# Bidirectional Preference-based Search for Multiobjective State Space Graph Problems

Lucie Galand, Anisse Ismaili, Patrice Perny, Olivier Spanjaard

# Bidirectional Preference-Based Search for Multiobjective State Space Graph Problems

**Lucie Galand**[1] and **Anisse Ismaili**[2] and **Patrice Perny**[2] and **Olivier Spanjaard**[2]
[1]University Paris Dauphine, France
*lucie.galand@dauphine.fr*
[2]University Pierre et Marie Curie, France
{*anisse.ismaili, patrice.perny, olivier.spanjaard*}*@lip6.fr*

## Abstract

In multiobjective state space graph problems, each solution-path is evaluated by a cost vector. These cost vectors can be partially or completely ordered using a preference relation compatible with Pareto dominance. In this context, multiobjective preference-based search (MOPBS) aims at computing the preferred feasible solutions according to a predefined preference model, these preferred solutions being a subset (possibly the entire set) of Pareto optima. Standard algorithms for MOPBS perform a unidirectional search developing the search tree forward from the initial state to a goal state. Instead, in this paper, we focus on bidirectional search algorithms developing simultaneously one forward and one backward search tree. Although bi-directional search has been tested in various single objective problems, its efficiency in a multiobjective setting has never been studied. In this paper, we present several implementations of bidirectional preference-based search convenient for the multiobjective case and investigate their efficiency.

## Introduction

Decision making in complex environments often requires taking into account different point of views in the analysis of preferences, thus leading to the definition of several objectives that must be optimized simultaneously. This is the case for instance in path-planning problems where we want to minimize distance, travel time and cost, and find the best tradeoffs between these objectives. This is also the case when the costs of paths depend on different possible scenarios, or different discordant sources of information. Such examples and many others are a permanent incentive to study multiobjective extensions of general problem solving methods used in AI, either for automated decision making or for human decision support.

Heuristic search in state space graphs is one of those domains where considering multiple objectives is natural. Although the standard problem involves a single objective and consists in finding one path with minimum cost among all solution paths from a given source node to a goal node, the multiobjective version consists in determining the set of Pareto-optimal tradeoffs corresponding to those solution paths. Recall that a solution is Pareto-optimal whenever one cannot diminish one component of its cost vector without increasing another one. Although standard heuristic search algorithms such as $A^*$ (Hart, Nilsson, and Raphael 1968) were initially introduced in the framework of single objective optimization, various extensions to deal with vector-valued state space graphs have been proposed to determine the Pareto set (see the $MOA^*$ algorithm and its variants in (Stewart and White III 1991; White, Stewart, and Carraway 1992; Mandow and Pérez de la Cruz 2005)).

As Pareto dominance leaves many pairs of solutions incomparable, the Pareto set is often very large. For example, one can find in (Hansen 1980) families of instances of biobjective shortest path problems for which the set of Pareto-optimal feasible tradeoffs is exponential in the number of vertices of the graph. Fortunately, there is generally no need to consider explicitly all possible tradeoffs within the Pareto set, because many of them do not fit to the type of compromise sought. When richer information about preferences is available, one can use a preference model to guide the search and determine the preferred tradeoffs without a prior generation of the Pareto set. A typical example of such a multiobjective preference-based search (MOPBS) is given in (Carraway, Morin, and Moskowitz 1990; Dasgupta, Chakrabarti, and DeSarkar 1995) where the search is directed by a multiattribute utility function. Other examples can be found in (Perny and Spanjaard 2002; Galand and Perny 2006; Galand and Spanjaard 2007) with various preference models leading to various algorithmic contributions. Following this line, in addition to Pareto dominance, we consider here finer models such as Lorenz dominance and ordered weighted averages to focus the search on those Pareto-optimal tradeoffs achieving a fair balance between the objectives.

Beside the choice of the preference model used to direct the search, other issues are worth investigating in MOPBS, in particular the strategy used to develop the search. Indeed, until now, MOA$^*$ and all its variants proposed for MOPBS perform a uni-directional search developing the search tree forward from the initial state to the goal state. However, in the case of single-objective optimization, an interesting alternative strategy named bi-directional search has been proposed and investigated by several authors, see e.g., (Pohl 1971; Kwa 1989; Kaindl and Kainz 1997;

Felner et al. 2010). It is essentially a variant of $A^*$ developing simultaneously one forward and one backward search tree, thus leading to two search trees of approximately half height, potentially expanding exponentially less labels. Several implementations of the bi-directional search scheme have been proposed and tested and it is empirically established that the bi-directional search is more efficient than uni-directional search in a number of cases (see (Kaindl and Kainz 1997) for a detailed analysis of the respective advantages of the two approaches).

