A Preference-Based Approach to Spanning Trees and Shortest Paths Problems

Patrice Perny, Olivier Spanjaard

A preference-based approach to spanning trees and shortest paths problems

Patrice Perny [a], Olivier Spanjaard [b]

[a] *LIP6, Univ. Paris VI, 4 place Jussieu, 75252 Paris Cedex 05, France*

[b] *LAMSADE, Univ. Paris IX, Place du M^{al} De Lattre de Tassigny, 75775 Paris Cedex 16, France*

Abstract

Comparison of solutions in combinatorial problems is often based on an additive cost function inducing a complete order on solutions. We investigate here a generalization of the problem, where preferences take the form of a quasi-transitive binary relation defined on the solutions space. We first propose preference-based search algorithms for two classical combinatorial problems, namely the preferred spanning trees problem (a generalization of the minimum spanning tree problem) and the preferred paths problem (a generalization of the shortest path problem). Then, we introduce a very useful axiom for preference relations called *independence*. Using this axiom, we establish admissibility results concerning our preference-based search algorithms. Finally, we address the problem of dealing with non-independent preference relations and provide different possible solutions for different particular problems (*e.g.* lower approximation of the set of preferred solutions for multicriteria spanning trees problems, or relaxation of the independence axiom for interval-valued preferred path problems).

Key words: Preference Modelling, Combinatorial Optimization, Preference-based search, Multiobjective Optimisation.

1 Introduction

In combinatorial optimization, the quality of a potential solution is usually evaluated by a single criterion, mostly a cost function to be minimized, defined as the sum of the costs of its elementary components (*e.g.* in a shortest path problem, the value of a path is often seen as the sum of the values of its arcs). This particular feature makes it possible to design constructive search algorithms based on local optimizations and implicit enumeration for identifying an "optimal" solution. However, in practical situations, preferences over solutions are not always representable by such a numerical cost function and the traditional search algorithms do not fit.

Indeed, even if preferences can be represented by a numerical cost function, the quality of each potential solution is not always decomposable as a function of the quality of its components.

Throughout this paper, we will see several examples in which the value of a path in a graph depends on the whole set of arcs composing the path and cannot be simply defined as a combination of the value of its arcs. Moreover, even if the value of a path is defined as a function of the value of its arcs, the function is not necessarily additive (*e.g* when the valuation represents risk levels attached to each arcs, the value of a path may be the minimum of the values of its arcs).

Moreover, in many practical applications, we have to deal with imprecision or uncertainty about the evaluation of elementary decisions and, consequently, in the evaluation of the quality of a feasible solution. Consider, for example, the case where the arcs of a graph are valued by intervals of possible costs. Assuming the costs are additive, the value of a path becomes itself an interval. Hence, we might decide that a path is at least as good as another path when the minimal possible cost of the former does not exceed the maximal possible cost of the latter. Such a comparison method defines a semi-order preference structure (see [28]) on the set of paths of the graph. Such preferences are quite natural but not transitive (due to intransitivity of the indifference) and therefore not representable by the traditional model based on a single criterion.

Finally, the relative quality of the potential solutions may not even be representable by a single criterion. In many cases, multiple viewpoints must be considered (diverging opinions of different experts, conflicting information provided by various sensors, multi-attribute or multicriteria evaluation of the quality of the solution) to decide whether a solution is better than another. Due to the potentially conflicting nature of these points of view, some pairs of feasible solutions remain incomparable (see e.g. the notion of dominance in multicriteria optimization) and the preference model is not reducible to a single criterion.

In all these examples, we have to deal with a preference structure over potential solutions that could not be represented by an additive scalar cost function to be minimized. Several works have already been done to provide solutions in such contexts, especially in the framework of multicriteria decision making. For example, in combinatorial optimization, several papers deal with the search of non-dominated solutions (Pareto solutions) in multicriteria problems. Among others, we could mention [20,23] for multicriteria shortest paths problems and [6,33,19] for multicriteria spanning trees problems. To go further, additional search algorithms have been proposed to focus on best compromise solutions in multicriteria combinatorial optimization, see for instance [38,16,15]. Besides these studies, algorithms for the search of robust solutions in combinatorial optimization problems (search for an adequate solution when multiple scenarios are considered) have been proposed in [21] and [40].

Similar problems have also been investigated in the framework of artificial intelligence and several heuristic search algorithms have been proposed. Let us mention, for example, the multicriteria versions of the A^* algorithm proposed in [35,15], or the algorithms proposed for game tree search with partially ordered evaluations in [27,9].

Finally, some other works focus on the search of best solutions when costs are qualitative. For example, we could mention [14,3,32] for minimal trees problem, [2] for minimal paths problem,

and [7] for lexicographic bottleneck optimization.

All these papers aim at proposing preference-based search algorithms for combinatorial problems with non-conventional preferences (*i.e.* preferences which are not representable by a single additive cost function). However, until now, the preference-based search problem formulated at this very general level has not received much attention [1] . The goal of this paper is to introduce a general framework for preference-based search in combinatorial problems and to investigate the possibility of obtaining the preferred solutions. After introducing a general formalism for preference-based combinatorial problems (Section 2), we propose various general algorithms for the search of the preferred solutions in combinatorial problems (Section 3). Then we investigate the ability of these algorithms to determine the set of preferred solutions. This lead us to introduce an independence axiom characterizing a subclass of preferences structures on which our algorithms are operational (Section 4). We finally investigate problems where preferences do not satisfy the independence axiom (Section 5) and show the potential use of our algorithm to approximate the desired result.

2 A preference-based framework

Let us first recall the following definitions about binary relations.

Definition 1 *From any binary relation $\succsim$ on a set E, the asymmetric and symmetric parts of $\succsim$ are respectively defined on E by:*

$$\forall e, e' \in E, \quad (e \succ e') \iff ((e \succsim e') \text{ and } not(e' \succsim e))$$
$$\forall e, e' \in E, \quad (e \sim e') \iff ((e \succsim e') \text{ and } (e' \succsim e))$$

Definition 2 *For any binary relation $\succsim$ defined on a set E, the set of* maximal *elements is defined by:*

$$M(E, \succsim) = \{e \in E \quad | \quad \forall e' \in E \quad not(e' \succ e)\}$$

Moreover, for any $X \subseteq E$, the set $M(X, \succsim)$ will be called the set of maximal elements in X.

Note that $M(X, \succsim)$ must not be confused with $M(E, \succsim) \cap X$.

In this paper, $\succsim$ represents a weak-preference relation and therefore $\succ$ is the associated strict preference relation. The proposition $e \succsim e'$ means e is at least as good as e' whereas $e \succ e'$ means e is strictly preferred to e'. In such a context, for any $X \subseteq E$, the elements of $M(X, \succsim)$ are said to be $\succsim$-*efficient in* X.

[1] except in algebraic combinatorial optimization (see e.g. [42,4]), but the link with decision theory has not been stated explicitly.

Definition 3 *A relation $\succsim$ defined on a set T is said to be:*

- *reflexive iff* $\forall e \in E,\ e \succsim e$
- *complete iff* $\forall e, e' \in E,\ e \succsim e'$ *or* $e' \succsim e$
- *antisymmetric iff* $\forall e, e' \in E,\ ((e \succsim e')$ *and* $(e' \succsim e)) \Longrightarrow (e = e')$
- *transitive iff* $\forall e, e', e'' \in E,\ ((e \succsim e')$ *and* $(e' \succsim e'')) \Longrightarrow (e \succsim e'')$
- *quasi-transitive iff* $\succ$ *is transitive*

Definition 4
- *A* partial order *is a reflexive, antisymmetric and transitive binary relation*[2] .
- *A* complete order *is a reflexive, antisymmetric, transitive and complete binary relation.*

Finally, the following notion will be useful.

Definition 5 *Let $G = (V, \succ)$ be a graph representing a partial order $\succ$. We call here* topological sorting *of G any strict complete order $\succ'$ on V such that $\succ' \supseteq \succ$.*

We introduce now the main issue of the paper:

Preference-based graph problems. The basic ingredients of a preference-based graph problem Π are the set of instances or input objects (each instance including the definition of a graph and a preference relation over the set of edges), the set of feasible solutions or output objects associated with any instance, and the goal of the problem. More formally, any preference-based graph problem Π is defined using the following components:

- $\mathcal{I}$ the set of instances of Π;
- Given $I \in \mathcal{I}$, $(V(I), E(I))$ is a finite connected graph ($V(I)$ representing the set of vertices and $E(I)$ the set of edges or arcs);
- Given $I \in \mathcal{I}$, $\mathcal{S}(I) \subseteq \mathcal{P}(E(I))$ is the set of feasible solutions of I;
- Given $I \in \mathcal{I}$, $\succsim (I)$ is a preference relation defined on the power set $\mathcal{P}(E(I))$;
- the goal (goal): we want to determine one or the whole set of maximal feasible solutions defined by $M(\mathcal{S}(I), \succsim (I))$.

Thus, a preference-based graph problem Π is characterized by a four-tuple $(\mathcal{I}, \mathcal{S}, \succsim, \text{goal})$[3] .

Many classical graph problems could be casted in that framework, *e.g.* search of optimal trees, paths, hamiltonian cycles, cuts, matchings... In this paper we focus on preferred spanning trees problems and preferred paths problems which are defined as follows:

$\succsim$-ST (Preferred Spanning Trees)
Input objects: A finite connected graph $G = (V, E)$ and a preference relation $\succsim$ on $\mathcal{P}(E)$;
Output objects: The set $\mathcal{T}$ of spanning trees on G;
Goal: We want to determine the whole set $M(\mathcal{T}, \succsim)$.

[2] Such relations are often denoted $\succ$ since their symmetric part consists only in reflexivity.
[3] For the sake of simplification, we work on a preference relation on $\mathcal{P}(E(I))$. Obviously, we could also work with a preference relation on valuations as well.

$\succsim$-P (Preferred Paths)
Input objects: A finite connected digraph $G = (V, E)$ without circuit, two vertices s and t included in V and a preference relation $\succsim$ on $\mathcal{P}(E)$.
Output objects: The set $\mathcal{P}$ of paths from s to t;
Goal: We want to determine the whole set $M(\mathcal{P}, \succsim)$.

