needs to take into account the way the components of solution y are ordered for solving $P_{Y'}$. The OWA function is actually piecewise linear. More precisely, it is linear within each subspace of $\mathbb{R}^p$ where all vectors are comonotonic, i.e. for any pair y, y' of vectors, there exists a permutation π of $(1, \ldots, p)$ such that $y_{\pi(1)} \geq \ldots \geq y_{\pi(p)}$ and $y'_{\pi(1)} \geq \ldots \geq y'_{\pi(p)}$. Problem $P_{Y'}$ can thus be divided into several subproblems, each one focusing on a particular comonotonic subspace of $\mathbb{R}^p$. The solution of $P_{Y'}$ reduces to solving each linear program $P_{Y',\pi}$ defined by a particular permutation π of $(1, \ldots, p)$:

$$(P_{Y',\pi}) \begin{cases} \min \displaystyle\sum_{i=1}^{p} w_i y_{\pi(i)} & \\ \text{s.t. } y_{\pi(i)} \geq y_{\pi(i+1)} \quad \forall i \in \{1, \ldots, p-1\}, & (3a) \\ \quad y_i \geq b_i \quad \forall i \in \{1, \ldots, p\}, & (3b) \\ \quad \displaystyle\sum_{i=1}^{p} y_i \geq b_0, & (3c) \\ \quad y \in \mathbb{R}^p. & \end{cases}$$

The value of the optimal solution $y^*_{Y'}$ to $P_{Y'}$ is then $\min_{\pi \in \Pi} \text{OWA}(y^*_{Y',\pi})$ where $y^*_{Y',\pi}$ denotes the optimal solution to linear program $P_{Y',\pi}$ and Π the set of all possible permutations. Note that for p components, there are $|\Pi| = p!$ linear

programs to solve. However, in practice, it is not necessary to solve the $p!$ linear programs. There exists indeed an easily computable permutation π^* for which $\text{OWA}(y^*_{Y'}) = \text{OWA}(y^*_{Y',\pi^*})$:

Proposition 2 (Galand and Spanjaard, 2007 [14])**.** *Let π^* denote the permutation such that $b_{\pi^*(1)} \geq b_{\pi^*(2)} \geq \ldots \geq b_{\pi^*(p)}$. For any feasible solution y to $P_{Y',\pi}$, there exists a feasible solution y' to P_{Y',π^*} such that $OWA(y') = OWA(y)$.*

Proof. The idea is to determine a feasible solution y' to P_{Y',π^*} such that $y'_{(i)}=y_{(i)}\ \forall i$ (where $y_{(1)} \geq \ldots \geq y_{(p)}$). It implies indeed $\text{OWA}(y) = \text{OWA}(y')$ and the conclusion is then straightforward. In this respect, we construct a sequence $(y^j)_{j=1,\ldots,k}$ of solutions and a sequence $(\pi^j)_{j=1,\ldots,k}$ of permutations such that y^j is feasible for P_{Y',π^j} (for $j=1,\ldots,k$), with $y^1=y$, $\pi^1=\pi$, $\pi^k=\pi^*$ and $y^1_{(i)}=y^2_{(i)}=\ldots=y^k_{(i)}\ \forall i$. Assume there exist $i_0, i_1 \in \{1,\ldots,p\}$ such that $i_0 < i_1$ and $b_{\pi^1(i_0)} < b_{\pi^1(i_1)}$. Let permutation π^2 be defined by $\pi^2(i_0)=\pi^1(i_1)$, $\pi^2(i_1)=\pi^1(i_0)$, and $\pi^2(i)=\pi^1(i)\ \forall i \neq i_0, i_1$. Let solution y^2 be defined by $y^2_{\pi^2(i)}=y^1_{\pi^1(i)}$ for $i=1,\ldots,p$. We now show that y^2 is a feasible solution to P_{Y',π^2}. Note first that $y^1_{\pi^1(i_0)} \geq b_{\pi^1(i_0)}$, $y^1_{\pi^1(i_1)} \geq b_{\pi^1(i_1)}$, $y^1_{\pi^1(i_0)} \geq y^1_{\pi^1(i_1)}$ and $b_{\pi^1(i_1)} > b_{\pi^1(i_0)}$. Hence, constraints 3b are satisfied since:
- $y^2_{\pi^2(i_0)}=y^1_{\pi^1(i_0)} \geq y^1_{\pi^1(i_1)} \geq b_{\pi^1(i_1)}=b_{\pi^2(i_0)}$,
- $y^2_{\pi^2(i_1)}=y^1_{\pi^1(i_1)} \geq b_{\pi^1(i_1)} > b_{\pi^1(i_0)}=b_{\pi^2(i_1)}$,
- $y^2_{\pi^2(i)}=y^1_{\pi^1(i)} \geq b_{\pi^1(i)}= b_{\pi^2(i)}$ for $i \neq i_0, i_1$.

Constraints 3a are also satisfied since $[y^1_{\pi^1(i)} \geq y^1_{\pi^1(i+1)}\ \ \forall i] \Rightarrow [y^2_{\pi^2(i)} \geq y^2_{\pi^2(i+1)}\ \forall i]$. Indeed, we have $y^1_{\pi^1(i)}=y^2_{\pi^2(i)}\ \forall i$. These equalities imply also that constraint 3c is satisfied and that $y^2_{(i)}= y_{(i)}\ \forall i$. Solution y^2 is therefore feasible for P_{Y',π^2} with $y^2_{(i)}=y_{(i)}\ \forall i$. Since any permutation is the product of elementary permutations, one always can construct in this way a sequence of permutations that leads to π^* (and the corresponding feasible solutions). By setting $y'=y^k$, one obtains the desired feasible solution to P_{Y',π^*}. □

An immediate consequence of this result is that $\text{OWA}(y^*_{Y'}) = \text{OWA}(y^*_{Y',\pi^*})$. Thus, the computation of $\text{OWA}(y^*_{Y'})$ reduces to solving linear program P_{Y',π^*}. For the sake of illustration, we present below an example.

Example 4. *Let us come back to Example 1 where bounds b_i's are defined from spanning trees T_1, T_2 and T_3 by $b_1 = 4$, $b_2 = 7$, $b_3 = 3$. The optimal spanning tree with respect to f_0 is spanning tree $T = \{[1,4],[2,3],[3,4]\}$ with $f_0(T) = 19$. The value of b_0 is thus 19. It yields the following program $P_{Y'}$ (the values of w_1, w_2 and w_3 do not matter):*

$$\begin{cases} \min & OWA(y) = w_1 y_{(1)} + w_2 y_{(2)} + w_3 y_{(3)} \\ s.t. & y_1 \geq 4 \quad y_2 \geq 7 \quad y_3 \geq 3, \\ & y_1 + y_2 + y_3 \geq 19, \\ & y_1 \in \mathbb{R},\ y_2 \in \mathbb{R},\ y_3 \in \mathbb{R}. \end{cases}$$

As a consequence of Proposition 2, solving $P_{Y'}$ amounts to solving linear program P_{Y',π^*} defined by:

$$\begin{cases} \min & w_1 y_2 + w_2 y_1 + w_3 y_3 \\ s.t. & y_2 \geq y_1 \geq y_3, \\ & y_1 \geq 4 \quad y_2 \geq 7 \quad y_3 \geq 3, \\ & y_1 + y_2 + y_3 \geq 19, \\ & y_1 \in \mathbb{R},\ y_2 \in \mathbb{R},\ y_3 \in \mathbb{R}. \end{cases}$$

We have indeed $\pi^*(1) = 2$, $\pi^*(2) = 1$ and $\pi^*(3) = 3$ since $b_2 \geq b_1 \geq b_3$. If $w_1 = 0.5$, $w_2 = 0.3$ and $w_3 = 0.2$, the value of the bound provided by programm P_{Y',π^*} is then 6.5 (we recall that the value of the OWA-optimal spanning tree is 7).