However, quite surprisingly, the potential of bi-directional search in a multiobjective setting has never been studied. This is precisely the aim of this paper. The paper is organized as follows: in the next section we introduce preference models used in the paper to compare cost vectors associated to paths. Then we introduce the necessary background on uni-directional multiobjective search and the main problems to overcome to design an admissible multiobjective bi-directional search algorithm. Then we propose different implementations of bi-directional search and provide numerical tests to assess the efficiency of this approach for various preference models.

## Preference models for MO problems

In multiobjective state space search, any admissible transition from a state to another induces a cost vector in $\mathbb{Z}_+^p$. Since any path in the state space graph represents a sequence of such admissible transitions, a cost vector can be associated to each path by componentwise summation of transition costs. Hence, any preference model defined on cost vectors in $\mathbb{Z}_+^p$ induces a preference over paths. Let us introduce now the preference models used in the paper.

### Pareto Dominance

We recall first some classical dominance notions used in multiobjective optimization. The Weak-Pareto dominance relation (WP-dominance for short) on cost vectors of $\mathbb{Z}_+^p$ is defined, for all $x, y \in \mathbb{Z}_+^p$ by:

$$x \succsim_P y \iff [\forall i \in \{1, \ldots, p\}, x_i \leq y_i)]$$

The Pareto dominance relation (P-dominance for short) is defined on $\mathbb{Z}_+^p$ as the asymmetric part of $\succsim_P$:

$$x \succ_P y \iff [x \succsim_P y \text{ and not}(y \succsim_P x)]$$

Within a set $X$ any $x$ is said to be *P-dominated* when $y \succ_P x$ for some $y$ in $X$, and *P-optimal* when there is no $y$ in $X$ such that $y \succ_P x$. The set of P-optimal elements in $X$ is denoted $\mathcal{M}(X, \succsim_P)$. As Pareto dominance is a partial relation, many Pareto-optimal solution paths may exist. In fact, the number of Pareto-optimal vectors corresponding to solution paths can grow exponentially with the number of states as shown in (Hansen 1980). Its complete determination may induce prohibitive computation times. Moreover, such a complete enumeration is generally not necessary because some preference information may be available concerning the type of tradeoff sought in the Pareto-set. We introduce now models favoring the selection of well-balanced solutions.

### Lorenz Dominance

To obtain well balanced solutions, one may be interested in finding a path that "distributes" costs over components. This idea is expressed by the *transfer principle*, widely used in Mathematical Economics and Social Choice theory for the measurement of inequalities (Marshall and Olkin 1979; Shorrocks 1983): Let $x \in \mathbb{Z}_+^p$ such that $x_i < x_j$ for some $i, j$. Then for all $\varepsilon > 0$ such that $\varepsilon \leq (x_j - x_i)$, then any vector of the form $x + \varepsilon e_i - \varepsilon e_j$ is preferred to $x$, where $e_k$ is the vector whose $k^{th}$ component equals 1, all others being null. When combined with P-monotonicity (i.e., compatibility of preferences with P-dominance), it becomes more powerful. In order to characterize the preferences that are compatible with P-dominance and transfers, we recall the definition of generalized Lorenz vectors and generalized Lorenz dominance (for more details see e.g. (Shorrocks 1983)).

The *generalized Lorenz Vector* associated to $x \in \mathbb{Z}_+^p$ is:

$$L(x) = (x_{(1)}, x_{(1)} + x_{(2)}, \ldots, x_{(1)} + x_{(2)} + \ldots + x_{(p)})$$

where $x_{(1)} \geq x_{(2)} \geq \ldots \geq x_{(p)}$ are the components of $x$ sorted by decreasing order. The *generalized Lorenz dominance* relation (L-dominance for short) on $\mathbb{Z}_+^p$ is defined by:

$$\forall x, y \in \mathbb{Z}_+^p,\ x \succsim_L y \iff L(x) \succsim_P L(y)$$

Its asymmetric part $x \succ_L y$ is: $L(x) \succ_P L(y)$. Within a set $X$, any element $x$ is said to be *L-dominated* when $y \succ_L x$ for some $y$ in $X$, and *L-optimal* when there is no $y$ in $X$ such that $y \succ_L x$. The preferences characterized by P-monotonocity and the transfer principle are linked to L-dominance by a result of Chong (1976):

**Theorem 1** *Lorenz dominance is the minimal transitive relation (with respect to inclusion) satisfying compatibility with P-dominance and the transfer principle.*

As a consequence, the set of L-optimal vectors is included in the set of P-optimal vectors. The subset of L-optimal cost vectors appears as a very natural solution concept in well-balanced multi-objective optimization problems.