Note that the classical minimum spanning tree and shortest path problems are particular instances of $\succsim$-ST and $\succsim$-P respectively, obtained for a preference defined by: $\forall A, B \subseteq E,\ A \succsim B \iff \sum_{e \in A} v(e) \leq \sum_{e \in B} v(e)$ where $v : E \to \mathbb{R}$ is a cost function.

In the general case, $\succsim$-ST is obviously intractable[4] since we can assume the preference relation $\succsim$ to be empty. In such a case, all the spanning trees of the graph are $\succsim$-efficient. Moreover, as shown by Cayley [5], a complete graph with n vertices has n^{n-2} spanning trees. This shows the intractability of $\succsim$-ST (since the output cannot be described by an expression having length bounded by a polynomial function of the input length). Concerning $\succsim$-P, note that an instance of that problem is the bicriteria shortest paths problem which is known as intractable [20]. As a consequence, $\succsim$-P is also intractable.

3 The algorithms

We propose here preference-based counterparts of standard algorithms used for minimal spanning tree problems and shortest path problems. We will investigate the correctness of these algorithms in the next section. The two following algorithms are designed to determine the set of $\succsim$-efficient spanning trees of a connected graph on n vertices. They realize a width-first search directed by the preference relation $\succsim$.

Our first algorithm (Algorithm 1) consists in a generalization of the Kruskal algorithm (see *e.g.* [18]) where, instead of choosing one edge at each step, we test all $\succsim$-efficient edges among the non-chosen edges which do not create cycles. Let $T_i^{(t)} = (V(T_i^{(t)}), E(T_i^{(t)}))$ be the i^{th} subtree enumerated in the algorithm where $V(T_i^{(t)})$ is the set of vertices and $E(T_i^{(t)})$ is the set of edges. Let t indicate that it contains t edges and $I^{(t)}$ denote the set containing the indexes of such subtrees. When all the subtrees of size t are generated, if there exists two indexes $i \neq j \in I^{(t)}$ such that $T_i^{(t)} = T_j^{(t)}$ then $I^{(t)} = I^{(t)} \setminus \{j\}$.

Our second algorithm (Algorithm 2) consists in a generalization of the Prim algorithm (see *e.g.* [18]), where, instead of choosing one edge at each step, we test all $\succsim$-efficient edges in the cocycle linking covered vertices and non-covered vertices. In order to present more formally the Prim-like algorithm we introduce additional notations. For any $X \subset V$, $\Omega(X)$ denotes the cocycle $\{(v, w) \in E,\ v \in X \text{ and } w \in V \setminus X\}$. The set $V(e)$ denotes the endpoints of e for any $e \in E$. Without loss of generality, the initial vertex is denoted $v_1 \in V$.

[4] Following [17], we shall refer to problems as *intractable* if it is so hard that no polynomial time algorithm can possibly solve it

Algorithm 1 *Kruskal-like algorithm for* $\succsim$-ST

Initialization: $I^{(0)} \leftarrow \{1\}$; $E(T_1^{(0)}) \leftarrow \emptyset$; $j \leftarrow 1$;
For $t \leftarrow 1$ to $n-1$ do
$I^{(t)} \leftarrow \emptyset$;
For every $i \in I^{(t-1)}$ do
For every $e \in M(E \setminus E(T_i^{(t-1)}), \succsim)$ such that $E(T_i^{(t-1)}) \cup \{e\}$ is without cycle do
$j \leftarrow j+1$;
$E(T_j^{(t)}) \leftarrow \mathrm{E}(T_i^{(t-1)}) \cup \{e\}$;
$I^{(t)} \leftarrow I^{(t)} \cup \{j\}$;
end
end
Eliminate the duplicates in the sequence $(T_i^{(t)} \mid i \in I^{(t)})$;
end
Output: $\{T_i^{(n-1)} \mid i \in I^{(n-1)}\}$;
end
end

Algorithm 2 *Prim-like algorithm for* $\succsim$-ST

Initialization: $I^{(0)} \leftarrow \{1\}$; $T_1^{(0)} \leftarrow (\{v_1\}, \emptyset)$; $j \leftarrow 1$;
For $t \leftarrow 1$ to $n-1$ do
$I^{(t)} \leftarrow \emptyset$;
For any $i \in I^{(t-1)}$ do
For any edge $e \in M(\Omega(V(T_i^{(t-1)})), \succsim)$
$j \leftarrow j+1$;
$V(T_j^{(t)}) \leftarrow V(T_i^{(t-1)}) \cup V(e)$;
$E(T_j^{(t)}) \leftarrow E(T_i^{(t-1)}) \cup \{e\}$;
$I^{(t)} \leftarrow I^{(t)} \cup \{j\}$;
end
end
Eliminate the duplicates in the sequence $(T_i^{(t)} \mid i \in I^{(t)})$;
end
Output: $\{T_i^{n-1} \mid i \in I^{(n-1)}\}$;
end

Our third algorithm is designed to determine the set of $\succsim$-efficient paths from s to t in a connected acyclic digraph. It consists in a generalization of the Bellman algorithm (see *e.g.* [18]), where at each step and at each vertex v one keeps all $\succsim$-efficient paths from s to v in a label-set $L(v)$. Denoting $\Gamma^{-1}(v) = \{w \in V, (w,v) \in E\}$ the set of precedents of v, the algorithm writes as follows:

Algorithm 3 *Bellman-like algorithm for* $\succsim$-P
$L(s) \leftarrow \emptyset$;
For any unlabelled vertex v for which all the previous are labelled do
$L(v) \leftarrow M(\bigcup_{w \in \Gamma^{-1}(v)} \{P \cup (w,v), \quad P \in L(w)\}, \succsim)$;
end
Output: $L(t)$;
end

4 The independence property

We investigate now under which conditions the above algorithms reach their goal. Consider the graph pictured on the left upper corner of Figure 1 with 3 colored edges (Blue, Yellow, Red). Assume we want to use Algorithm 1 to determine the $\succsim$-efficient spanning trees of this graph, for the preference relation $\succsim$ defined on subsets of colors[5] and pictured on the left side of Figure 2 (case 1). On this preference graph, every vertex represents a subset of colors and every arc represents a strict preference of type $A \succ B$ between two subsets of colors A and B. Note that this preference relation is *strictly monotonic* with respect to set inclusion (*i.e.* $A \subset B$ implies $A \succ B$ for all subsets A, B). Since the only strict-preference that holds on singletons is Blue $\succ$ Yellow, the $\succsim$-efficient edges of the graph are the Blue one and the Red one. Applying Algorithm 1, we get two $\succsim$-efficient trees which are {Blue, Yellow} and {Blue, Red}. The complete search tree outputs {Blue, Yellow} and {Blue, Red} as pictured on the right side of Figure 1. Note that this result is correct; we remark that we would obtain the same search tree and the same result with Algorithm 2 starting from the starred vertex on Figure 1.

Similarly, consider the colored graph pictured at the bottom of Figure 1 with 5 arcs. When applying Algorithm 3 with the same preferences on colors to identify the $\succsim$-efficient paths from the leftmost vertex to the rightmost, we obtain the upper path as the preferred solution with colors {Blue, Red} (the labels obtained at each step and storing the $\succsim$-efficient subpaths at each intermediary vertex are represented by boxes above the vertices). Hence the path {Blue, Red} is output by Algorithm 3, which is the right result in case 1.

However, in the general case, our preference-based search algorithms may lead to non $\succsim$-efficient solutions. This can be easily shown by considering the preference relation pictured on the right side of Figure 2 (case 2). It is clear that the outputs of Algorithms 1, 2 and 3 on the previous problems would remain unchanged after substituting the preference relation of case 2 for the initial one. Indeed, by definition, Algorithms 1 and 2 take only into account preferences on singletons (note that such preferences are identical in case 1 and 2). Concerning Algorithm 3, the search is early conditioned by preferences on partial solutions (*myopic view*), which explains the result. More precisely, the following change in the preference relation misleads the

[5] The preference relation on subsets of arcs directly derives from preferences on subsets of colors. For the sake of simplification, we use the latter instead of the former. Moreover, we mean a color to indicate an edge.

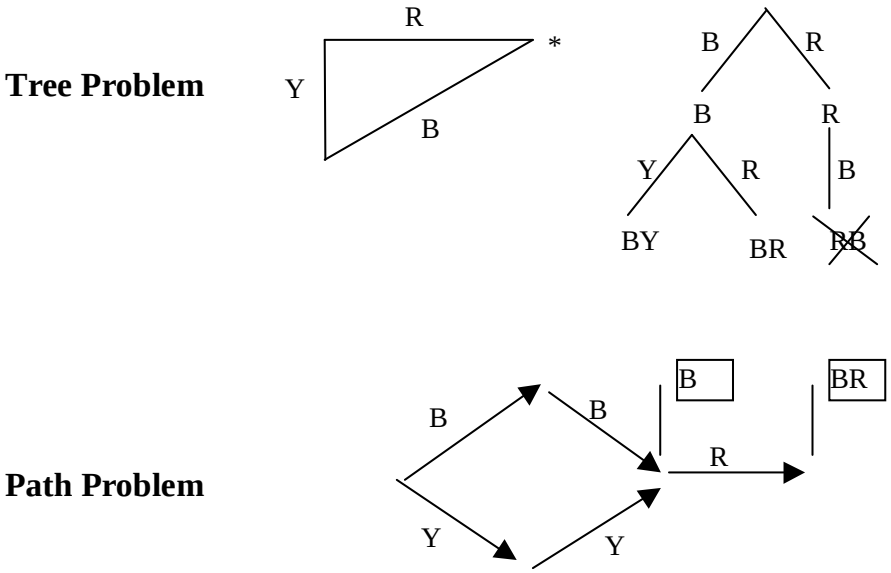

Fig. 1. Illustration of the algorithms.