4.3. Defining φ' and solving problem $P_{\varphi'}$

The second bound for P is obtained by relaxation of the objective function. The idea is to use a linear function φ' defined by $\varphi'(y) = \sum_{i=1}^{p} \lambda_i y_i$ where the vector $\lambda = (\lambda_1, \ldots, \lambda_p)$ of coefficients satisfy the following constraints:

$$\sum_{i \in I} \lambda_i \leq \sum_{i=1}^{|I|} w_i \quad \text{for all subset } I \subseteq \{1,\ldots,p\} \text{ of objectives} \tag{4}$$

When using such weight vectors, we have indeed $\varphi'(y) \leq \varphi(y)$ for all y in Y, as established by the following:

Proposition 3. *Let Λ denote the set of all vectors $\lambda \in \mathbb{R}^p$ satisfying Constraint 4. If $\lambda \in \Lambda$, then for all $y \in \mathbb{R}^p$, $\sum_{i=1}^{p} \lambda_i y_i \leq OWA(y)$.*

Proof. Let π denote a permutation such that $y_{\pi(1)} \geq \ldots \geq y_{\pi(p)}$. Note first that $\text{OWA}(y) = \sum_{i=1}^{p} w_i y_{\pi(i)} = \sum_{i=1}^{p}(\sum_{j=1}^{i} w_j)(y_{\pi(i)} - y_{\pi(i+1)})$ where $y_{\pi(p+1)} = 0$. By Constraint 4 with subset of objectives $I = \{\pi(1), \ldots, \pi(i)\}$, we have $\sum_{j=1}^{i} w_j \geq \sum_{j=1}^{i} \lambda_{\pi(j)}\ \forall i$. Thus $\sum_{i=1}^{p}(\sum_{j=1}^{i} w_j)(y_{\pi(i)} - y_{\pi(i+1)}) \geq \sum_{i=1}^{p}(\sum_{j=1}^{i} \lambda_{\pi(j)})(y_{\pi(i)} - y_{\pi(i+1)})$. Since $\sum_{i=1}^{p}(\sum_{j=1}^{i} \lambda_{\pi(j)})(y_{\pi(i)} - y_{\pi(i+1)}) = \sum_{i=1}^{p} \lambda_{\pi(i)} y_{\pi(i)}$ and $\sum_{i=1}^{p} \lambda_{\pi(i)} y_{\pi(i)} = \sum_{i=1}^{p} \lambda_i y_i$, we have $\text{OWA}(y) \geq \sum_{i=1}^{p} \lambda_i y_i$. □

This result can be seen as a specific instanciation of a more general result of Schmeidler [40, 41] on Choquet integrals in decision under uncertainty (OWA being a particular subclass of Choquet integrals). Schmeidler's result has been used to provide bounds in Choquet-based optimization under uncertainty [12], as well as under multiple objectives [13]. Note that, in the case of decreasing weights in OWA, a useful by-product of Schmeidler's result is the existence of a normalized set of weights (i.e., summing up to 1) such that Proposition 3 holds: for instance, when setting $\lambda_i = 1/p$, we have $\sum_i (1/p) y_i \leq \text{OWA}(y)$ by Chebyshev's sum inequality. Nevertheless, when the weights are not decreasing, the existence of normalized weights satisfying Constraint 4 is not guaranteed.

Solving problem $P_{\varphi'}$ can be efficiently done by valuing every edge e^k by $\sum_{i=1}^{p} \lambda_i v_i^k$ and then performing a standard minimum spanning tree algorithm. Besides, in order to obtain the best lower bound as possible, the optimal set of weights $\lambda \in \Lambda$ (i.e., providing the best lower bound according to Proposition 3) can be obtained by solving the following program:

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^p} \quad & z(\lambda) = \min_{y \in Y} \sum_{i=1}^{p} \lambda_i y_i, \\ & \text{s.t.} \quad \sum_{i \in I} \lambda_i \leq \sum_{i=1}^{|I|} w_i \quad \forall I \subseteq \{1, \ldots, p\}, \\ & \qquad \lambda_i \geq 0 \quad \forall i = 1, \ldots, p. \end{aligned}$$

Given that z is a concave piecewise linear function of λ for a fixed y (since it is the lower envelope of a set of linear functions $\{\sum_{i=1}^{p} y_i \lambda_i : y \in Y\}$), we solve this program by using the SolvOpt library [19], which is an implementation of Shor's r-algorithm [42]. This algorithm is indeed especially convenient for non-differentiable optimization, and the implemented SolvOpt library enables to perform constrained optimization. This approach is closed to the lower bounding procedure proposed by Punnen and Aneja [37] for min-max combinatorial optimization. In broad outline, it can be viewed as a sequence of minimum spanning tree computations according to a varying weight vector λ, until convergence towards a weight vector maximizing $\min_{y \in Y} \sum_{i=1}^{p} \lambda_i y_i$.

Example 5. *Let us come back again to the instance of Example 1. Running Shor's r-algorithm yields $\lambda = (0.28, 0.5, 0.22)$ and $y = (6, 8, 6)$ (the image of spanning tree $\{[1, 2], [1, 4], [2, 3]\}$, which is actually OWA-optimal). The bound is therefore $0.28 \times 6 + 0.5 \times 8 + 0.22 \times 6 = 7$ (which is the value of an OWA-optimal tree). This is better than the bound obtained by the previous relaxation (value 6.5). The computational burden is however more important, since it requires to solve much more single objective problems.*

5. Experimental results

Before we study more carefully the behavior of our algorithms, we give some insights into previous results on related topics. The most widely studied related topic is the generation of the Pareto set in the bi-objective spanning tree problem [4, 16, 38, 43, 44]. To our knowledge, the most efficient algorithm for this problem enables to solve randomly drawn instances on complete graphs containing up to 400 vertices [43] (note that these results significantly improved the size of the instances that could be handled, since it grew from 40 to 400 vertices). However these results do not extend to more than two objectives. More generally, even when looking for a single compromise solution within the Pareto set, it seems that there is very few available numerical experiments in the literature for more than two objectives. Although several works deal with the min-max spanning tree problem (i.e. determining a spanning tree minimizing the max criterion) [2, 3, 16, 47, 52], the content of these works is indeed mainly theoretical. Actually, the

only numerical results we know for more than two objectives are devoted to the determination of a Choquet-optimal spanning tree [13]. The size of the tackled instances goes from 30 to 70 vertices according to the number of objectives and the parameterization of the Choquet integral.

5.1. Experimental details

All the algorithms have been implemented in C++ and were run on either an Intel Xeon 2.5 GHz personal computer with a memory of 4GB for 3 and 5 objectives, or an Intel Core 3.0 GHz personal computer with a memory of 8GB for 8 objectives. The test instances are defined as follows. All considered graphs are complete. On each edge, the components of cost vectors are randomly drawn according to an uniform distribution on $[1, 100]$. For each kind of instances (depending on the number of nodes and on the number of objectives), 30 instances were randomly drawn to obtain average results.

The global procedure to solve problem P_{OWA} consists of two phases:

1. *Preprocessing phase*: making the most possible edges blue (mandatory) or red (forbidden) by applying first the generalized optimality conditions, and then a shaving procedure taking into account the current coloration.
2. *Resolution phase*: determining the OWA-optimal spanning tree by running solver IBM ILOG CPLEX 12 on the reduced MIP.

For initializing the shaving procedure in the coloration phase, the k-best ranking algorithm proposed by Katoh et al. [20] is launched for k varying from 500 to 5000 depending on the size of the instance. In all cases, the order of magnitude of its running time is about a few milliseconds. Concerning the shaving itself, the procedure by relaxation of the objective function is denoted by $sh_{\varphi'}$, while the one by relaxation of the image set is denoted by $sh_{Y'}$. Note that every edge is examined only once in the shaving procedure, since preliminary numerical tests have shown that there is no significant gain in repeating the process to take into account new colored edges.