### Ordered Weighted Averages

Ordered Weighted Averages (OWA for short) are defined, for any cost vector $x = (x_1, \ldots, x_p)$ and any weighting vector $w = (w_1, \ldots, w_p)$ such that $\Sigma_{i=1}^p w_i = 1$, by: $\psi(x) = \sum_{i=1}^p w_i x_{(i)}$. Quantity $\psi(x)$ represents an overall cost that must be minimized. The associate preference is:

$$x \succsim_\psi y \iff \psi(x) \leq \psi(y)$$

It can be used for the measurement of inequalities provided weights satisfy the following inequalities: $w_1 > w_2 > \ldots > w_p > 0$. By choosing a decreasing sequence of weights, one puts indeed more weight on the least satisfied objective, then on the second least satisfied objective and so on... This is an extension of minmax operator, which allows for more compensation among the objectives. More formally, the idea of choosing decreasing weights to preserve equity directly derives from the following relation linking OWA operators and Lorenz vectors:

$$\psi(x) = \sum_{j=1}^p (w_j - w_{j+1}) L_j(x)$$

where $w_{p+1} = 0$. Hence, if $x \succ_L y$ then $\psi(x) > \psi(y)$. As a consequence, any cost vector $x$ minimizing $\psi$ over the feasible set $X$ belongs to the set of L-optimal vectors. Moreover, any such $x$ is Pareto-optimal.

We will see later in the paper how preference relations $\succsim_P, \succsim_L$ and $\succsim_\psi$ introduced in this section can be used to focus the search on the preferred solutions in multiobjective state space search problems.

## Preference-based Search for MO problems

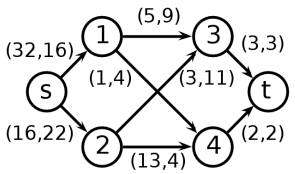
In this section, we write a generalized framework for existing monodirectional preference-based search algorithms. These algorithms will be generalized to bidirectional search in a further section.

Let $G = (N, A)$ denote a state space graph where $N$ is a finite set of nodes (possible states), and $A$ is a set of arcs representing transitions. Set $A$ is defined by $A = \{(n, n') : n \in N, n' \in S(n)\}$ where $S(n) \subseteq N$ is the set of all successors of node $n$. A cost vector $c(n, n')$ is attached to each arc (transition) $(n, n') \in A$. A *path* $P$ is a sequence of $x$ nodes $\langle n_1, n_2, \ldots, n_x \rangle$ such that for all $i < x$, $(n_i, n_{i+1})$ is in $A$. Note that we can indifferently define path $P$ as a sequence of arcs $\langle (n_1, n_2), (n_2, n_3), \ldots, (n_{x-1}, n_x) \rangle$. In the following, we denote by $s \in N$ the initial node in $G$ and $t \in N$ the goal node in $G$, and a *solution path* is a path from $s$ to $t$ (i.e. $n_1 = s$ and $n_x = t$). The cost vector of a path $P$ is defined by $c(P) = \sum_{(n,n') \in P} c(n, n')$. Let $\mathcal{P}(n, n')$ be the set of all paths from node $n \in N$ to node $n' \in N$. We say that a path $P$ is $\succsim$-optimal if there is no path $P'$ such that $P' \succ P$. We denote by $\mathcal{M}(\mathcal{P}(n, n'), \succsim)$ the set of $\succsim$-optimal paths from $n$ to $n'$. The Multiobjective Preference-Based Search (MOPBS) problem can be stated as follows.

**MOPBS PROBLEM.** Given a preference relation $\succsim$, a state space graph $G(N, A)$, and two nodes $s \in N$ and $t \in N$, one wants to determine one path per equivalence class in the quotient set $\mathcal{M}(\mathcal{P}(s, t), \succsim)/\sim$.

This latter set is called a *complete set of optimal paths* in the following. Furthermore, we assume that the set $\mathcal{M}(\mathcal{P}(s, t), \succsim)$ is not empty. Note that even if the number of states is polynomially bounded, the MOPBS problem is NP-hard in the general case, for $\succsim \in \{\succsim_P, \succsim_L, \succsim_\psi\}$ (e.g. (Perny and Spanjaard 2003)).