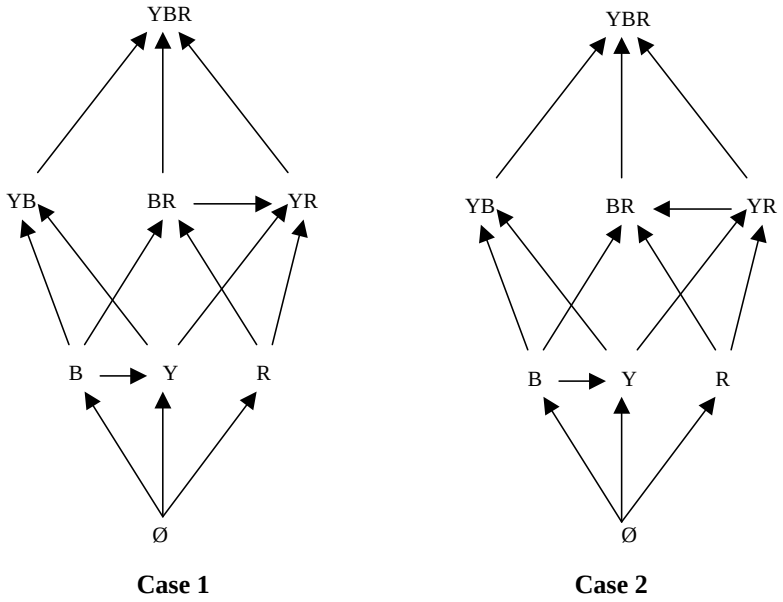

Fig. 2. Two preference relations.

algorithms:

$$\{B\} \succ \{Y\} \text{ and } \{B, R\} \succ \{Y, R\} \text{ in case 1}$$

whereas:

$$\{B\} \succ \{Y\} \text{ but } \{Y, R\} \succ \{B, R\} \text{ in case 2}$$

In case 1, the preference between $\{B\} \succ \{Y\}$ remains unchanged by addition of $\{R\}$ on both sides of the inequality. This property is no longer true in case 2.

The above negative examples have been obtained from a non-complete and non-linear preference relation. We might think that these properties are involved in the malfunction of the algorithms. However, even if the preference relation $\succsim$ induces a complete order on $\mathcal{P}(E)$, Algorithms 1, 2 and 3 might lead to non $\succsim$-efficient solutions, as shown in the following examples:

Example 6 *Let G be a complete graph with three vertices. Let a, b, c be the edges of the graph. Suppose that $a \succ b \succ c \succ \{a,c\} \succ \{a,b\} \succ \{b,c\} \succ \{a,b,c\}$. Note that we have:*

$$b \succ c \text{ and however } \{a,c\} \succ \{a,b\}$$

Hence, Algorithm 1 yields $\{a,b\}$ as output instead of $\{a,c\}$ which is the only $\succsim$-efficient solution. Algorithm 2 yields the same result whatever the starting vertex.

Example 7 *Let G be a graph on 4 vertices with the following arcs: $(v_1,v_2), (v_1,v_3), (v_1,v_4), (v_2,v_4), (v_3,v_4)$. The qualitative valuations are: $v(v_1,v_2) = c, v(v_1,v_3) = d, v(v_1,v_4) = a, v(v_2,v_4) = b$, $v(v_3,v_4) = d$. Suppose that $\succsim$ is a strictly monotonic preference relation inducing a complete order on $\mathcal{P}(\{a,b,c,d\})$ such that $a \succ b \succ c \succ d$ and $\{a,c,d\} \succ \{a,b,d\}$. Let us notice that:*

$$b \succ c \text{ and however } \{a,c,d\} \succ \{a,b,d\}$$

Applying Algorithm 1 yields the two trees valued $\{a,b,d\}$ whereas any tree with valuations $\{a,c,d\}$ was strictly better. The same statement holds with Algorithm 2 whatever the starting vertex.

Thus, we have to study on which class of preference structures our preference-based search algorithms are able to determine the set of $\succsim$-efficient solutions. In this respect, we introduce now the independence axiom:

Independence (I)

$$\forall A, B, C \in \mathcal{P}(E) \text{ such that } C \cap (A \cup B) = \emptyset, \quad (A \succ B \Longrightarrow A \cup C \succ B \cup C)$$

This independence axiom is a weak version of the De Finetti's qualitative additivity [10]. It can also be seen as a qualitative counterpart of the monotonicity property considered in dynamic programming (see *e.g* [24,25]). More generally, let us consider $v : \mathcal{P}(E) \to U$ a valuation function (where $(U, \succ_U)$ is a partially ordered set) such that $v(A) \succ_U v(B) \iff A \succ B$, and $\otimes$ an internal composition operator on U such that $v(A \cup B) = v(A) \otimes v(B)$ for all $A, B \in \mathcal{P}(E)$. Then, considering the ordered semigroup $(U, \otimes, \succ_U)$, axiom (I) translates into monotonicity of $\otimes$ over $\succ_U$. This type of monotonicity is a classical assumption in the algebraic path problem [42,30].

The independence condition is naturally satisfied in various classical problems. We give here some examples of usual preference relations obviously satisfying axiom (I).

Example 8 *Assuming that v is a valuation on E, axiom (I) holds for the numerical additive*

preference relation defined by:

$$A \succsim B \iff \sum_{e \in A} v(e) \leq \sum_{e \in B} v(e)$$

This example shows that, in the classical framework, we choose implicitly a preference relation that does satisfy the independence axiom. Now, in the context of multicriteria optimization where the quality of each element $e \in E$ is defined by the valuation vector $(v_1(e), \ldots, v_q(e))$, we have the following result:

Example 9 *In multicriteria problems, axiom (I) holds for lexicographic preference relations defined as follows:*

$$A \succ B \iff \exists k \in \{1, \ldots, q\},\ \sum_{e \in A} v_k(e) < \sum_{e \in B} v_k(e) \text{ and } \forall j < k \quad \sum_{e \in A} v_j(e) = \sum_{e \in B} v_j(e)$$

The same result holds for the dominance relation and other oligarchic preferences as shown by the following:

Example 10 *In multicriteria problems, axiom (I) holds for oligarchic preference relations defined for any subset of criteria $O \subseteq \{1, \ldots, q\}$ by:*

$$A \succsim B \iff (\forall j \in O, \sum_{e \in A} v_j(e) \leq \sum_{e \in B} v_j(e))$$

Our last example concerns a preference relation on sets derived from a preference relation on elements using a construction proposed in [2]:

Definition 11 *Let $\succsim_E$ be a reflexive preference relation on a finite set E. We define the preference relation $\succsim_P$ on $\mathcal{P}(E)$ by: $A \succsim_P B$ if and only if $|B| \geq |A|$ and there exists an injective mapping $\pi : A \to B$ such that $\forall a \in A,\ a \succsim_E \pi(a)$.*

Hence, we have the following result:

Lemma 12 *If $\succsim_E$ is transitive then $\succsim_P$ is transitive.*

PROOF. If $A \succsim_P B$ and $B \succsim_P C$ there exist two injective mappings $\pi_1 : A \to B$ and $\pi_2 : B \to C$ such that:

$$\forall a \in A, \quad a \succsim_E \pi_1(a) \succsim_E \pi_2 \circ \pi_1(a)$$

By transitivity of $\succsim_E$, $\forall a \in A \quad a \succsim_E \pi_2 \circ \pi_1(a)$. Moreover, the composition of injective mappings is an injective mapping. □

Moreover, we have the following positive result:

Proposition 13 *If $\succsim_E$ is reflexive and transitive, then*

$$\forall A, B, C \in \mathcal{P}(E) \text{ such that } C \cap (A \cup B) = \emptyset,\ (A \succsim_P B \iff A \cup C \succsim_P B \cup C)$$

PROOF. Consider A, B, C, three subsets of edges such that $C \cap (A \cup B) = \emptyset$. We establish the result in two stages:

• $A \succsim_P B \Longrightarrow A \cup C \succsim_P B \cup C$
We know that there exists $\pi : A \to B$ an injective mapping such that $\forall a \in A, a \succsim_E \pi(a)$. As π is a mapping and $A \cap C = \emptyset$, we can construct a mapping $\mu : A \cup C \to B \cup C$ as follows :

$$\begin{cases} \forall a \in A,\ \mu(a) = \pi(a) \\ \forall c \in C,\ \mu(c) = c \end{cases}$$

As π is injective and $B \cap C = \emptyset$, μ is injective. Moreover, we have:

$$\begin{cases} \forall a \in A,\ a \succsim_E \mu(a) \\ \forall c \in C,\ c \succsim_E \mu(c) \text{ (reflexivity of } \succsim_E) \end{cases}$$

Therefore we get: $A \cup C \succsim_P B \cup C$.

• $A \cup C \succsim_P B \cup C \Longrightarrow A \succsim_P B$
By construction, we know that there exists an injective mapping verifying:

$$\forall e \in A \cup C,\ e \succsim_E \pi(e)$$

For any $a \in A$, we set $\pi^0(a) = a$, $\pi^k(a) = \pi \circ \pi^{k-1}(a)$ for $k \geq 1$, and

$$k(a) = \begin{cases} \min\{k \mid k \geq 1 \text{ and } \pi^k(a) \notin C\} & \text{if } \exists k \geq 1, \pi^k(a) \notin C \\ +\infty \ \text{ otherwise} \end{cases}$$

We claim that, for any a in A we have: $\forall k \in \{2, \ldots, k(a)\},\ \pi^k(a) \notin \{\pi^l(a), 1 \leq l \leq k-1\}$. Indeed, if that is not the case, we can define (i, j) as the smallest pair such that $\pi^i(a) = \pi^j(a)$. By construction, we know that $\pi^{(i-1)}(a) \neq \pi^{(j-1)}(a)$ whereas $\pi(\pi^{(i-1)}(a)) = \pi(\pi^{(j-1)}(a))$ which contradicts the injectivity of π (if $i = 1$, $\pi^{j-1}(a) \neq \pi^0(a)$ since $A \cap C = \emptyset$ by hypothesis). Hence the $\pi^k(a)$ for $1 \leq k \leq k(a)$ are all different and therefore $k(a) \leq |C| + 1$. As C is finite, $k(a)$ is finite and for all a in A, $\pi^{k(a)}(a)$ is well defined. Let $\mu : A \longrightarrow B$ be the mapping defined by: $\forall a \in A, \mu(a) = \pi^{k(a)}(a)$. We prove the following assertions: i) μ is an injection; ii) $\forall a \in A,\ \ a \succsim_E \mu(a)$.

proof of i). Assume that $\mu(a) = \mu(a')$ for some pair $(a, a') \in A \times A$ such that $a \neq a'$. Then, the sequences $(\pi^k(a))_{k \geq 1}$ and $(\pi^l(a'))_{l \geq 1}$ necessarily meet at some point. Let (i, j) be the smallest

pair of indices such that $\pi^i(a) = \pi^j(a')$ $(i, j \neq 0$ since $a \neq a')$. By definition, we know that $\pi^{(i-1)}(a) \neq \pi^{(j-1)}(a')$ whereas $\pi(\pi^{(i-1)}(a)) = \pi(\pi^{(j-1)}(a'))$ which contradicts the injectivity of π.

proof of ii). By transitivity of $\succsim_E$, $a \succsim_E \pi(a) \succsim_E \ldots \succsim_E \pi^{k(a)}(a) = \mu(a) \Longrightarrow a \succsim_E \mu(a)$. □

From Proposition 13 we derive immediately the following:

Corollary 14 *If $\succsim_E$ is reflexive and transitive, then $\succsim_P$ satisfies (I).*

The importance of axiom (I) in $\succsim$-ST problems is established by the following result which generalizes a result from Serafini [33] (see also [12]):

Theorem 15 *If $\succsim$ is quasi-transitive and satisfies (I), for any $\succsim$-efficient spanning tree T there exists a topological sorting of the preference graph on E for which Kruskal algorithm yields T.*

PROOF. Let $E = \{e_1, \ldots, e_m\}$ be the set of edges of the initial graph and $G = (E, \succ)$ be the preference graph, where $\succ = \{(e_i, e_j)$ such that $e_i \succ e_j\}$. G is without circuit since $\succsim$ is quasi-transitive. Let $A \subseteq E$ be a $\succsim$-efficient spanning tree and $B = E \setminus A$. Let us consider the augmented preference graph:

$$G_A = (E, \succ \cup F_A) \text{ where } F_A = \bigcup_{b \in B} \{(a, b) : a \in C(b)\}$$

(where $C(b)$ is the chain linking the endpoints of b in A).