This section is compounded of two subsections: the first one is dedicated to the case of decreasing weights, and the second one to the case of arbitrary weights. Whatever the weights, one compares three resolution procedures:

- solution by CPLEX without any preprocessing of the initial instance, denoted by P (using formulation P_{OWA} or $P_{\text{OWA}^{\downarrow}}$ depending on the weights);
- solution by CPLEX on the instance preprocessed by using generalized optimality conditions (provided the weights are decreasing) and shaving $sh_{\varphi'}$, denoted by $\text{P}_{\varphi'}$;
- solution by CPLEX on the instance preprocessed by using generalized optimality conditions (provided the weights are decreasing) and shaving $sh_{Y'}$, denoted by $\text{P}_{Y'}$.

5.2. Tests with decreasing weights

In this section, one summarizes the results one has obtained for two classes of $\text{OWA}^{\downarrow}$, named hereafter *$\text{OWA}^{\downarrow}$ with exponential weights* and *k-centra*. Criterion $\text{OWA}^{\downarrow}$ with exponential weights is defined by setting the weights as follows: $w_i = (\frac{i}{p})^{\beta} - (\frac{i-1}{p})^{\beta}$ where $\beta \geq 1$. For $\beta = 1$, the subsequent $\text{OWA}^{\downarrow}$ function reduces to the arithmetic mean, while it converges towards max when β increases. The k-centra criterion is defined by setting the weights as follows: $w_1 = \ldots = w_k = \frac{1}{k}$ and $w_{k+1} = \ldots = w_p = 0$ for $k \in \{1, \ldots, p\}$. For $k = p$, here again, the subsequent $\text{OWA}^{\downarrow}$ function reduces to the arithmetic mean, while it converges towards max when k decreases.

Tables 1, 2 and 3 summarize the results obtained for $\text{OWA}^{\downarrow}$ with exponential weights for $\beta = 1.5$, 2 and 3. The algorithms have been launched on complete graphs with 3, 5 or 8 objectives, for various numbers of vertices for which the average resolution time is lower than 15 minutes. Symbol "-" indicates that the average execution times is beyond 15 minutes. The upper parts of the tables summarize the informations about the preprocessing method (line *goc* for the generalized optimality conditions and line $sh_{\varphi'}$ or $sh_{Y'}$ for the shaving procedures). For each procedure and each size of instance, the average execution time is indicated. Furthermore, below line *goc* (resp. $sh_{\varphi'}$ or $sh_{Y'}$), the average number of edges made blue ($\#b$) or red ($\#r$) after applying the generalized optimality conditions (resp. after applying the generalized optimality conditions *and* shaving) is indicated (couple $(\#b - \#r)$). In the lower part of the tables are indicated the average total resolution times. To evaluate the variability of the resolution time, the minimal running time (min) as well as the maximal one (max) are also indicated (couple min – max under the average execution time).

	n	20	30	40	60	80	100
$\beta = 1.5$	goc	0.01	0.02	0.05	0.21	0.55	1.18
		(7.7 - 151.3)	(11.3 - 370.2)	(15.3 - 695)	(23 - 1637.23)	(30.2 - 2982.23)	(38.57 - 4730.53)
	$sh_{\varphi'}$	0.27	1.63	5.55	40.13	158.54	461.84
		(14.97 - 165.03)	(21.43 - 396.17)	(30.93 - 730.47)	(45.43 - 1692.7)	(60.2 - 3058.83)	(74.23 - 4824.07)
	$sh_{Y'}$	0	0.01	0.02	0.05	0.10	-
		(10.57 - 158.67)	(14.13 - 382.63)	(21.4 - 712.97)	(29.57 - 1661.2)	(41.37 - 3016.3)	
	P	0.32	1.38	20.60	78.06	725.88	-
		0.03 - 1.18	0.14 - 12.36	0.24 - 195.06	4.52 - 438.38	14.68 - 4413.27	
	$P_{\varphi'}$	0.28	1.68	5.73	40.69	163.40	469.45
		0.15 - 0.48	1.09 - 2.24	4.73 - 9.72	24.21 - 62.01	122.49 - 230.01	364.25 - 606.18
	$P_{Y'}$	0.07	0.19	3.22	6.19	44.42	-
		0.01 - 0.73	0.04 - 0.80	0.11 - 48.58	0.28 - 71.84	0.71- 496.61	
$\beta = 2$	goc	0.01	0.02	0.05	0.20	0.56	-
		(3.64 - 133.68)	(5.52 - 341.42)	(6.78 - 642.24)	(9.46 - 1554.83)	(13.4 - 2868.43)	
	$sh_{\varphi'}$	0.48	2.72	10.38	72.43	304.18	-
		(12.89 - 161.3)	(20.02 - 392.12)	(26.86 - 723.48)	(40.33 - 1684.81)	(56 - 3051.07)	
	$sh_{Y'}$	0.01	0.02	0.04	0.10	-	-
		(6.95 - 148.47)	(11.08 - 371.24)	(14.38 - 685.26)	(22.68 - 1620.56)		
	P	0.47	3.64	20.24	156.23	-	-
		0.04 - 4.17	0.08 - 111.26	0.31 - 166.89	6.51 - 1613.53		
	$P_{\varphi'}$	0.53	3.06	11.17	75.86	331.24	-
		0.36 - 1.57	2.19 - 14.12	8.30 - 27.73	64.85 - 125.14	260.66 - 831.34	
	$P_{Y'}$	0.17	1.62	5.98	23.45	-	-
		0.03 - 2.27	0.09 - 50.48	0.24 - 64.86	0.20 - 243.15		
$\beta = 3$	goc	0	0.02	0.05	0.21	0.59	-
		(1.1 - 108.72)	(1.66 - 297.74)	(1.84 - 573.52)	(2.5 - 1420.3)	(3.5 - 2666.37)	
	$sh_{\varphi'}$	0.76	4.64	19.55	146.04	619.47	-
		(10.84 - 157.18)	(18.34 - 388.24)	(24.42 - 717.52)	(38.07 - 1674.87)	(52.27 - 3027.57)	
	$sh_{Y'}$	0.01	0.03	0.06	-	-	-
		(5.92 - 142.02)	(10.96 - 364.8)	(12.82 - 670.92)			
	P	1.18	7.29	18.74	-	-	-
		0.03 - 32.61	0.10 - 143.58	0.28 - 159.61			
	$P_{\varphi'}$	0.81	4.92	21.45	160.04	792.7	-
		0.51 - 1.62	3.69 - 8.19	14.58 - 41.36	108.69 - 336.43	159.85 - 4310.91	
	$P_{Y'}$	0.39	2.14	6.00	-	-	-
		0.03 - 10.14	0.10 - 33.93	0.23 - 54.93			

Table 1: Synthesis of the numerical results for 3 objectives and exponential weights. Execution times are indicated in seconds.