**Example.** In Figure 1, the $\succsim$-optimal solutions for $\succsim \in \{\succsim_P, \succsim_L, \succsim_\psi\}$ ($w = (3, 1)$ in $\succsim_\psi$) are indicated by stars.




| solution | P-vector | L-vector | $\psi$ |
|---|---|---|---|
| $\langle s, 1, 3, t \rangle$ | $(40, 28)$ | $(40, 68)$ | 37.00 |
| $\langle s, 1, 4, t \rangle$ | $(35, 22)^*$ | $(35, 57)^*$ | 31.75 |
| $\langle s, 2, 3, t \rangle$ | $(22, 36)^*$ | $(36, 58)$ | 32.50 |
| $\langle s, 2, 4, t \rangle$ | $(31, 28)^*$ | $(31, 59)^*$ | $30.25^*$ |

Figure 1: An MOPBS instance.

**Algorithm PBMOA$^*$.** We now present algorithm PBMOA$^*$ ("PB" stands for "Preference-based") which determines a complete set of optimal paths for any P-monotonic preference relation $\succsim$. This algorithm summarizes several contributions on multiobjective heuristic search (Stewart and White III 1991; Mandow and Pérez de la Cruz 2005; Galand and Perny 2006; Machuca et al. 2010) and on preference-based search (White, Stewart, and Carraway 1992; Perny and Spanjaard 2002). In algorithm PBMOA$^*$, to any path $P$ from $s$ is associated a label $\ell = [n^\ell, g^\ell, P]$ where $n^\ell$ is the terminal node of $P$ and $g^\ell$ is the cost vector $c(P)$. Note that we denote by $P^\ell$ the path associated to label $\ell$, and we denote by $\langle P, n \rangle$ the path obtained from path $P$ (from $s$ to $n'$) by adding arc $(n', n)$ to $P$. In a multiobjective setting, several paths from $s$ to a node $n$ can be Pareto optimal (which is a necessary condition for a path to lead to an optimal solution path with respect to any P-monotonic preference relation). Thus at any node $n$ is attached a set of labels $\mathcal{L}(n)$ that corresponds to a set of current best known paths from $s$ to $n$.The set of labels $\mathcal{L}(n)$ is divided into two disjoint subsets: the set $\mathcal{T}(n)$ of *temporary* labels (not yet expanded) and the set $\mathcal{P}(n)$ of *permanent* labels (already expanded). The temporary labels corresponds to temporarily-optimal paths. These labels have then to be expanded. The set $\mathcal{O}$ (resp. $\mathcal{C}$) is the set of *open* (resp. *closed*) labels over all the nodes during the search, it is defined by $\mathcal{O} = \bigcup_{n \in N} \mathcal{T}(n)$ (resp. $\mathcal{C} = \bigcup_{n \in N} \mathcal{P}(n)$). Finally, the set $M$ contains the currently $\succsim$-optimal solution-paths that have been found and may be improved.

In order to limit the expansion of paths leading to non-optimal solution paths, two discarding rules are used in PBMOA$^*$: a *global discarding rule* and a *local discarding rule*. The global discarding rule checks whether all solution paths one can obtain from a detected path $P$ are not better than an already detected solution path. As in the scalar case, this test can be strenghtened by using a heuristic evaluation of the costs of optimal solution paths one can obtain from a path. The heuristic evaluation is given by a set of cost vectors $H(n)$ on a node $n$ which estimate the cost of any optimal path from $n$ to $t$ with respect to $\succsim$. A multiobjective heuristic $H(n)$ on a node $n$ is said to be optimistic with respect to $\succsim$ if for all paths $P$ in $\mathcal{M}(\mathcal{P}(n, t), \succsim)$, there exists a cost vector $h \in H(n)$ such that $h \succsim c(P)$. The algorithm PBMOA$^*$ is admissible when heuristic $H$ is optimistic with respect to $\succsim_P$. For any label $\ell$, the global discarding rule is:

**GLOBAL-DIS**$_\succsim(M, \ell)$: returns true if for all $h \in H(n^\ell)$, there exists $\ell' \in M$ such that $g^{\ell'} \succsim g^\ell + h$; otherwise false.