To illustrate these notions, we give an example on Figure 3 (the edges are $\{x, y, \alpha, \beta, \gamma\}$ and the preference relation is: $x \succ y$, $\alpha \succ \beta \succ \gamma$).

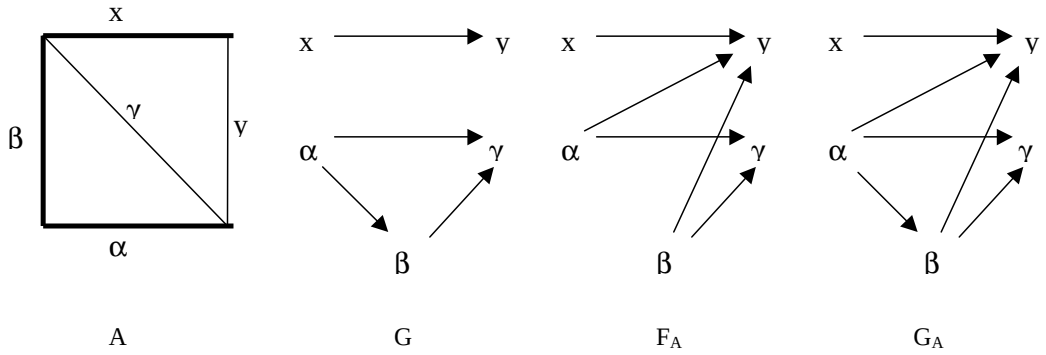

Fig. 3. The preference graph.

We show now that G_A does not contain a circuit. Assume there exist circuits in G_A. Let $\mathcal{C}$ be a circuit of minimum size (among them). Since there is no loop, $\mathcal{C}$ contains at least two arcs. We claim that:

i) $\mathcal{C}$ iteratively alternates one vertex from A and one vertex from B,
ii) $\mathcal{C}$ iteratively alternates one arc from F_A and one arc from $\succ$.

proof of i). Assume there exist (v,a), (a,a') in $\mathcal{C}$ with $a,a' \in A$. As $(v,a) \in \succ$ [6] and $(a,a') \in \succ$, we deduce that $(v,a') \in \succ$ by quasi-transitivity of $\succsim$. So we get a contradiction with respect to the minimality of $\mathcal{C}$. Similarly, if there exists a sequence (b,b'), (b',v) in $\mathcal{C}$ with $b,b' \in B$, we have $(b,b') \in \succ$ and $(b',v) \in \succ$, which yields a contradiction.

proof of ii). By construction, any arc $(b,a) \in B \times A$ does not belong to F_A and therefore belongs to $\succ$. Suppose now there exists an arc $(a,b) \in A \times B$ such that $(a,b) \notin F_A$ in $\mathcal{C}$. Then $(a,b) \in \succ$. As there is no loop, there exists an arc $(b,v) \in \succ$ in $\mathcal{C}$ ($v \neq a$ by quasi-transitivity of $\succsim$). By quasi-transitivity of $\succsim$, we get a contradiction with respect to the minimality of $\mathcal{C}$.

We have proved that $\mathcal{C}$ is of the following form: $(a_1, b_1, \ldots, a_k, b_k, a_1)$ where a_i's are in A and b_i's are in B. Let $C = \{a_i, i = 1, \ldots, k\} \cup \{b_i, i = 1, \ldots, k\}$. Now, we show by contradiction that $A' = (A \cup (C \cap B)) \setminus (C \cap A)$ is a spanning tree. Assume that A' is not a spanning tree. Then, consider the sequence of partial graphs of G defined by $A_0 = A$, $A_i = (A_{i-1} \setminus \{a_i\}) \cup \{b_i\}$ for $i = 1, \ldots, k$. Thus we have $A_k = A'$. Note that the following proposition holds for any $i \in \{1, \ldots, k\}$:

$[A_{i-1}$ is a spanning tree of G and $a_i \in C_{i-1}(b_i)] \Longrightarrow A_i$ is a spanning tree of $G \quad (*)$

where $C_{i-1}(b_i)$ denotes the chain in A_{i-1} linking the endpoints of b_i. Let j be the smallest index in $\{1, \ldots, k\}$ such that A_j is not a tree. Then thanks to $(*)$ we deduce that $a_j \notin C_{j-1}(b_j)$. Therefore an edge $a_l \in C(b_j), l \in \{1, \ldots, j-1\}$ initially present in A has been removed (since $(a_j, b_j) \in F_A \Rightarrow a_j \in C(b_j)$). Hence, $(a_l, b_j) \in F_A$ and constitutes a shortcut of $\mathcal{C}$ which contradicts its minimality. Therefore, A' is a spanning tree.

In order to show that $A' \succ A$, let us first note that A' could be derived from A by iteratively replacing a_{i+1} by b_i, for all $i = 1, \ldots, k-1$, and then a_1 by b_k. From the sequence of preferences $b_i \succ a_{i+1}$ ($i \leq k-1$) and $b_k \succ a_1$, we deduce by (I) and quasi-transitivity of $\succsim$ that $A' \succ A$ which contradicts the $\succsim$-efficiency of A. Therefore there is no circuit in G_A and the notion of topological sorting of G_A makes sense.

Finally, we prove by contradiction that the Kruskal algorithm directed by any topological sorting of G_A yields T. Assume that the Kruskal algorithm yields a tree different from T for a given topological sorting of G_A. Let e_1 be the first edge outside T which is selected during the search. By definition of G_A, we know that any edge e outside T is ranked after all the edges of $C(e)$ in the topological sorting. Consequently, as long as there remains an edge in $C(e_1)$ which is not selected, the Kruskal algorithm cannot select e_1 (note that all the edges of $C(e_1)$ cannot create a cycle since all the selected edges are in T). Once all the edges of $C(e_1)$ have been selected, e_1 will not be selected since it would create a cycle, which leads to a contradiction. □

From the previous result we get the following important corollary:

[6] Any arc in F_A gets its initial endpoint in A and its terminal endpoint in B.

Corollary 16 *If $\succsim$ is quasi-transitive and satisfies (I), then Algorithm 1 yields a superset of $M(\mathcal{T}, \succsim)$.*

PROOF. At the end of the algorithm, $\bigcup_{j \in I^{(n-1)}} T_j^{(n-1)}$ is a superset of $M(\mathcal{T}, \succsim)$ by Theorem 15 since implicitly we enumerate all the topological sortings of the preference graph in our search tree. Indeed, we develop a search tree $\mathcal{A}$ such that at any vertex the path from the root to that vertex represents a set A of selected elements and each branch starting from that vertex represents an element $e \succsim$-efficient in $E \setminus A$ such that $A \cup \{e\}$ is acyclic. The length of a branch is equal to $n-1$. Thus, we enumerate all the trees of $\bigcup_j M(\mathcal{T}, \succ'_j)$ for all topological sortings $\succ'_j$ of the preference graph (with duplicates since in each branch we do not explicit the complete order on non-considered elements). □

However, non $\succsim$-efficient trees may be generated by the Algorithm 1, as it will be shown in Section 5. Nevertheless, it suffices to apply a test of $\succsim$-efficiency on the output set of the algorithm to get the set of $\succsim$-efficient trees.

Another approach to get all $\succsim$-efficient trees of the graph consists in a generalization of the Prim algorithm. In order to prove the correctness of Algorithm 2, we need the following lemma:

Lemma 17 *Let X be a set and $\succsim$ a quasi-transitive preference relation on X, then for any finite set $Y \subset X$ such that $Y \cap M(X, \succsim) = \emptyset$ then:*

$$\forall y \in Y,\ \exists x \in X \setminus Y,\ x \succ y$$

PROOF. For any $y \in Y$, we know that $y \notin M(X, \succsim)$ and therefore there exists $z_1 \in X$ such that $z_1 \succ y$. If $z_1 \in X \setminus Y$ then the result follows. If $z_1 \in Y$, then there exists $z_2 \in X$ such that $z_2 \succ z_1$. By this way, we construct a sequence $z_1, \ldots, z_n$ of distinct elements in X. As Y is finite there exists k such that $z_k \in X \setminus Y$. Hence we have $z_k \succ z_{k-1} \succ \ldots z_1 \succ y$ and therefore $z_k \succ y$ by quasi-transitivity of $\succsim$ which establishes the result. □

Hence, we can establish the following theorem:

Theorem 18 *Let T be a $\succsim$-efficient spanning tree of G. If $\succsim$ is quasi-transitive and satisfies (I), then for any cocycle $\Omega \neq \emptyset$ of G, there is an edge in $\Omega \cap T$ that is $\succsim$-efficient in Ω.*

PROOF. Let A be a $\succsim$-efficient spanning tree on the graph $G = (V, E)$ and $B = E \setminus A$. We prove the result by contradiction. Let us assume that there exists a cocycle $\Omega \neq \emptyset$ such that there is no edge of $\Omega \cap A$ that is $\succsim$-efficient in Ω. There exists $a_1 \in \Omega \cap A$ since Ω is a cocycle. Then, by Lemma 17, there exists $b_1 \in \Omega \cap B$ such that $b_1 \succ a_1$. We cannot have $a_1 \in C(b_1)$ [7]. Indeed, if $a_1 \in C(b_1)$ then $b_1 \succ a_1$ implies $(A \setminus \{a_1\}) \cup \{b_1\} \succ (A \setminus \{a_1\}) \cup \{a_1\} = A$ by (I)

[7] Here again, $C(b_1)$ denotes the chain in A linking the endpoints of b_1.

which contradicts the $\succsim$-efficiency of A since by construction $(A \setminus \{a_1\}) \cup \{b_1\}$ is a tree. Since $a_1 \notin C(b_1)$ and Ω is a cocycle, the tree A necessarily contains another edge $a_2 \in \Omega \cap C(b_1)$.