	n	20	30	40	50	60
$\beta = 1.5$	goc	0.01	0.02	0.06	0.13	0.24
		(3.23 - 132.27)	(5.3 - 335.6)	(6.33 - 632.13)	(7.47 - 1027.9)	(8.73 - 1530.2)
	$sh_{\varphi'}$	0.83	4.72	18.75	56.18	131.17
		(12.43 - 161.4)	(17.63 - 389.2)	(21.03 - 712.13)	(26.47 - 1139.67)	(31.33 - 1665.7)
	$sh_{\gamma'}$	0.01	0.03	0.07	0.12	-
		(3.67 - 134.3)	(6.57 - 345.2)	(7.37 - 647.7)	(8.37 - 1037.2)	
	P	0.83	2.86	18.10	127.41	-
		0.06 - 4.92	0.29 - 13.76	1.17 - 116.95	2.93 - 610.01	
	$P_{\varphi'}$	0.91	5.20	22.77	68.87	154.16
		0.62 - 2.08	3.71 - 9.66	14.63 - 58.79	50.18 - 204.46	110.53 - 232.27
	$P_{\gamma'}$	0.48	1.86	11.88	58.55	-
		0.04 - 2.26	0.09 - 10.73	0.54 - 79.06	2.33 - 328.19	
$\beta = 2$	goc	0.01	0.02	0.06	0.14	0.26
		(1.2 - 96.47)	(1.13 - 260.33)	(1.6 - 517.53)	(2.1 - 862.37)	(2.23 - 1306.43)
	$sh_{\varphi'}$	1.38	8.61	34.77	97.05	248.61
		(7.33 - 151.27)	(10.63 - 369.8)	(13.87 - 684.37)	(17.63 - 1115.67)	(20.27 - 1623.27)
	$sh_{\gamma'}$	0.02	0.06	0.13	0.23	-
		(2.03 - 107.67)	(1.86 - 283.53)	(2.6 - 543.57)	(3.23 - 896.3)	
	P	1.21	20.76	203.71	906.97	-
		0.17 - 5.14	0.77 - 163.95	2.78 - 4038.47	7.10 - 12796	
	$P_{\varphi'}$	1.86	13.35	150.04	398.42	474.06
		1.10 - 3.27	7.49 - 67.12	29.00 - 2640.69	82.18 - 7290.92	219.79 - 2666.74
	$P_{\gamma'}$	1.01	11.29	231.39	884.61	-
		0.17 - 4.72	0.12 - 106.11	1.20 - 5007.4	4.39 - 17244.5	
$\beta = 3$	goc	0.01	0.02	0.06	0.14	0.28
		(0.13 - 59.87)	(0.07 - 177.7)	(0.1 - 358.3)	(0.07 - 630.97)	(0.17 - 988.27)
	$sh_{\varphi'}$	1.91	12.40	52.33	162.58	429.13
		(6.47 - 142.13)	(7.23 - 349.03)	(8.57 - 652.4)	(10.2 - 1067.93)	(15.5 - 1578.1)
	$sh_{\gamma'}$	0.03	0.09	0.20	-	-
		(0.9 - 88.4)	(1.03 - 211)	(0.73 - 426.47)		
	P	2.68	79.78	225.84	-	-
		0.08 - 26.07	1.79 - 1693.07	4.46 - 2931.52		
	$P_{\varphi'}$	2.80	46.66	176.22	592.35	1552.49
		1.60 - 6.69	10.93 - 623.80	48.83 - 1897.01	144.43 - 6532.91	357.75 - 26292.2
	$P_{\gamma'}$	2.32	108.57	205.24	-	-
		0.07 - 25.62	1.35 - 2485.93	4.77 - 2281.07		

Table 2: Synthesis of the numerical results for 5 objectives and exponential weights. Execution times are indicated in seconds.

	n	20	30	40	50
$\beta = 1.5$	goc	0	0.01	0.02	0.05
		(1.73 - 107.23)	(2.03 - 282.33)	(2.47 - 555.67)	(2.37 - 913.3)
	$sh_{\varphi'}$	0.86	2.50	7.30	24.90
		(12.23 - 157.63)	(13.2 - 375.4)	(15.27 - 692.7)	(17.87 - 1104.93)
	$sh_{\gamma'}$	0	0.02	0.04	0.09
		(1.77 - 107.9)	(2.03 - 282.33)	(2.53 - 560.3)	(2.37 - 913.3)
	P	4.14	13.26	217.66	1459.31
		0.27 - 84.59	0.73 - 123.83	0.87 - 4512.49	13.18 - 28451.6
	$P_{\varphi'}$	1.41	5.60	33.95	459.65
		0.64 - 8.9	2.08 - 31.4	6.26 - 189.59	22.48 - 8809.44
	$P_{\gamma'}$	2.01	11.19	109.46	986.29
		0.07 - 35.94	0.67 - 103.11	0.92 - 1517	11.09 - 15509.1
$\beta = 2$	goc	0	0.01	0.02	-
		(0.17 - 58.3)	(0.13 - 172.53)	(0.27 - 353.4)	
	$sh_{\varphi'}$	1.10	4.27	14.05	-
		(4.33 - 136.5)	(4.1 - 330.77)	(5.6 - 621.97)	
	$sh_{\gamma'}$	0.01	0.03	0.07	-
		(0.17 - 60.9)	(0.27 - 178.1)	(0.4 - 358.4)	
	P	3.72	63.11	1839.15	-
		0.87 - 10.66	4.48 - 529.61	6.45 - 40479.9	
	$P_{\varphi'}$	2.90	37.36	908.85	-
		1.16 - 6.73	6.15 - 302.54	14.81 - 20912.8	
	$P_{\gamma'}$	3.28	66.68	1270.19	-
		0.63 - 11.12	5.27 - 621.6	7.27 - 26536.1	
$\beta = 3$	goc	0	0.01	-	-
		(0.03 - 22.9)	(0 - 71.8)		
	$sh_{\varphi'}$	1.47	6.03	-	-
		(1.27 - 108.43)	(1.37 - 273.93)		
	$sh_{\gamma'}$	0.01	0.05	-	-
		(0.03 - 26.9)	(0 - 81.9)		
	P	19.70	372.15	-	-
		1.93 - 159.87	10.22 - 1367.46		
	$P_{\varphi'}$	11.78	233.49	-	-
		1.72 - 77.35	11.21 - 978.84		
	$P_{\gamma'}$	17.09	419.93	-	-
		2.6 - 151.42	10.33 - 1941.46		

Table 3: Synthesis of the numerical results for 8 objectives and exponential weights. Execution times are indicated in seconds.

Tables 1, 2 and 3 show that the coloration phase has a significant impact on the resolution time of the MIP formulation. First and foremost, note that the number of blue edges has a more important impact on the size of the reduced instance than the number of red ones. The former number is indeed upper bounded by $n-1$ while the latter is upper bounded by $(n-2)(n-1)/2$. One observes that applying the generalized optimality conditions makes it possible to color a lot of edges in a negligible amount of time. Among the three coloration primitives (*goc*, $\text{sh}_{\varphi'}$ and $\text{sh}_{Y'}$), primitives *goc* and $\text{sh}_{Y'}$ are really interesting concerning the ratio number of colored edges over computation time spent. Nevertheless, primitive $\text{sh}_{\varphi'}$ becomes very useful when the size of the instance grows: the additionnal time spent during the coloration phase becomes indeed largely offset by the time saved during the resolution phase itself, since this primitive outperforms the two other ones in terms of detected blue edges. For instance, in Table 1, for $\beta = 3$ and $n = 80$, applying primitives *goc* and $\text{sh}_{\varphi'}$ turns blue 52 edges instead of 3 for primitive *goc* alone. Overall, the preprocessing procedures enable to save a large amount of time during the resolution phase. Primitive $\text{sh}_{Y'}$ should be preferred for medium-sized instances thanks to its low computationnal cost, while primitive $\text{sh}_{\varphi'}$ should be preferred for larger instances since it colors blue much more edges. As could be expected, the resolution time considerably increases with the number p of objectives and the number n of vertices. Besides, the resolution time exponentially grows with parameter β: for $\beta = 1.5$, the behavior of the $\text{OWA}^{\downarrow}$ criterion is close to the arithmetic mean, and therefore the resolution time is much lower than for $\beta = 2$ or 3.