The local discarding rule checks whether a path $P$ from $s$ to $n$ is dominated by another already detected path from $s$ to $n$. As in the scalar case, in the Pareto dominance case a label $\ell$ can be discarded if there exists another label $\ell' \in \mathcal{L}(n^\ell)$ for which $P^{\ell'} \succsim_P P^\ell$ (by principle of optimality). The validity of this principle ("Bellman's principle") indeed follows from the monotonicity assumption (Mitten 1964), that is, for any paths $P, P'$ at node $n$, $P \succsim P' \Rightarrow \langle P, n' \rangle \succsim \langle P', n' \rangle$ for all successors $n'$ of $n$. However, depending on the preference model used,

this assumption does not necessarily hold. In particular, it does not hold for $\succsim_L$ and $\succsim_\psi$ (Perny and Spanjaard 2003; Galand and Spanjaard 2007). Nevertheless, one can resort to a *local preference relation* $\succsim'$ such that 1) the monotonicity assumption holds for $\succsim'$, and 2) $P \succsim' P' \Rightarrow P \succsim P'$ (weak principle of optimality (Carraway, Morin, and Moskowitz 1990)). For example, for any P-monotonic preference relation (this is the case of $\succsim_L$ and $\succsim_\psi$), one can set $\succsim'$=$\succsim_P$. Note that more refined local preference relation can be used for $\succsim_\psi$ (Galand, Lesca, and Perny 2013). For any label $\ell$ the local discarding rule is then defined by:

LOCAL-DIS$_{\succsim'}(\mathcal{L}(n^\ell), \ell)$: returns true if there exists $\ell' \in \mathcal{L}(n^\ell)$ such that $g^{\ell'} \succsim' g^\ell$; and false otherwise.

**Pseudo-code.** The pseudo-code of algorithm PBMOA$^*$ is given in Algorithm 1. Furthermore, in all what follows, we set $H(n) = \{(h_1^I(n,t), \ldots, h_p^I(n,t))\}$, where $h_i^I(n,t)$ is the value of an optimal single objective shortest path from $n$ to $t$ according to objective $i$. This so-called ideal-point heuristic is obviously optimistic, and easy to compute (in most cases). We assume that the data structures for the sets $\{\mathcal{L}(n) = \mathcal{T}(n) \cup \mathcal{P}(n)\}_{n\in N}$, $\mathcal{O} = \bigcup_{n\in N} \mathcal{T}(n)$, $\mathcal{C} = \bigcup_{n\in N} \mathcal{P}(n)$ and $M$ are created when needed. A major distinction with the scalar version is that one cannot detect the termination of the algorithm by closing the goal node. Finally, the operation **update** consists in inserting the new label $\ell'$ in $\mathcal{T}(n')$ (and $\mathcal{O}$) while discarding any label $\ell$ in $\mathcal{T}(n')$ (and $\mathcal{O}$) such that $g^\ell$ is dominated by $g^{\ell'}$ w.r.t. the local preference relation $\succsim'$. In the same way, it inserts $\ell'$ in $M$ but with respect to $\succsim$.

**Implementation.** In this work, we apply PBMOA$^*$for $\succsim \in \{\succsim_P, \succsim_L, \succsim_\psi\}$ as witness experiments. Whatever the definition of $\succsim$, we set $\succsim'$=$\succsim_P$ as the local preference relation. A way to trigger the termination of the algorithm is to empty set $\mathcal{O}$. However, due to the potentially high cardinality of $\mathcal{O}$, this condition is not satisfactory from the computational viewpoint. Another –more efficient– way to trigger the termination of the algorithm is to compare a *lower bound* (describing optimistically what could be developped further from $\mathcal{O}$) to an *upper bound* (describing what is already $\succsim$-dominated by the solution paths found). For $\succsim$=$\succsim_P$, this termination condition amounts to comparing *bounding sets*, which is highly computation times consuming as soon as we have more than two objectives (Sourd and Spanjaard 2008; Przybylski, Gandibleux, and Ehrgott 2010). However for $\succsim$=$\succsim_\psi$ and $\succsim$=$\succsim_L$, one can resort to easily computable *scalar linear bounds*.