Following this way, we construct two sequences (a_p) and (b_p) of distinct elements by taking alternately an edge $b_{p-1} \succ a_{p-1}$ in $\Omega \cap B$ and an edge $a_p \in C(b_{p-1}) \cap \Omega$ so as to satisfy the following properties:

$$\forall k \in \{1, \ldots, p-1\}, C(b_k) \cap \{a_1, \ldots, a_k\} = \emptyset \quad (1)$$
$$\forall k \in \{1, \ldots, p-1\}, \forall j \in \{1, \ldots, k\}, \text{ not}(b_j \succ a_{k+1}) \quad (2)$$

Note that Equation 1 implies that $a_1, \ldots, a_p$ are all distinct (since $a_{k+1} \in C(b_k)$) and Equation 2 implies that $b_1, \ldots, b_{p-1}$ are all distinct (since $b_{k+1} \succ a_{k+1}$).

• *Definition of b_p from $\{a_1, \ldots, a_p\}$ and $\{b_1, \ldots, b_{p-1}\}$*

As a_p is not $\succsim$-efficient and not$(b_j \succ a_p)$ for all $j \in \{1, \ldots, p-1\}$, we deduce that there exists an edge $b_p \in \Omega \cap B$ such that $b_p \succ a_p$ and $b_p \notin \{b_1, \ldots, b_{p-1}\}$.

Now, we show by contradiction that:

$$C(b_p) \cap \{a_1, \ldots, a_p\} = \emptyset \quad (*)$$

Assume that there exists $j \in \{1, \ldots, p\}$ such that $a_j \in C(b_p)$. Then, there exists a greatest j such that $a_j \in C(b_p)$, so that:

$$\forall k \in \{j+1, \ldots, p\}, \; C(b_k) \cap \{a_{j+1}, \ldots, a_k\} = \emptyset \quad (3)$$

Let us show now that the set of edges $A' = (A \setminus \{a_j, \ldots, a_p\}) \cup \{b_j, \ldots, b_p\}$ is a spanning tree on G. This can be established by considering the following finite sequence $(A_{j-1}, \ldots, A_p)$ of sets of edges defined by $A_{j-1} = A$, $A_k = (A_{k-1} \setminus \{a_{k+1}\}) \cup \{b_k\}$ for $k = j, \ldots, p-1$ and $A_p = (A_{p-1} \setminus \{a_j\}) \cup \{b_p\} = A'$. We claim that if A_{k-1} is a spanning tree ($k \in \{j, \ldots, p-1\}$), then A_k is a spanning tree. Indeed, for $k \in \{j, \ldots, p-1\}$, Equation 3 implies that $C_{A_{k-1}}(b_k) = C(b_k)$ where $C_{A_{k-1}}(b_k)$ denotes the cycle generated by b_k in A_{k-1}. Consequently we have $a_{k+1} \in C_{A_{k-1}}(b_k)$. Since $A_{j-1} = A$ is a spanning tree, A_{p-1} is therefore a spanning tree. Moreover, Equation 3 implies that $C_{A_{p-1}}(b_p) = C(b_p)$ and consequently $a_j \in C_{A_{p-1}}(b_p)$. So $A_p = A'$ is a spanning tree.

Furthermore, we show that $A' \succ A$. Let us first note that A' could be derived from A by iteratively replacing a_k by b_k, for all $k = j, \ldots, p$. From the sequence of preferences $b_k \succ a_k$ for $k = j, \ldots, p$, we deduce by (I) and quasi-transitivity of $\succsim$ that $A' \succ A$ which contradicts the $\succsim$-efficiency of A and establishes $(*)$.

• *Definition of a_{p+1} from $\{a_1, \ldots, a_p\}$ and $\{b_1, \ldots, b_p\}$*

By Equation $(*)$ and since Ω is a cocycle, the tree A necessarily contains another edge $a_{p+1} \in C(b_p) \cap \Omega$. As $C(b_p) \cap \{a_1, \ldots, a_p\} = \emptyset$ we deduce that $a_{p+1} \notin \{a_1, \ldots, a_p\}$. We show now by

contradiction that:

$$\forall j \in \{1, \ldots, p\},\ \text{not}(b_j \succ a_{p+1}) \quad (**)$$

Assume that there exists $j \in \{1, \ldots, p\}$ such that $b_j \succ a_{p+1}$.

Let us show that the set of edges $A' = (A \setminus \{a_{j+1}, \ldots, a_{p+1}\}) \cup \{b_j, \ldots, b_p\}$ does form a spanning tree on G. This can be established by considering the following finite sequence $(A_{j-1}, \ldots, A_p)$ of sets of edges defined by $A_{j-1} = A$, $A_k = (A_{k-1} \setminus \{a_{k+1}\}) \cup \{b_k\}$ for $k = j, \ldots, p$. We claim that if A_{k-1} is a spanning tree ($k \in \{j, \ldots, p\}$), then A_k is a spanning tree. Indeed, for $k \in \{j, \ldots, p\}$, Equations 1 and $(*)$ imply that $C_{A_{k-1}}(b_k) = C(b_k)$ where $C_{A_{k-1}}(b_k)$ denotes the cycle generated by b_k in A_{k-1}. Consequently we have $a_{k+1} \in C_{A_{k-1}}(b_k)$. Since $A_{j-1} = A$ is a spanning tree, A_p is therefore a spanning tree.

Furthermore, we show that $A' \succ A$. Let us first note that A' could be derived from A by iteratively replacing a_k by b_k, for all $k = j+1, \ldots, p$, and then a_{p+1} by b_j. From the sequence of preferences $b_k \succ a_k$ for $k = j+1, \ldots, p$ and $b_j \succ a_{p+1}$, we deduce by (I) and quasi-transitivity of $\succsim$ that $A' \succ A$ which contradicts the $\succsim$-efficiency of A and establishes $(**)$.

Note that Equation $(*)$ for index $p = |\Omega \cap A|$ writes $C(b_{|\Omega \cap A|}) \cap \{a_1, \ldots, a_{|\Omega \cap A|}\} = \emptyset$. As $\{a_1, \ldots, a_{|\Omega \cap A|}\} = \Omega \cap A$, we get $C(b_{|\Omega \cap A|}) \cap \Omega \cap A = \emptyset$ which is impossible since Ω is a cocycle. Thus, we get a contradiction and the result follows. □

Now, we claim:

Corollary 19 *If $\succsim$ is quasi-transitive and satisfies (I) then Algorithm 2 for $\succsim$-ST yields a superset of the set of $\succsim$-efficient spanning trees.*

PROOF. Since each cocycle is met once in a branch of the search tree during the execution of the algorithm, we deduce from Theorem 18 that every $\succsim$-efficient tree is reachable by Algorithm 2 for $\succsim$-ST. □

Note that non $\succsim$-efficient trees might be generated by Algorithm 2 (here again, note that we can get the set of $\succsim$-efficient trees by applying a test of $\succsim$-efficiency on the output set of the algorithm). Indeed, consider the graph on Figure 4 with a reflexive and transitive preference relation $\succsim_E$ on the edges whose asymmetric part is ($a \succ_E b \succ_E c$, $\alpha \succ_E \beta \succ_E \gamma$); and consider the $\succsim_P$ preference relation of Definition 11. We see that the tree $\{b, \alpha, \gamma\}$ in bold is not $\succsim_P$-efficient (since $\{a, \alpha, \beta\} \succ_P \{b, \alpha, \gamma\}$), although each edge we choose, starting from the surrounded vertex, is $\succsim_E$-efficient in its cocycle and the preference relation $\succsim_P$ satisfies quasi-transitivity and the independence axiom, as shown in Lemma 12 (since transitivity implies quasi-transitivity) and Corollary 14. It is important to note that such a problem cannot occur with Algorithm 1 when the preference is defined like in Definition 11. Indeed, in this case we have:

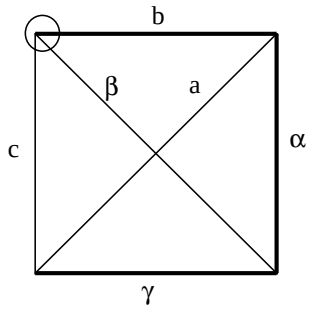

Fig. 4. The bold tree is not $\succsim_P$-efficient.

Proposition 20 *If $\succsim_E$ induces a partial order on E then Algorithm 1 for $\succsim_P$-ST yields exactly the set of $\succsim_P$-efficient spanning trees.*

PROOF. Since $\succsim_E$ is reflexive and transitive, we deduce from Lemma 12 and Corollary 14 that $\succsim_P$ is transitive and satisfies (I). Thus, by Corollary 16, Algorithm 1 yields a superset of the set of $\succsim_P$-efficient spanning trees.