Tables 4, 5 and 6 show the results obtained when using the k-centra criterion for various values of k (according to the number p of objectives). Clearly, the closer k is to p (resp. 1), the more the k-centra criterion behaves like the arithmetic mean (resp. the max criterion). The analysis of the numerical results yields the same observations as for the exponential weights: the preprocessing procedures enables to save a large amount of time, and the resolution time exponentially grows with the value of $p-k$.

	n	40	50	60	70	80	90
$k=1$	goc	0.05 (0.08 - 469.22)	0.08 (0.13 - 793)	0.22 (0.03 - 1203.17)	0.39 (0 - 1719.83)	0.62 (0.03 - 2320.97)	-
	$\text{sh}_{\varphi'}$	29.62 (21.04 - 709.54)	91.86 (29.92 - 1138.29)	245.41 (35.47 - 1661.8)	510.30 (41.13 - 2297.07)	959.67 (49.6 - 3024.9)	-
	$\text{sh}_{Y'}$	0.09 (9.72 - 638.26)	0.14 (17.13 - 1056.62)	0.26 (13.7 - 1440.17)	-	-	-
	P	66.09 1.26 - 1126.95	66.09 1.26 - 1126.95	-	-	-	-
	$\text{P}_{\varphi'}$	34.63 25.88 - 85.18	34.63 25.88 - 85.18	281.28 190.59 - 843.22	575.19 389.11 - 1403.64	1064.6 737.67 - 2175.44	-
	$\text{P}_{Y'}$	20.68 0.28 - 286.27	20.68 0.28 - 286.27	295.37 0.52 - 2831.34	-	-	-
$k=2$	goc	0.05 (2.72 - 626.12)	0.08 (3.47 - 1019.33)	0.20 (3.93 - 1514.67)	0.34 (3.77 - 2101.67)	0.55 (4.23 - 2792.8)	0.78 (5.1 - 3580.6)
	$\text{sh}_{\varphi'}$	14.81 (26.52 - 722.42)	45.25 (34 - 1153.4)	115.28 (40.3 - 1686.17)	244.80 (48.5 - 2318.23)	467.99 (58.67 - 3049.4)	833.78 (63.46 - 3876.97)
	$\text{sh}_{Y'}$	0.04 (15 - 693.24)	0.07 (18.67 - 1107.43)	0.12 (15.6 - 1598.17)	0.18 (18.47 - 2211.57)	0.24 (33.07 - 2964.67)	-
	P	14.43 0.39 - 122.35	18.96 1.75 - 116.80	-	-	-	-
	$\text{P}_{\varphi'}$	16.48 12.1 - 51.6	45.85 40.48 - 55.88	119.18 102.59 - 197.42	257.66 194.91 - 533.60	488.58 365.83 - 938.68	916.08 678.75 - 2860.15
	$\text{P}_{Y'}$	6.54 0.23 - 70.5	5.83 0.16 - 63.01	106.47 0.34 - 1422.78	234.99 0.53 - 1983.71	504.91 0.78 - 11416	-

Table 4: Synthesis of the numerical results for 3 objectives and k-centra criterion. Execution times are indicated in seconds.

	n	20	30	40	50	60
$k = 1$	goc	0.01 (0 - 21.53)	0.02 (0 - 72.43)	0.05 (0 - 167.5)	-	-
	$sh_{\varphi'}$	2.63 (2.3 - 117.83)	17.43 (4.03 - 316.17)	73.27 (5.03 - 601.9)	-	-
	$sh_{Y'}$	0.04 (0.33 - 51.67)	0.12 (0.6 - 150.1)	-	-	-
	P	11.45 0.69 - 111.26	90.79 0.92 - 339.21	-	-	-
	$P_{\varphi'}$	10.56 2.33 - 92.63	65.79 14.28 - 229.17	627.62 63.08 - 2637.86	-	-
	$P_{Y'}$	11.41 0.64 - 118.39	85.48 0.81 - 333.85	-	-	-
$k = 3$	goc	0.01 (0.43 - 82.43)	0.02 (0.3 - 226.63)	0.06 (0.4 - 452.9)	0.14 (0.33 - 764.83)	0.27 (0.4 - 1166.37)
	$sh_{\varphi'}$	1.90 (7.77 - 149.73)	12.76 (8.47 - 364.87)	51.61 (14.47 - 684.87)	148.01 (17.6 - 1105.83)	335.44 (17.87 - 1614.77)
	$sh_{Y'}$	0.02 (1.4 - 105.83)	0.07 (1.3 - 263.1)	0.15 (2.33 - 510.33)	0.29 (1.5 - 814.9)	-
	P	0.83 0.15 - 1.97	31.14 1.22 - 592.79	78.27 1.18 - 609.88	2039.07 8.49 - 56913.1	-
	$P_{\varphi'}$	2.26 1.42 - 3.48	24.26 10.32 - 224.53	86.82 41.33 - 442.88	420.05 118.68 - 7359.52	659.58 274.63 - 5825.48
	$P_{Y'}$	0.75 0.15 - 2.14	36.85 0.96 - 832.75	65.36 1.15 - 616.93	1896.52 6.11 - 54045.5	-
$k = 4$	goc	0.01 (2.7 - 129.37)	0.02 (3.2 - 325.67)	0.06 (4.57 - 623.93)	0.13 (4.67 - 1008.23)	0.23 (5.03 - 1493.67)
	$sh_{\varphi'}$	0.92 (12.2 - 161.13)	5.20 (18 - 388.4)	18.66 (24.43 - 717.07)	54.55 (28.4 - 1143.43)	134.86 (31.9 - 1667.03)
	$sh_{Y'}$	0.01 (3.37 - 134.67)	0.04 (4.4 - 333.83)	0.07 (6.2 - 640.17)	0.14 (7.17 - 1035.53)	0.21 (8.1 - 1529.9)
	P	0.44 0.05 - 2.13	11.57 0.27 - 184.38	12.90 0.43 - 225.17	242.63 0.90 - 4108.18	1115.72 8.92 - 11015.1
	$P_{\varphi'}$	0.96 0.67 - 1.31	5.60 4.31 - 9.94	19.43 14.69 - 29.53	60.31 45.77 - 103.19	393.96 115.43 - 3672.51
	$P_{Y'}$	0.34 0.04 - 2.22	4.13 0.15 - 64.42	13.75 0.16 - 273.18	58.11 0.52 - 366.36	842.55 3.70 - 9777.22

Table 5: Synthesis of the numerical results for 5 objectives and k-centra criterion. Execution times are indicated in seconds.

	n	20	30	40	50	60	70
$k = 2$	goc	0.001 (0 - 4.23)	0.00466667 (0 - 14.6)	-	-	-	-
	$sh_{\varphi'}$	1.8 (0.3 - 68.17)	7.36 (0.6 - 214.3)	-	-	-	-
	$sh_{Y'}$	0.02 (0 - 5.8)	0.05 (0 - 21.1)	-	-	-	-
	P	56.66 5.18 - 435.96	1278.29 36.9 - 6647.53	-	-	-	-
	$P_{\varphi'}$	50.15 3.9 - 393.43	1023.7 25.43 - 6000.2	-	-	-	-
	$P_{Y'}$	59.07 4.72 - 380.91	1269.18 22.29 - 5583.47	-	-	-	-
$k = 4$	goc	0 (0 - 24.6)	0.01 (0 - 80.07)	0.02 (0.03 - 169.47)	-	-	-
	$sh_{\varphi'}$	1.47 (1.17 - 105.33)	6.52 (2 - 283.5)	25.12 (1.93 - 539.4)	-	-	-
	$sh_{Y'}$	0.01 (0 - 30.2)	0.05 (0 - 88.07)	0.12 (0.1 - 184.03)	-	-	-
	P	23.42 1.68 - 134.88	295.01 31.04 - 3414.13	1776.19 95.06 - 11819.5	-	-	-
	$P_{\varphi'}$	16.61 1.78 - 81.39	188.76 18 - 1992.24	1040.29 66.8 - 5028.68	-	-	-
	$P_{Y'}$	20.65 2.32 - 112.13	274.61 14.28 - 3108.76	1697.1 75.52 - 10283	-	-	-
$k = 7$	goc	0 (2.9 - 134.03)	0.01 (5.3 - 336.77)	0.02 (5.5 - 635.63)	0.04 (6.17 - 1027.83)	0.08 (7.4 - 1529.7)	0.13 (8.83 - 2113.57)
	$sh_{\varphi'}$	0.64 (12.33 - 162.27)	1.86 (17.53 - 390.63)	5.11 (23.2 - 717)	15.44 (27.27 - 1143.6)	32.13 (33.23 - 1672.23)	57.11 (39.37 - 2297.2)
	$sh_{Y'}$	0 (2.93 - 134.3)	0.01 (5.43 - 337.43)	0.03 (5.63 - 637.2)	0.05 (6.57 - 1032.03)	0.08 (7.43 - 1530.03)	-
	P	1.42 0.27 - 7.87	3.02 0.6 - 25.63	121.89 0.91 - 3133.83	157.18 11.26 - 1532.2	480.38 11.46 - 7599.19	-
	$P_{\varphi'}$	0.88 0.52 - 2.51	2.58 1.51 - 8.36	17.96 4.39 - 320.91	34.65 14.72 - 158.46	73.99 26.6 - 526.27	341.13 41.68 - 3632.88
	$P_{Y'}$	0.85 0.14 - 3.06	2.26 0.33 - 22.98	77.66 0.46 - 1958.95	99.63 1.53 - 875.78	277.83 9.75 - 3608.48	-

Table 6: Synthesis of the numerical results for 8 objectives and k-centra criterion. Execution times are indicated in seconds.