Given a label $\ell$ and its solution-path $P^\ell$, the upper bound $\mathbf{UB}_{\succsim}(\ell)$ is defined for $\succsim_L$ and $\succsim_\psi$ as follows: $\mathbf{UB}_{\succsim_\psi}(\ell) = p \cdot \psi(g^\ell)$ and $\mathbf{UB}_{\succsim_L}(\ell) = p \cdot \max_i g_i^\ell$, where $p$ is the number of objectives. Given a set of already detected solution paths $M$, we define $\mathbf{UB}_{\succsim}(M)$ as $\min_{\ell\in M}\{\mathbf{UB}_{\succsim}(\ell)\}$. It upper bounds $\sum_i c_i(P)$ for any further solution path $P$. For $\succsim_L$ and labels $\ell, \ell'$, one can prove that $\mathbf{UB}_{\succsim_L}(\ell) \leq \sum_i c_i(P^{\ell'})$ implies $\ell \succsim_L \ell'$.

For any label $\ell$, a scalar linear lower bound on $\Sigma_i c_i(P)$ for any extension $P$ of $P^\ell$ to the goal node is: $\mathbf{LB}_{\Sigma f}(\ell) = \sum_i g_i^\ell + h_i^I(n^\ell)$ (where $x_i$ denotes the $i^{th}$ component of vector $x$). We define $\min\{\mathbf{LB}_{\Sigma f}(\ell) : \ell \in \mathcal{O}\}$ as $\mathbf{LB}_{\Sigma f}(\mathcal{O})$. By the use of an appropriate heap structure, a label $\ell_{top}$ such that $\mathbf{LB}_{\Sigma f}(\ell_{top}) = \mathbf{LB}_{\Sigma f}(\mathcal{O})$ can be accessed in constant time through $\text{top}_{\Sigma f}(\mathcal{O})$. By monotonicity of $\sum_i h_i^I$ (in the single objective heuristic standard sense), $\mathbf{LB}_{\Sigma f}(\mathcal{O})$ is a lower bound over $\Sigma_i c_i(P)$ for any solution path $P$ that could be discovered later. For $\succsim_\psi$, one can prove that when $w_1 > w_2 > \ldots > w_p > 0$ and $\sum_i w_i = 1$, we have $\sum_i x_i \leq p\psi_w(x)$.

Consequently, for $\succsim$ in $\{\succsim_\psi, \succsim_L\}$, the algorithm can be safely terminated when $\mathbf{UB}_{\succsim}(M) < \mathbf{LB}_{\Sigma f}(\mathcal{O})$.

**Algorithm 1**: MOPBS: unidirectional PBMOA$^*$

**Input**: State Space Graph $G$, starting node $s$, terminal node $t$, global preference relation $\succsim$, local preference relation $\succsim'$

1. INITIALIZATION
**insert** label $[s, \vec{0}, \langle s\rangle]$ into $\mathcal{T}(s)$ and $\mathcal{O}$
2. CHECK TERMINATION
**if** $\mathcal{O} = \emptyset$ or $\mathbf{LB}_{\Sigma f}(\mathcal{O}) > \mathbf{UB}_{\succsim}(M)$ **then**
  **return** $M$
3. LABEL EXPANSION
$\ell \leftarrow \text{top}_{\Sigma f}(\mathcal{O})$
**for** *each node* $n' \in S(n^\ell)$ **do**
  **create** $\ell' = [n', g^\ell + c(n^\ell, n'), \langle P^\ell, n'\rangle]$
  **if** LOCAL-DIS$_{\succsim'}(\mathcal{L}(n^{\ell'}), \ell')$ **then discard** $\ell'$**else**
  **if** GLOBAL-DIS$_{\succsim}(M, \ell')$ **then discard** $\ell'$**else**
    **update** $\mathcal{T}(n')$ (thus $\mathcal{O}$) with $\ell'$
    **if** $n = t$ **then update** $M$ with $\ell'$
**move** $\ell$ **from** $\mathcal{T}(n^\ell)$ (thus $\mathcal{O}$) **to** $\mathcal{P}(n^\ell)$ (thus $\mathcal{C}$)
**Go to** 2

This algorithm performs a unidirectional search. We now study how to adapt this algorithm to bidirectional search.

## Towards a bidirectional multiobjective search

We now recall the main features of single-objective bidirectional heuristic search, and we identify the difficulties raised by its extension to a multiobjective setting. An informed unidirectional search expands like a cone, and the number of examined nodes exponentially increases with the height of that cone. Bidirectional heuristic search performs simultaneously two unidirectional searches, in opposite directions, one from $s$ to $t$ (*forward search*), and the other from $t$ to $s$ (*backward search*). A new solution-path is found, each time the searches meet on a node. It might be less demanding to expand two cones of height $\mathcal{H}/2$ than one single cone of height $\mathcal{H}$, shedding interest to bidirectional search.