Consider a spanning tree T that is not $\succsim_P$-efficient. Then, there exists T' such that $T' \succ_P T$. By Proposition 13, it implies that $T' \setminus (T \cap T') \succ_P T \setminus (T \cap T')$. Let call $X = (T \cup T') \setminus (T \cap T')$, $Y = T \setminus (T \cap T')$ and $Z = T' \setminus (T \cap T')$. Note that $Y \cup Z = X$ and $Y \cap Z = \emptyset$. Clearly, we have $Y \cap M(X, \succsim_E) = \emptyset$ since $Z \succ_P Y$. By Lemma 17, for every edge $e \in Y$ there exists an edge $e' \in X \setminus Y = Z$ such that $e' \succ_E e$. Moreover, the edges of Z cannot create a cycle by appending edges of $T \cap T'$ since $Z \cup (T \cap T') = T'$ and T' is a tree. So, for every topological sorting of the preference graph such that, as long as possible, only edges of T are taken in the tree constructed by Algorithm 1, at least one edge of Z is taken before any edge of Y. So this algorithm cannot yield T since $Z \cap T = \emptyset$. □

We introduce here a preliminary lemma[8] that will be useful in the proof of the next result:

Lemma 21 *If $\succsim$ is quasi-transitive, then:*

$$\forall A, B \subseteq \mathcal{P}(E), M(A, \succsim) \subseteq B \subseteq A \Longrightarrow M(B, \succsim) \subseteq M(A, \succsim)$$

PROOF. Let $e \in M(B, \succsim)$. By definition, $\forall b \in B, \text{not}(b \succ e)$. As $M(A, \succsim) \subseteq B$, it implies that $\forall a \in M(A, \succsim), \text{not}(a \succ e)$. Hence we have $\forall a \in A, \text{not}(a \succ e)$ $(*)$. Indeed, if $a \succ e$ for some $a \in A \setminus M(A, \succsim)$, by quasi-transitivity of $\succsim$ there exists $a' \in M(A, \succsim)$ such that $a' \succ a \succ e$, which yields a contradiction. Since $e \in M(B, \succsim) \subseteq B \subseteq A$, we get by $(*)$ that $e \in M(A, \succsim)$. □

The following result establishes the correctness of the Algorithm 3:

[8] This property is a particular instance of the so-called Aizerman axiom on choice functions [1] and, as such, directly derives from the characterization of choice functions rationalizable by a quasi-transitive preference relation [31].

Theorem 22 *For any $\succsim$-P problem from a vertex s to a vertex t, if the preference $\succsim$ is quasi-transitive and satisfies (I) then Algorithm 3 yields exactly the set $M(\mathcal{P}, \succsim)$ where $\mathcal{P}$ denotes the set of paths from s to t.*

PROOF. We have to show that $L(t) = M(\mathcal{P}, \succsim)$.

• We first prove that $M(\mathcal{P}, \succsim) \subseteq L(t)$. Consider $P \in \mathcal{P}$ such that $P \notin L(t)$. It implies:

$$\exists v_1 \in V(P) \quad \exists Q_1 \in L(v_1) \text{ such that } Q_1 \succ P(s, v_1)$$

where $V(P)$ denotes the set of vertices on P and $P(s, v_1)$ denotes the subpath from s to v_1. Note that $P(v_1, t) \cap (Q_1 \cup P(s, v_1)) = \emptyset$ by acyclicity of the graph. Hence, by (I) we have:

$$Q_1 \cup P(v_1, t) \succ P(s, v_1) \cup P(v_1, t) = P$$

Therefore $P \notin M(\mathcal{P}, \succsim)$.

• The converse inclusion is implied by Lemma 21. Indeed, $M(\mathcal{P}, \succsim) \subseteq L(t) \subseteq \mathcal{P}$ implies $M(L(t), \succsim) \subseteq M(\mathcal{P}, \succsim)$. As $M(L(t), \succsim) = L(t)$, we get $L(t) \subseteq M(\mathcal{P}, \succsim)$. □

Remark 23 *When $\succsim$ satisfies the independence axiom, the Bellman principle is verified: any subpath of a $\succsim$-efficient path is $\succsim$-efficient. Indeed, assume that a path P contains a subpath $P(v_1, v_2)$ that is non $\succsim$-efficient. Then, there exists $Q(v_1, v_2)$ such that $Q(v_1, v_2) \succ P(v_1, v_2)$ and by (I) we get $P(s, v_1) \cup Q(v_1, v_2) \cup P(v_2, t) = P(s, v_1) \cup P(v_2, t) \cup Q(v_1, v_2) \succ P(s, v_1) \cup P(v_2, t) \cup P(v_1, v_2) = P(s, v_1) \cup P(v_1, v_2) \cup P(v_2, t) = P$.*

5 Combinatorial problems based on a non-independent preference relation

Examples 6 and 7 have shown that condition (I) does not always hold. In this section, we consider specific cases where the preference relation does not satisfy (I) and we suggest some ways to overcome this difficulty using the material introduced before.

5.1 Recovering independence by approximation of preferences

Example 24 (The plainest paths problem) *We consider here the problem of determining the preferred paths on a graph $G = (V, E)$ with colored arcs for a preference relation $\succsim_C$ defined on subsets of arcs by:*

$$\forall A, B \subseteq E, \quad A \succsim_C B \iff |C(A)| \leq |C(B)|$$

where $C(A)$ denotes the set of colors represented in A.

Unfortunately, $\succsim_C$ does not verify (I) since a plain yellow path A is strictly preferred to a blue and red path B and this preference is inverted by appending a blue and red path C to A and B respectively. Indeed, the path $A \cup C$ contains three colors (Yellow,Blue,Red) whereas $B \cup C$ contains only two colors (Blue,Red).

Example 25 (The best compromise spanning trees problem) *Let us consider a* multivalued graph $G = (V, E, v)$ *where each edge $e \in E$ is valued by a cost vector $v(e) = (v_1(e), v_2(e), \ldots, v_q(e))$, with $v_j : E \to \mathbb{N}$, $j = 1, \ldots, n$ and q is the number of dimensions or criteria ($q \geq 2$).*

In multicriteria analysis, a classical preference relation is the dominance relation defined on $\mathbb{N}^q$ by:

$$\forall x, y \in \mathbb{N}^q, x \succeq y \iff \forall j \in \{1, \ldots, q\} \quad x_j \leq y_j$$

*Multiple works deal with the determination of the set of $\succeq$-efficient solutions in combinatorial problems: [37,13,6,8,41,36,29,26,19,23,11]. However, in most of these problems, we can construct instances such that all the solutions are $\succeq$-efficient, as shown in [20] for the bicriteria shortest paths problem or in [19] for the bicriteria spanning trees problem. Therefore, $\succeq$-*MOSP *(search for the $\succeq$-efficient paths in a multivalued graph) and $\succeq$-*MOST *(search for the $\succeq$-efficient trees in a multivalued graph) are intractable. Moreover, in practical applications, most of the $\succeq$-efficient solutions are irrelevant because they are ill-balanced and thus, cannot be considered as acceptable compromises.*

That's why we suggest looking for a narrower set of $\succeq$-efficient solutions limited to the best "compromise solutions". In multiobjective mathematical programming, the notion of best compromise solutions often refers to points minimizing a scalarizing function representing a distance to a given ideal $(v_1^*, v_2^*, \ldots, v_q^*)$ within the criteria space [34,39]. This function is generally based on a weighted augmented Chebychev metric, and is defined, for any solution $A \subseteq E$, by:

$$s_\lambda(A) = \max_{j \in \{1,\ldots,q\}} \{\lambda_j (v_j(A) - v_j^*)\} + \varepsilon \sum_{j=1}^{q} \lambda_j v_j(A)$$

where $v_j(A) = \sum_{a \in A} v_j(a), j = 1, \ldots, q$ are cost functions to minimize and ε is an arbitrary small and strictly positive coefficient. It induces the following preference relations between solutions:

$$A \succsim_{s_\lambda} B \iff s_\lambda(A) \leq s_\lambda(B)$$

It has been proved in [40] that $\succsim_{s_\lambda}$-MOSP is NP-hard (when looking for one $\succsim_{s_\lambda}$-efficient solution), and it could be similarly shown that $\succsim_{s_\lambda}$-MOST is NP-hard.

We would like to apply Algorithms 1 or 2 to solve $\succsim_{s_\lambda}$-MOST. Unfortunately, $\succsim_{s_\lambda}$ does not verify (I). Indeed, consider a bicriteria minimization problem where the ideal point is (0,0) and $\varepsilon = 0.01$. Let A, B, C be three feasible solutions valued as follows: $v(A) = (1, 2)$, $v(B) = (3, 1)$, $v(C) = (0, 2)$. It can be easily checked that $A \succ_{s_\lambda} B$ whereas $A \cup C \prec_{s_\lambda} B \cup C$ for $\lambda = (1, 1)$.

Thus, in the two previous examples, the violation of independence does not allow the use of Algorithms 1, 2, 3 to determine the $\succsim$-efficient solutions. Nevertheless, these algorithms might be properly used with an approximation of $\succsim$ which satisfies axiom (I). For this reason, we introduce the following definition:

Definition 26 *A preference relation $\succsim'$ is an approximation of $\succsim$ if and only if:*

$$M(\mathcal{S}, \succsim) \subseteq M(\mathcal{S}, \succsim')$$

Then, we have:

Proposition 27 *Let $\succsim$ be a preference relation and $\succsim'$ an approximation of $\succsim$ which satisfies (I), applying Algorithms 1, 2 and 3 with $\succsim'$ yields a superset of $M(\mathcal{S}, \succsim)$.*

PROOF. Thanks to independence, we know by Corollary 16, Corollary 19 and Theorem 22 that Algorithms 1, 2 and 3 yield a superset of $M(\mathcal{S}, \succsim')$, which is itself a superset of $M(\mathcal{S}, \succsim)$.□

Hence, for any preference relation $\succsim$ which does not verify (I), the set $M(\mathcal{S}, \succsim)$ can be obtained in three steps:

(1) Look for an approximation $\succsim'$ of $\succsim$ that do satisfy (I)
(2) Determine a superset S of $\succsim'$-efficient solutions using either Algorithm 1, 2 or 3
(3) Eliminate non $\succsim$-efficient solutions in S by pairwise comparisons

As a first illustration, in Example 24, a natural approximation of $\succsim_C$ is the relation $\succsim'_C$ defined by:

$$A \succsim'_C B \iff C(A) \subseteq C(B)$$

which obviously satisfies (I) and quasi-transitivity.

We consider the preferred paths problem $\succsim_C$-P on an edge-colored graph[9] with colors: Blue, Red , Yellow and Green. Applying Algorithm 3 for $\succsim_C$-P on the instance of Figure 5 (case 1) yields the two non $\succsim_C$-efficient paths:

$$\{(s,2),(2,3),(3,5),(5,6),(6,7),(7,t)\}$$
$$\{(s,2),(2,3),(3,5),(5,6),(6,8),(8,t)\}$$

However, applying the same algorithm for $\succsim'_C$-P (case 2) yields a strict superset of the set of $\succsim_C$-efficient paths (the upper path is the unique $\succsim_C$-efficient path on that graph). The two irrelevant paths will be eliminated by pairwise comparisons (step (3)).