5.3. Tests with arbitrary weights

We present here the results obtained when the weights are arbitrary. In this subsection, we study two particular cases of the OWA operator, namely the Hurwicz criterion and the k-trimmed criterion. The graph instances are defined similarly to the previous subsection, with 3 and 5 objectives. The notation conventions are the same as in the previous subsection. Note that the generalized optimality conditions are not valid anymore in this case, and are of course disabled in the numerical tests.

We recall that, for a vector y, the Hurwicz criterion is defined as $\alpha \max_i y_i + (1-\alpha)\min_i y_i$. In the numerical tests performed, parameter α varies from 0.4 to 0.8. Tables 7 and 8 summarize the results obtained.

	n	10	20	30	40	45	50
$\alpha = 0.4$	$sh_{\varphi'}$	0.13	4.36	39.4	-	-	-
		(0 - 0.02)	(0 - 0)	(0 - 0)			
	$sh_{Y'}$	0.01	0.04	0.12	0.29	0.42	0.57
		(0.16 - 12.9)	(0 - 1.06)	(0 - 0)	(0 - 0)	(0 - 0)	(0 - 0)
	P	0.3	3.82	25.16	221.99	296.19	613.27
		0.04 - 1.18	0.49 - 8.68	8.55 - 164.58	25.62 - 4526.81	32.31 - 3239.19	68.83 - 4407.86
	$P_{\varphi'}$	0.43	8.18	64.26	-	-	-
		0.16 - 1.35	3.82 - 14.59	39.21 - 203.13			
	$P_{Y'}$	0.16	3.94	24.96	219	282.75	616.22
		0.02 - 0.82	0.62 - 8.89	8.76 - 159.35	26.76 - 4433.47	31.96 - 2995.53	69.35 - 4488.94
$\alpha = 0.6$	$sh_{\varphi'}$	0.1	3.01	26.77	133.38	-	-
		(0.44 - 12.64)	(0 - 36.34)	(0 - 33.5)	(0 - 24.33)		
	$sh_{Y'}$	0	0.03	0.11	0.28	-	-
		(1.96 - 23.4)	(0.08 - 81.64)	(0 - 109.27)	(0 - 112.47)		
	P	0.2	2.16	16.32	188.25	-	-
		0.02 - 0.72	0.41 - 5.62	1.29 - 167.94	14.74 - 1287.06		
	$P_{\varphi'}$	0.24	4.98	40.97	332.09	-	-
		0.09 - 0.86	2.67 - 11.09	24.95 - 148.47	122.18 - 1505.85		
	$P_{Y'}$	0.06	1.53	19.33	188.19	-	-
		0.02 - 0.44	0.12 - 3.94	1.99 - 183.77	11.96 - 1519.72		
$\alpha = 0.8$	$sh_{\varphi'}$	0.07	1.39	10.61	49.06	93.5	169.5
		(5.84 - 29.38)	(12.86 - 154.62)	(19.4 - 377.63)	(27.33 - 702.3)	(31.27 - 893.4)	(32.97 - 1125.87)
	$sh_{Y'}$	0	0.03	0.09	0.21	0.31	0.43
		(2.68 - 24.28)	(5.8 - 134.34)	(10.37 - 357.53)	(13.7 - 657.43)	(13.83 - 802.6)	(11.47 - 905.17)
	P	0.17	5.78	27.32	57.12	243.09	1195.13
		0.03 - 0.6	0.3 - 69.02	0.67 - 433.42	4.29 - 290.64	5.22 - 4415.44	5.42 - 15903.6
	$P_{\varphi'}$	0.11	1.58	12.49	52.66	101.91	173.31
		0.05 - 0.33	1.19 - 3.52	9.66 - 49.89	44.95 - 105.42	85.84 - 230.68	149.67 - 280.22
	$P_{Y'}$	0.09	2.28	9.68	14.22	232.46	706.79
		0.01 - 0.49	0.06 - 27.97	0.15 - 141.73	0.69 - 82.01	0.88 - 5522.99	1.4 - 15271.3

Table 7: Synthesis of the numerical results for 3 objectives and Hurwicz criterion. Execution times are indicated in seconds.

By observing these tables, it appears that the value of parameter α has opposite impacts on the two phases of the resolution procedures:

- when the number n of vertices increases, the resolution time of the MIP increases with the following sequence of values: $\alpha = 0.4, 0.8, 0.6$;
- the shaving is more efficient (in terms of number of colored edges) when α increases.

Let us elaborate on this latter point. Primitive $sh_{\varphi'}$ is all the more so efficient than the bound provided by φ' is tight. A prerequisite in this aim is that $\sum_i \lambda_i = 1$ in φ', i.e., the weights are normalized. However, it can happen that no set of weights sum up to 1 in Λ (as defined by the constraints in Equation 4 of Section 4.3). Interestingly, when using the Hurwicz criterion, it is possible to simply detect if such a set of normalized weights exists. Let us show that it exists if and only if $\alpha \geq (p-1)/p$. Assume first that condition $\alpha \geq (p-1)/p$ holds. By setting $\lambda_i = 1/p$ for all i, we have $\sum_{i \in I} \lambda_i = |I|/p \leq (p-1)/p$ for $|I| \leq p-1$. Furthermore, by definition of the weights in the Hurwicz criterion,

	n	20	30	40	50	60
$\alpha = 0.4$	$sh_{\varphi'}$	0.26	1.44	5.92	18.99	50.66
		(0 - 0)	(0 - 0)	(0 - 0)	(0 - 0)	(0 - 0)
	$sh_{Y'}$	0.01	0.02	0.05	0.1	0.18
		(0 - 1.37)	(0 - 0.77)	(0 - 0)	(0 - 0)	(0 - 0)
	P	1.19	4.56	18	123.93	460.94
		0.26 - 2.34	2.35 - 10.1	6.45 - 50.76	21.67 - 323.57	40.96 - 4131.31
	$P_{\varphi'}$	1.45	6.05	23.88	143.39	511.73
		0.5 - 2.59	3.64 - 11.84	11.51 - 57.04	38.79 - 352.73	86.94 - 4174.91
	$P_{Y'}$	1.14	4.55	18.14	124.62	461.06
		0.28 - 2.25	2.2 - 10.11	6.59 - 45.25	21.54 - 335.79	40.47 - 4092.85
$\alpha = 0.6$	$sh_{\varphi'}$	0.25	1.43	5.73	17.42	-
		(0 - 0)	(0 - 0)	(0 - 0)	(0 - 0)	
	$sh_{Y'}$	0.01	0.02	0.05	0.1	-
		(0 - 1.4)	(0 - 1.43)	(0 - 0)	(0 - 0)	
	P	1.83	8.91	46.24	367.3	-
		0.88 - 3.39	1.92 - 41.7	7.92 - 155.77	26.34 - 2374.14	
	$P_{\varphi'}$	2.09	10.32	51.95	384.77	-
		1.06 - 3.37	3.1 - 44.26	13.31 - 163.56	41.71 - 2404.13	
	$P_{Y'}$	1.69	9.13	46.12	368.01	-
		0.58 - 2.79	1.61 - 43.76	7.73 - 155.16	25.96 - 2394.34	
$\alpha = 0.8$	$sh_{\varphi'}$	0.2	1	3.7	10.84	26.61
		(0.93 - 17.7)	(2.2 - 54.03)	(2 - 104.33)	(3.03 - 188.3)	(4.46 - 300.13)
	$sh_{Y'}$	0.01	0.02	0.04	0.08	0.14
		(0.67 - 16.4)	(2.13 - 51.73)	(1.9 - 98.37)	(2.97 - 174.43)	(4.2 - 278.97)
	P	1.74	28.07	70.52	297.75	1725.29
		0.58 - 4.3	1.54 - 104.74	6.53 - 211.65	9.67 - 3664.45	6.38 - 16778.8
	$P_{\varphi'}$	1.52	18.62	53.44	217.57	866.861
		0.51 - 3.36	1.16 - 60.74	7.14 - 184.8	23.27 - 1972.08	25.15 - 9426.5
	$P_{Y'}$	1.49	18.41	51.73	225.35	1586.19
		0.5 - 3.05	0.45 - 90.47	4.06 - 225.98	14.53 - 1978.86	2.59 - 29575.1