In the single-objective setting, there are no labels but only nodes. As far as bidirectional search is concerned, let us denote by $d \in \{\triangleright, \triangleleft\}$ the direction (forward or backward) and $g^d$ the associate cost function defined on $\mathbb{N}$ (such that $g^\triangleright(n) = g(s,n)$ and $g^\triangleleft(n) = g(n,t)$), and $\mathcal{O}^\triangleright$ (resp. $\mathcal{O}^\triangleleft$) the set of open nodes in the forward (resp. backward) search. Assuming that $c(n,m) = c(m,n)$, the algorithm typically alternates between steps of the forward and backward searches, by expanding a node in $\mathcal{O}^d$. To this end, it

employs two heuristic evaluation functions $h^{\triangleright}(n) = h(n,t)$ (forward evaluation) and $h^{\triangleleft}(n) = h(s,n)$ (backward evaluation) that are optimistic estimates of the cost of an optimal path from $n$ to $t$ and from $s$ to $n$ respectively. We then define $f^d(n) = g^d(n) + h^d(n)$. When both searches meet at a node $n$, a solution-path is detected, with cost $g^{\triangleright}(n) + g^{\triangleleft}(n)$. However such a solution-path is not necessarily an optimal one, and the search is therefore not terminated. Let us denote by $\mu$ the single value of the best single-objective solution-path found so far. The phase before a meeting point appears is called the *main phase*, and the phase after a meeting point is encountered is called the *post-phase*.

If $h \equiv 0$, the A$^*$ algorithm is nothing but Dijkstra's algorithm, bidirectional A$^*$ is the bidirectional variant of Dijkstra's algorithm, and $\mathbf{LB}_{g^d}(\mathcal{O}^d) = \min\{g^d(n) : n \in \mathcal{O}^d\}$ is a lower bound for all further labels from $\mathcal{O}^d$, for $d \in \{\triangleright, \triangleleft\}$. Hence, further meetings can only have a cost higher than $\mathbf{LB}_{g^{\triangleright}}(\mathcal{O}^{\triangleright}) + \mathbf{LB}_{g^{\triangleleft}}(\mathcal{O}^{\triangleleft})$. A termination condition is then for the two searches to have a common *closed* node, and the shortest path is the best path found so far (that does not necessarily pass through the common closed node). This termination condition clearly enables a short post-phase, and the bidirectional variant of Dijkstra's algorithm proved to outperform the unidirectional variant for a point-to-point request (searching for a path from $s$ to $t$). However, if a heuristic function is known, the A$^*$ algorithm uses the priority $f^d = g^d + h^d$ to decide which node to expand next, and the addition of $\mathbf{LB}_{g^{\triangleright}+h^{\triangleright}}(\mathcal{O}^{\triangleright})$ and $\mathbf{LB}_{g^{\triangleleft}+h^{\triangleleft}}(\mathcal{O}^{\triangleleft})$ loses its meaning, because of the two incompatible heuristics.

The key issues in implementing a bidirectional *heuristic* search is (1) to design an efficient termination condition, and (2) to avoid that the two searches overlap by avoiding nodes to be examined twice (once for each direction).

**Stopping criteria (single-objective case).** The first proposed algorithm for single-objective bidirectional *heuristic* search was BHPA (Pohl 1971), but the termination condition used (fronts-to-ends) did not enable a short post-phase. On the opposite, the more precise front-to-front evaluations are very time-consuming. Consequently, it was believed for a while that bidirectional search was not compatible with an informed search because one was not able to conciliate the use of an efficient termination condition and a heuristic guidance of which node to expand next. However, Ikeda et al. (1994) were able to reconcile both aspects. Let us recall that when heuristics are monotonic, A$^*$ is equivalent to running Dijkstra's algorithm over the modified costs $\underline{c}(n,m) = c(n,m) + h(m) - h(n)$. By resorting to a balanced heuristic function, that is, $h^{\triangleright}(n) + h^{\triangleleft}(n)$ equals a constant value $c$ for any node $n$, the modified costs $\underline{c}^{\triangleright}(n,m) = c(n,m) + h^{\triangleright}(m) - h^{\triangleright}(n)$ and $\underline{c}^{\triangleleft}(m,n) = c(m,n) + h^{\triangleleft}(n) - h^{\triangleleft}(m)$ in the forward and backward searches become consistent. Therefore, we can retrieve the meaning of the addition of $\mathbf{LB}_{g^{\triangleright}+h^{\triangleright}}(\mathcal{O}^{\triangleright})$ and $\mathbf{LB}_{g^{\triangleleft}+h^{\triangleleft}}(\mathcal{O}^{\triangleleft})$ by seeing it through the modified costs, and the termination condition of the bidirectional variant of Dijkstra's algorithm can therefore be recovered. It has been shown that this termination condition is equivalent to $\mu \leq \mathbf{LB}_{g^{\triangleright}+h^{\triangleright}}(\mathcal{O}^{\triangleright}) + \mathbf{LB}_{g^{\triangleleft}+h^{\triangleleft}}(\mathcal{O}^{\triangleleft}) - c$.