[9] Note that this problem has been proved polynomial in [22]. Nevertheless, let us observe that the proof uses the result that linear programming problems are polynomial (since interior point algorithms are polynomial), so in this respect it can be considered as a "weakly polynomial"-proved problem.

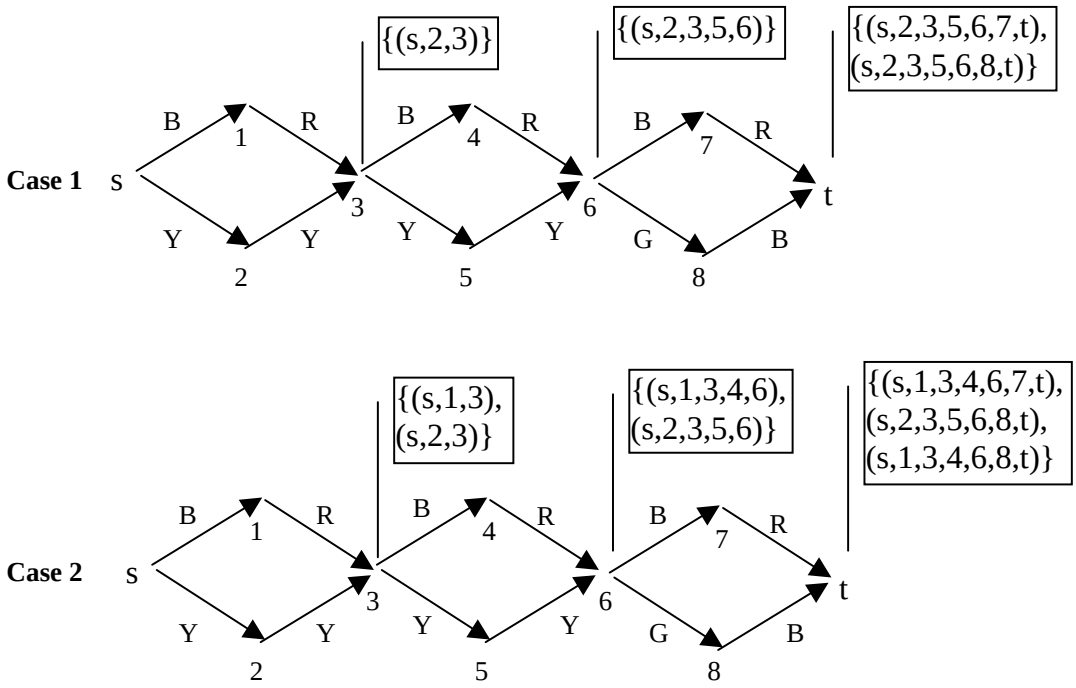

Fig. 5. An instance of the plainest path problem.

As a second illustration, let us come back to the best compromise spanning tree problem considered in Example 25. Here again the three steps procedure is useful:

(1) Approximate $\succsim_{s_\lambda}$ by the dominance relation $\succeq$ [10] which obviously satisfies (I) and quasi-transitivity.
(2) Run Algorithm 1 or Algorithm 2 with $\succeq$ so as to provide a superset of $M(\mathcal{T}, \succsim_{s_\lambda})$. For example, consider the particular instance depicted on Figure 6 and assume we want to find the best compromise solution for the scalarizing function s_λ defined for $\lambda = (1, 1)$, $v^* = (0, 0)$ and $\varepsilon = 0.01$. On this particular instance, we get a strict superset of $M(\mathcal{T}, \succsim_{s_\lambda})$. Indeed, it can easily be checked that the tree $\{a, c, d\}$ whose cost vector is $(7, 7)$ belongs to the output. This is however a mistake since $\{b, c, e\} \succ_{s_\lambda} \{a, c, d\}$ (note that $v(\{(b, c, e\}) = (6, 6)$ which strictly dominates $(7, 7)$).
(3) Perform pairwise comparisons so as to eliminate the irrelevant elements.

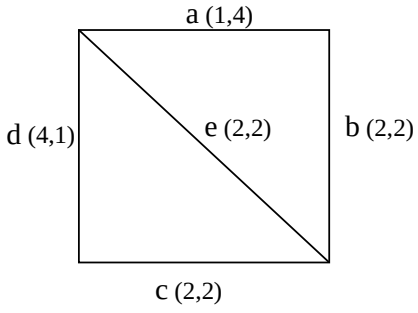

Fig. 6. An instance of best compromise spanning trees problem.

[10] Note that a solution minimizing s_λ is always $\succeq$-efficient [39].

Note that, besides these particular examples, the preference relation $\succsim_P$, as defined in Definition 11, could be used as a relevant approximation provided that $M(\mathcal{S}, \succsim) \subseteq M(\mathcal{S}, \succsim_P)$.

5.2 The shortest path with interval-valued costs

Another approach to deal with preferences that do not satisfy (I) is to establish correctness results using weaker axioms. In this respect, we introduce the following axiom:

Weak Independence (WI)

$\forall A, B, C, D \in \mathcal{P}(E)$ such that $C \cap (A \cup B) = \emptyset \quad (A \succ B) \Longrightarrow (B \cup C \succ D \Longrightarrow A \cup C \succ D)$

Note that, for quasi-transitive preferences, the Weak Independence axiom is a weakening of axiom (I), as shown by the following proposition.

Proposition 28 *For any quasi-transitive preference relation, Independence implies Weak Independence.*

PROOF. Let $A, B, C \subseteq E$ such that $C \cap (A \cup B) = \emptyset$, and let $\succsim$ be a preference relation on $\mathcal{P}(E)$ satisfying (I). Assume that:

$$A \succ B$$

$$B \cup C \succ D$$

Then $A \cup C \succ B \cup C$ by (I) and $A \cup C \succ D$ by quasi-transitivity of $\succsim$. Therefore, Weak Independence is satisfied. □

Remark 29 *The converse statement is not true: Weak Independence does not imply Independence. Indeed, let $E = \{a, b, c\}$ and consider a preference relation $\succsim$ on $\mathcal{P}(E)$ whose only strict preference is $a \succ b$. Clearly, (WI) is satisfied by default. Besides, (I) is not satisfied since $\{a\}$ is strictly preferred to $\{b\}$ and there is no preference between $\{a, c\}$ and $\{b, c\}$.*

Remark 30 *Weak Independence of $\succsim$ implies quasi-transitivity of $\succsim$. Indeed, assume that $A \succ B$ and $B \succ D$. Applying (WI) with $C = \emptyset$, we deduce that $A \succ D$.*

Using this weaker axiom, we establish the following result:

Theorem 31 *If $\succsim$ satisfies (WI) then Algorithm 3 for $\succsim$-ST yields a subset of the set of $\succsim$-efficient paths.*

PROOF. We show that $L(t) \subseteq M(\mathcal{P}, \succsim)$ by proving that $P \notin M(\mathcal{P}, \succsim) \Longrightarrow P \notin L(t)$.

Consider $P \in \mathcal{P}$ such that $P \notin M(\mathcal{P}, \succsim)$. It implies:

$$\exists Q_0 \in M(\mathcal{P}, \succsim) \text{ such that } Q_0 \succ P$$

If $Q_0 \notin L(t)$:

$$\exists v_1 \in V(Q_0) \quad \exists Q_1 \in L(v_1) \text{ such that } Q_1 \succ Q_0(s, v_1) \text{ with } v_1 \neq t$$

If $Q_1 \cup Q_0(v_1, t) \notin L(t)$, then:

$$\exists v_2 \in V(Q_0(v_1, t)) \quad \exists Q_2 \in L(v_2) \text{ such that } Q_2 \succ Q_1 \cup Q_0(v_1, v_2) \text{ with } v_2 \neq v_1$$

Since Q_0 is finite, by iterating we will find $v_i \in V(Q_0(v_{i-1}, t))$ and $Q_i \in L(v_i)$ such that:

$$Q_i \succ Q_{i-1} \cup Q_0(v_{i-1}, v_i) \text{ and } Q_i \cup Q_0(v_i, t) \in L(t)$$

with $v_i \notin \{v_1, \ldots, v_{i-1}\}$ (possibly $v_i = t$, which guarantees that $Q_i \cup Q_0(v_i, t) = Q_i \in L(t)$)

Let show by induction that:

$$\forall j \in \{1, \ldots, i\} \quad Q_j \cup Q_0(v_j, t) \succ P$$

• It is true for $j = 1$:
By construction,

$$Q_1 \succ Q_0(s, v_1) \tag{4}$$

and

$$Q_0 = Q_0(s, v_1) \cup Q_0(v_1, t) \succ P \tag{5}$$

From Equations 4 and 5, we deduce that $Q_1 \cup Q_0(v_1, t) \succ P$ by (WI).

• If it is true for $j = k < i$, then it is true for $j = k + 1$:
Assume that $Q_k \cup Q_0(v_k, t) \succ P$. Since $Q_0(v_k, t) = Q_0(v_k, v_{k+1}) \cup Q_0(v_{k+1}, t)$, we have

$$Q_k \cup Q_0(v_k, v_{k+1}) \cup Q_0(v_{k+1}, t) \succ P \tag{6}$$

Moreover, by construction:

$$Q_{k+1} \succ Q_k \cup Q_0(v_k, v_{k+1}) \tag{7}$$

Hence, we derive $Q_{k+1} \cup Q_0(v_{k+1}, t) \succ P$ from Equations 7 and 6 by (WI).