Table 8: Synthesis of the numerical results for 5 objectives and Hurwicz criterion. Execution times are indicated in seconds.

$\sum_{i=1}^{|I|} w_i = \alpha$ for $1 \leq |I| \leq p-1$. Consequently, for $1 \leq |I| \leq p-1$, we have $\sum_{i\in I} \lambda_i \leq \sum_{i=1}^{|I|} w_i$ since $\alpha \geq (p-1)/p$. Constraint $\sum_{i\in I} \lambda_i \leq 1$ obviously holds for $I = \{1, \ldots, p\}$, which allows to conclude that normalized weights $\lambda_i = 1/p$ satisfy all the constraints in Equation 4. Conversely, assume now that $\alpha < (p-1)/p$. For weights λ in Λ, one has in particular $\sum_{i\in I} \lambda_i \leq \sum_{i=1}^{|I|} w_i = \alpha < (p-1)/p$ for $|I| = p-1$. It implies that $\sum_{I:|I|=p-1} \sum_{i\in I} \lambda_i < p(p-1)/p$, i.e., $(p-1)\sum_{i=1}^{p} \lambda_i < p-1$, which yields $\sum_{i=1}^{p} \lambda_i < 1$. Hence if $\alpha < (p-1)/p$, there is no normalized weights in Λ. Condition $\alpha \geq (p-1)/p$ is therefore necessary and sufficient for the existence of normalized weights in Λ. Coming back to the interpretation of the numerical results, it means that the shaving will be more efficient for $\alpha \geq 0.66$ if $p = 3$, and for $\alpha \geq 0.8$ if $p = 5$. This is consistent with what can be observed in the tables: the number of colored edges is much more important for $\alpha = 0.8$ (which is greater or equal than the required value, both for $p = 3$ and $p = 5$) than for the other values.

Let us now comment on the results of the tests carried out on the k-trimmed criterion (as defined in Section 2.2). Note that primitive $sh_{\varphi'}$ is useless here since all weights in φ' should be set to 0 to satisfy the constraints in Equation 4. Consequently, only primitive $sh_{Y'}$ has been enabled in the tests. Even with this latter primitive, the number of colored edges is very low. It can be indeed easily shown that if the k-trimmed criterion is used, then the bound obtained by relaxation of the image set is equal to $\text{OWA}(b_1, \ldots, b_p)$, where b_i denotes the optimal value on objective i. In a shaving procedure, this bound is of course quite loose since it is over-optimistic. This is confirmed by the numerical tests, summarized in Tables 9 and 10, where we can see that the number of colored edges is almost 0. One can conclude that for this criterion, other preprocessing procedures should be found to reduce the size of the instance.

	n	10	20	30	35	40	45
$k = 1$	$\text{sh}_{Y'}$	0.01 (0.13 - 4.1)	0.04 (0 - 0)	0.12 (0 - 0)	0.19 (0 - 0)	0.29 (0 - 0)	0.41 (0 - 0)
	P	0.17 0.05 - 0.41	4.39 0.87 - 12.85	23.8 3 - 235.7	70.45 14.9 - 660.15	186.3 13.61 - 2346	484.95 40.93 - 6223.8
	$\text{P}_{Y'}$	0.14 0.01 - 0.4	4.42 0.85 - 12.92	23.82 3.06 - 235.91	70.6 14.96 - 660.52	186.47 14.03 - 2340.87	485.82 41.50 - 6226.8

Table 9: Synthesis of the numerical results for 3 objectives and k-trimmed criterion. Execution times are indicated in seconds.

	n	20	30	40	50	60
$k = 1$	$\text{sh}_{Y'}$	0.01 (0 - 0)	0.05 (0 - 0)	0.10 (0 - 0)	0.17 (0 - 0)	0.28 (0 - 0)
	P	1.45 0.46 - 2.41	101.15 6.82 - 2187	1159.53 9.48 - 12874.1	733.14 28.71 - 15268.7	378.87 45.27 - 1504.7
	$\text{P}_{Y'}$	1.45 0.5 - 2.38	101.77 6.76 - 2209.26	1120.3 9.55 - 13655.3	732.07 29 - 15239.2	378.96 45.94 - 1502.62
$k = 2$	$\text{sh}_{\varphi'}$	0.61 (0 - 0)	20.92 (0 - 0)	71.39 (0 - 0)	208.18 (0 - 0)	495.74 (0 - 0)
	P	2.5 1.24 - 5.22	467.84 8.56 - 9778.22	168.41 19.59 - 1028.96	190.68 48.74 - 1215.18	829.43 83.05 - 6406.81
	$\text{P}_{Y'}$	2.49 1.2 - 5.4	433.69 8.7 - 8719.89	168.85 19.65 - 1038.66	190.9 48.72 - 1214.38	828.14 83.23 - 6383.18

Table 10: Synthesis of the numerical results for 5 objectives and k-trimmed criterion. Execution times are indicated in seconds.

6. Conclusion

In this paper we have proposed MIP formulations and preprocessing methods to solve the OWA-optimal spanning tree problem. The validity of the MIP formulations depends on the weights used in the OWA operator. One is valid only when the weights are decreasing and the other one is valid whatever are the weights, but its resolution is more time consuming. We have shown that, when the weights are decreasing, the use of generalized optimality conditions, as well as shaving procedures, makes it possible to considerably reduce the size of the problem, and therefore speeds up the resolution. However, when the weights are arbitrary, the efficiency of the preprocessing procedures strongly depends on the type of used criterion: the shaving procedures enable to reduce the size of the instance prior to its resolution for some subclasses of the Hurwicz criterion, but are useless for the k-trimmed criterion.

Despite a greater resolution time than in the decreasing case, the second MIP formulation proposed here is a first step towards tackling the arbitrary case. A challenging task for future works would be to investigate new preprocessing methods for this case. Another interesting research direction would be to study some interesting variations of OWA, namely the non-monotonic OWA operator [50] (where negative weights are allowed) and the weighted OWA operator [46] (where importance weights specific to each objective are allowed in addition to the weights of the OWA operator). These research tracks are especially important because the search for a single best compromise solution is nearly the only operational approach when there are more than three objectives and the problem does not fit into the dynamic programming framework.

Acknowledgements

This work has been funded by project ANR-09-BLAN-0361 GUaranteed Efficiency for PAReto optimal solutions Determination (GUEPARD), which is gratefully acknowledged.

References

[1] H. Aissi, C. Bazgan, and D. Vanderpooten. Approximating min-max (regret) versions of some polynomial problems. In *COCOON*, pages 428–438, 2006.

[2] H. Aissi, C. Bazgan, and D. Vanderpooten. Approximation of min-max and min-max regret versions of some combinatorial optimization problems. *European Journal of Operational Research*, 179(2):281–290, 2007.