Thus, for $j = i$, we have $Q_i \cup Q_0(v_i, t) \succ P$ and $Q_i \cup Q_0(v_i, t) \in L(t)$. Therefore $P \notin L(t)$ and the result follows. □

To illustrate the interest of such a result, consider the graph of Figure 7, with interval-valued arcs. We are looking for determining the set of preferred paths from s to t. We use the preference

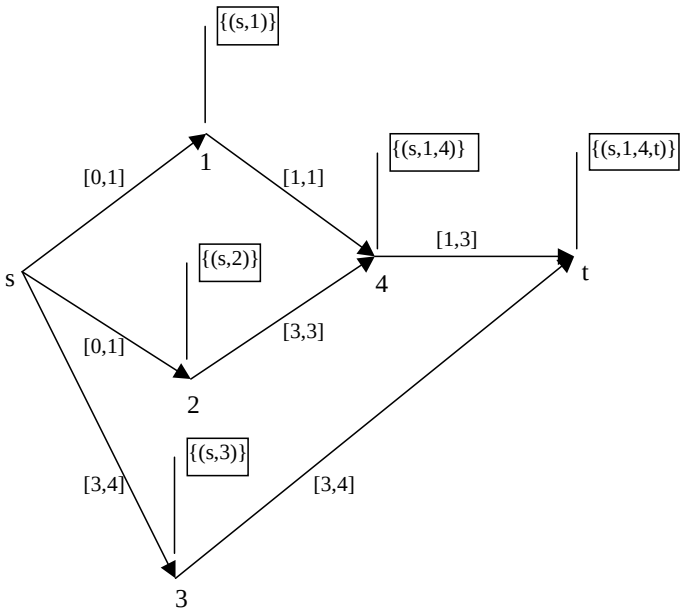

Fig. 7. An interval-valued graph.

relation $\succsim_{[\,]}$ on intervals, which is defined as follows: For any $A, B \subseteq E$ such that $v(A) = [l(A), r(A)]$ and $v(B) = [l(B), r(B)]$,

$$A \succ_{[\,]} B \iff r(A) < l(B)$$
$$A \sim_{[\,]} B \iff l(B) \leq r(A) \text{ and } r(B) \geq l(A)$$

When we define $v(A \cup B)$ as $[l(A)+l(B), r(A)+r(B)]$, we remark first that $\succsim_{[\,]}$ does not verify (I). Indeed, consider three sets A, B, C ($(A \cup B) \cap C = \emptyset$) such that $v(A) = [3, 4]$, $v(B) = [5, 6]$ and $v(C) = [1, 2]$. We have $A \succ_{[\,]} B$ whereas $A \cup C \sim_{[\,]} B \cup C$. Consequently, we can not use Theorem 22 to guarantee the correctness of Algorithm 3 on such a problem. However, $\succsim_{[\,]}$ verifies (WI). Indeed, suppose $A \succ_{[\,]} B$ and $B \cup C \succ_{[\,]} D$:

$$A \succ_{[\,]} B \iff r(A) < l(B)$$
$$B \cup C \succ_{[\,]} D \iff r(B \cup C) = r(B) + r(C) < l(D)$$

Now, $r(A \cup C) = r(A) + r(C) < l(B) + r(C) \leq r(B) + r(C) < l(D)$. Therefore $A \cup C \succ_{[\,]} D$, which shows that $\succsim_{[\,]}$ satisfies (WI). By Theorem 31, we deduce that Algorithm 3 yields a subset of $M(\mathcal{P}, \succsim_{[\,]})$. On the graph of Figure 7, it yields the path {(s,1),(1,4),(4,t)}. This path is indeed $\succsim_{[\,]}$-efficient, but we miss the path {(s,2),(2,4),(4,t)} which is also $\succsim_{[\,]}$-efficient. Here, we have so obtained a strict subset of $M(\mathcal{P}, \succsim_{[\,]})$. Note that the path {(s,3)(3,t)} has been soundly eliminated. However, applying Algorithm 3 with a preference relation that does not satisfy (I) nor (WI) might lead to irrelevant results. As an illustration, let us come back to the plainest path problem. Note that the preference relation $\succsim_C$ doesn't satisfy (I) nor (WI). Applying Algorithm 3 on the graph of Figure 5 (case 1) yields only $\succsim_C$-dominated paths, whereas there exists a unique $\succsim_C$-efficient path. Such a situation cannot happen with a preference relation satisfying (WI).

6 Conclusion

In this paper, we have presented a preference-based framework for combinatorial problems. This approach is general in the sense that it could be used to solve a lot of variations of usual combinatorial problems. The main drawback of our approach is a possibly high computational demand. However, this difficulty could be partially overcome by making cuts in the search tree (using a depth-first search and adequate bounds). In order to reduce the size of the search tree, it could be interesting to restrict the search to *a complete set of alternatives*, i.e. a set of solutions such that any $\succsim$-efficient solution is represented in that set by an equivalent solution. Last but not least, other axioms on preferences and other combinatorial problems may be explored to get new theoretical results.

Acknowledgement. We wish to thank Jérôme Monnot for helpful comments on an earlier version of the paper.

References

[1] M.A. Aizerman and A.V. Malishevski. General theory of best variants choice. *IEEE Trans. Automat. Control*, pages 1031–1041, 1981.

[2] U. Bossong and D. Schweigert. Minimal paths on ordered graphs. Report in Wirtschaftsmathematik no. 24/1997, University of Kaiserslautern, 1997.

[3] U. Bossong and D. Schweigert. Minimal trees on ordered graphs. Working Paper, 1998.

[4] R. E. Burkard, R. A. Cuninghame-Green, and U. Zimmermann. *Algebraic and Combinatorial Methods in Operations Research*. Number 19 in Annals of Discrete Mathematics. North-Holland, Amsterdam, 1984.

[5] A. Cayley. A theorem on trees. *Quaterly Journal of Mathematics*, 23:376–378, 1889.

[6] H. W. Corley. Efficient spanning trees. *Journal of Optimization Theory and Applications*, 45(3):481–485, 1985.

[7] F. Della Croce, V. Th. Paschos, and A. Tsoukias. An improved general procedure for lexicographic bottleneck problems. *Operations Research Letters*, 24:187–194, 1999.

[8] J. Current and M. Marsh. Multiobjective transportation network design: Taxonomy and annotation. *European Journal of Operational Research*, 65:4–19, 1993.

[9] P. Dasgupta, P.P. Chakrabarti, and S.C. DeSarkar. Searching game trees under a partial order. *Artificial Intelligence*, 82:237–257, 1996.

[10] B. De Finetti. *Probability Theory Vol. I.* Wiley, London, 1974.

[11] L.C. Dias and J.N. Clímaco. Shortest path problems with partial information: Models and algorithms for detecting dominance. *European Journal of Operational Research*, 121(1):16–31, 2000.

[12] M. Ehrgott. *Multicriteria optimization.* Number 491 in Lecture Notes in Economics and Mathematical Systems. Springer, Berlin, 2000.

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

[14] C. Flament and B. Leclerc. Arbres minimaux d'un graphe préordonné. *Discrete Mathematics*, 46:159–171, 1983.

[15] M. Futtersack and P. Perny. BCA*: une généralisation d'A* pour la recherche de solutions de compromis dans des problèmes d'optimisation multi-objectifs. In *RFIA*, 2000.

[16] V. Gabrel and D. Vanderpoooten. Generation and selection of efficient paths in a multiple criteria graph: the case of daily planning the shots taken by a satellite with an interactive procedure. Cahier du Lamsade no. 136, Université Paris Dauphine, 1996.

[17] M.R. Garey and D.S. Johnson. *Computers and intractability.* W.H. Freeman and company, 1979.

[18] M. Gondran and M. Minoux. *Graphes et algorithmes.* Collection de la Direction des Études et Recherches d'Électricité de France. Eyrolles, 1995.

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

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

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

[22] X. Li, S. Zhang, and H. J. Broersma. Directed paths with few or many colors in colored directed graphs. Memorandum no. 1543, University of Twente, 2000.

[23] E. Q.V. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16:236–245, 1984.

[24] L.G. Mitten. Composition principles for the synthesis of optimal multi-stage processes. *Operations Research*, 12, 1964.

[25] T.L. Morin. Monotonicity and the principle of optimality. *Journal of Mathematical Analysis and Applications*, 86:665–674, 1982.

[26] J. Mote, I. Murthy, and D.L. Olson. A parametric approach to solving bicriterion shortest path problems. *European Journal of Operational Research*, 53:81–92, 1991.

[27] M. Müller. Partial order bouding: a new approach to evaluation in game tree search. *Artificial Intelligence*, 129:279–231, 2001.

[28] M. Pirlot and Ph. Vincke. *Semiorders: properties, representations, applications.* Kluwer Academic Publishers, 1997.

[29] R.M. Ramos, S. Alonso, J. Sicilia, and C. Gonzalez. The problem of the optimal biobjective spanning tree. *European Journal of Operational Research*, 111:617–628, 1998.

[30] G. Rote. Path problems in graphs. In G. Tinhofer, E. Mayr, H. Noltemeier, and M. M. Syslo, editors, *Computational Graphs Theory*, volume 7 of *Computing Supplementum*, 1990.

[31] T. Schwartz. Choice functions, "rationality" conditions, and variations on the weak axiom of revealed preference. *Journal of Economic Theory*, 13:414–427, 1976.

[32] D. Schweigert. Ordered graphs and minimal spanning trees. *Foundations of Computing and Decision Sciences*, 24(4):219–229, 1999.

[33] P. Serafini. Some considerations about computational complexity for multiobjective combinatorial problems. In J. Jahn and W. Krabs, editors, *Recent advances and historical development of vector optimization*, volume 294 of *Lecture Notes in Economics and Mathematical Systems*, Berlin, 1986. Springer-Verlag.

[34] R.E. Steuer. *Multiple criteria optimization: theory, computation and application.* John Wiley and Sons, New York, 1986.

[35] B. S. Stewart and C. C. White III. Multiobjective A*. *Journal of the Association for Computing Machinery*, 38(4):775–814, 1991.

[36] C. Tung Tung and K.L. Chew. A multicriteria pareto-optimal path algorithm. *European Journal of Operational Research*, 62:203–209, 1992.

[37] E. L. Ulungu and J. Teghem. Multi-objective combinatorial optimization: a survey. *Journal of Multicriteria Decision Analysis*, 3:83–104, 1994.

[38] C. C. White, B. S. Stewart, and R. L. Carraway. Multiobjective, preference-based search in acyclic OR-graphs. *European Journal of Operational Research*, 56:357–363, 1992.

[39] A.P. Wierzbicki. On the completeness and constructiveness of parametric characterizations to vector optimization problems. *OR Spektrum*, 8:73–87, 1986.

[40] G. Yu and G. Yang. On the robust shortest path problem. *Computers and Operations Research*, 25(6):457–468, 1998.

[41] G. Zhou and M. Gen. Genetic algorithm approach on multi-criteria minimum spanning tree problem. *European Journal of Operational Research*, 114:141–152, 1999.

[42] U. Zimmermann. *Linear and Combinatorial Optimization in Ordered Algebraic Structures.* Number 10 in Annals of Discrete Mathematics. North-Holland, Amsterdam, 1981.