[3] H. Aissi, C. Bazgan, and D. Vanderpooten. Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427–438, 2009.

[4] K.A. Andersen, K. Jörnsten, and M. Lind. On bicriterion minimal spanning trees: An approximation. *Computers & Operations Research*, 23(12):1171–1182, 1996.

[5] G. Andreatta and F.M. Mason. Properties of the *k*-centra in a tree network. *Networks*, 15:21–25, 1985.

[6] N. Boland, P. Domínguez-Marín, S. Nickel, and J. Puerto. Exact procedures for solving the discrete ordered median problem. *Computers & OR*, 33(11):3270–3300, 2006.

[7] M. Ehrgott. *Multicriteria Optimization, second edition*. Springer, 2005.

[8] M. Ehrgott and X. Gandibleux. A survey and annoted bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 22:425–460, 2000.

[9] M. Ehrgott and X. Gandibleux. Approximative solution methods for multiobjective combinatorial optimization. *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research*, 12(1):1–63, 2004.

[10] M. Ehrgott and A. J.V. Skriver. Solving biobjective combinatorial max-ordering problems by ranking methods and a two-phases approach. *European Journal of Operational Research*, 147(3):657–664, 2003.

[11] J. Fodor, J.-L. Marichal, and M. Roubens. Characterization of the ordered weighted averaging operators. *IEEE Transactions on Fuzzy Systems*, 3(2):236–240, 1995.

[12] L. Galand and P. Perny. Search for Choquet-optimal paths under uncertainty. In *Proceedings of the 23rd conference on Uncertainty in Artificial Intelligence*, pages 125–132, Vancouver, Canada, 2007. AAAI Press.

[13] L. Galand, P. Perny, and O. Spanjaard. A branch and bound algorithm for Choquet optimization in multicriteria problems. In T.J. Stewart M. Ehrgott, B. Naujoks and J. Wallenius, editors, *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, volume 634 of *Lecture Notes in Economics and Mathematical Systems*, 2009.

[14] L. Galand and O. Spanjaard. Deux approches complémentaires pour un problème d'arbre couvrant robuste. In *8ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide la Décision*, pages 129–137. Presses Universitaires de Grenoble, 2007.

[15] M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research*, 89(3):445–456, 1996.

[16] H.W. Hamacher and G. Ruhe. On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52:209–230, 1994.

[17] P. Hansen. Bicriterion path problems. In G. Fandel and T. Gal, editors, *Multicriteria Decision Making*, 1980.

[18] S.-P. Hong, S.-J. Chung, and B. H. Park. A fully polynomial bicriteria approximation scheme for the constrained spanning tree problem. *Operations Research Letters*, pages 233–239, 2004.

[19] F. Kappel and A.V. Kuntsevich. An implementation of Shor's *r*-algorithm. *Computational Optimization and Applications*, 15:193–205, 2000.

[20] N. Katoh, T. Ibaraki, and H. Mine. An algorithm for finding *k* minimum spanning trees. *SIAM Journal on Computing*, 10(2):247–255, 1981.

[21] P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*. Kluwer Academic Publisher, 1997.

[22] T.L. Magnanti and L. Wolsey. *Optimal trees*, volume 7, pages 503–615. North-Holland, 1995.

[23] J.L. Marichal. *Aggregation operators for multicriteria decision aid*. PhD thesis, University of Liège, 1998.

[24] A. Marín, S. Nickel, J. Puerto, and S. Velten. A flexible model and efficient solution strategies for discrete location problems. *Discrete Applied Mathematics*, 157(5):1128–1145, 2009.

[25] P.D. Martin and D.B. Shmoys. A new approach to computing optimal schedules for the job shop scheduling problem. In S.T. McCormick W.H. Curnigham and M. Queyranne, editors, *Proceedings fifth international IPCO conference, Vancouver, Canada*, pages 389–403. Lecture Notes in Computer Science 1084, 1996.

[26] H. Moulin. *Axioms of Cooperative Decision Making*. Cambridge Books. Cambridge University Press, 1991.

[27] S. Nickel and J. Puerto. A unified approach to network location problems. *Networks*, 34:283–290, 1999.

[28] S. Nickel and J. Puerto. *Location Theory: A Unified Approach*. Springer-Verlag, Heidberg, Germany, 2005.

[29] S. Nickel, J. Puerto, A. Rodriguez-Chia, and A. Weissler. Multicriteria planar ordered median problems. *Journal of Optimization Theory and Applications*, 126(3):657–683, 2005.

[30] W. Ogryczak. Inequality measures and equitable approaches to location problems. *European Journal of Operational Research*, 122(2):374–391, 2000.

[31] W. Ogryczak. On efficient WOWA optimization for decision support under risk. *International Journal of Approximate Reasoning*, 50:915–928, 2009.

[32] W. Ogryczak and T. Sliwinski. On solving linear programs with the ordered weighted averaging objective. *European Journal of Operational Research*, 148(1):80–91, 2003.

[33] W. Ogryczak and A. Tamir. Minimizing the sum of the *k* largest functions in linear time. *Information Processing Letters*, 85(3):117–122, 2003.

[34] C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *Proceedings of the 41th IEEE Symposium on FOundations of Computer Science*, pages 86–92, 2000.

[35] P. Perny and O. Spanjaard. An axiomatic approach to robustness in search problems with multiple scenarios. In *Proceedings of the 19th conference on Uncertainty in Artificial Intelligence*, pages 469–476, 2003.

[36] P. Perny, O. Spanjaard, and L.-X. Storme. A decision-theoretic approach to robust optimization in multivalued graphs. *Annals of Operations Research*, 147(1):317–341, 2006.

[37] A.P. Punnen and Y. P. Aneja. Minmax combinatorial optimization. *European Journal of Operational Research*, 81(3):634–643, 1995.

[38] R. M. Ramos, S. Alonso, J. Sicilia, and C. Gonzalez. The problem of the optimal biobjective spanning tree. *European Journal of Operational Research*, 111(3):617–628, 1998.
[39] E.E. Rosinger. Beyond preference information based multiple criteria decision making. *European Journal of Operational Research*, 53(2):217–227, 1991.
[40] D. Schmeidler. Integral representation without additivity. *Prooceedings of the American Mathematical Society*, 97(2):255–261, 1986.
[41] D. Schmeidler. Subjective probability and expected utility without additivity. *Econometrica*, 57:571–587, 1989.
[42] N.Z. Shor. *Minimization methods for non-differentiable functions*. Springer-Verlag, New York, 1985.
[43] F. Sourd and O. Spanjaard. A multi-objective branch-and-bound framework. Application to the bi-objective spanning tree problem. *INFORMS Journal of Computing*, 20(3):472–484, 2008.
[44] S. Steiner and T. Radzik. Computing all efficient solutions of the biobjective minimum spanning tree problem. *Computers & Operations Research*, 35(1):198–211, 2008.
[45] R.E. Tarjan. *Data structures and network algorithms*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983.
[46] V. Torra. The weighted OWA operator. *International Journal of Intelligent Systems*, 12:153–166, 1997.
[47] A. Warburton. Worst case analysis of greedy and related heuristics for some min-max combinatorial optimization problems. *Mathematical Programming*, 33:234–241, 1985.
[48] A.P. Wierzbicki. The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *Multiple Criteria Decision Making – Theory and application*. Springer, 1980.
[49] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. In *IEEE Transactions on Systems, Man and Cybernetics*, volume 18, pages 183–190, 1988.
[50] R.R. Yager. Nonmonotonic OWA operators. *Soft Computing*, 3(3):187–196, 1999.
[51] H. Yaman, O.E. Karaşan, and M.Ç. Pinar. The robust spanning tree problem with interval data. *Operations Research Letters*, 29:31–40, 2001.
[52] G. Yu. Min-max optimization of several classical discrete optimization problems. *Journal of Optimization Theory and Applications*, 98(1):221–242, 1998.