**Disabling intersections (single-objective case).** The *post-phase* can also be shortened by using techniques such as *nipping*. In a forward (resp. backward) expansion, if a new node $n$ is closed in the backward (resp. forward) search, then $\mu$ is updated, but $n$ should neither be extended, nor put in $\mathcal{O}^{\triangleright}$ (resp. $\mathcal{O}^{\triangleleft}$). Indeed, the optimal path from $n$ to $t$ (resp. $s$ to $n$) is already known, since $n$ is closed in the reverse search.

**A first bidirectional multiobjective search algorithm.** We now extend Algorithm 1 to bidirectional multiobjective preference-based search. Algorithm 2 performs simultaneously a forward PBMOA$^*$ and a backward one. Given a direction $d$, we denote by $\bar{d}$ the opposite direction. The operator $\langle P, \cdot \rangle$ is extended to append nodes in the two directions and also to append opposite paths. When the two searches meet, new solution-paths are found and the set $M$ is updated. We assume that the data structures for $\{\mathcal{L}(n)^d = \mathcal{T}(n)^d \cup \mathcal{P}(n)^d\}_{n \in N}$, $\mathcal{O}^d = \bigcup_{n \in N} \mathcal{T}^d(n)$ and $\mathcal{C}^d = \bigcup_{n \in N} \mathcal{P}^d(n)$ for $d \in \{\triangleright, \triangleleft\}$, and $M$ are created when needed.

**Algorithm 2**: MOPBS: bidirectional PBMOA$^*$

**Input**: State Space Graph $G$, starting node $s$, terminal node $t$, global preference relation $\succsim$, local preference relation $\succsim'$

1. INITIALIZATION
**insert** label $[s, \vec{0}, \langle s \rangle]$ into $\mathcal{T}^{\triangleright}(s)$ and $\mathcal{O}^{\triangleright}$
**insert** label $[t, \vec{0}, \langle t \rangle]$ into $\mathcal{T}^{\triangleleft}(t)$ and $\mathcal{O}^{\triangleleft}$

2. CHECK TERMINATION
**if** $\mathcal{O}^{\triangleright} = \emptyset$ or $\mathbf{LB}_{\Sigma f^{\triangleright}}(\mathcal{O}^{\triangleright}) > \mathbf{UB}_{\succsim}(M)$
or $\mathcal{O}^{\triangleleft} = \emptyset$ or $\mathbf{LB}_{\Sigma f^{\triangleleft}}(\mathcal{O}^{\triangleleft}) > \mathbf{UB}_{\succsim}(M)$
or **"stopping criterion" then**
  **return** $M$

3. DIRECTION AND LABEL SELECTIONS
$d \leftarrow \text{argmin}_{d \in \{\triangleright, \triangleleft\}} \{|\mathcal{O}^d|\}$
$\ell \leftarrow \text{top}_{\Sigma f^d}(\mathcal{O}^d)$

4. COUPLING
$C \leftarrow \bigcup_{\ell' \in \mathcal{L}^{\bar{d}}(n^{\ell})} \{[t, g^{d,\ell} + g^{\bar{d},\ell'}, \langle P^{\ell}, P^{\ell'} \rangle]\}$
**update** $M$ with $C$

5. LABEL EXPANSION
**for** *each node* $n' \in S^d(n^{\ell})$ **do**
  **create** $\ell' = [n', g^{d,\ell} + c(n^{\ell}, n'), \langle P^{\ell}, n' \rangle]$
  **if** LOCAL-DIS$_{\succsim'}(\mathcal{L}^d(n^{\ell'}), \ell')$ **then discard** $\ell'$**else**
  **if** GLOBAL-DIS$_{\succsim}(M, \ell')$ **then discard** $\ell'$**else**
    **update** $\mathcal{T}^d(n')$ (thus $\mathcal{O}^d$) with $\ell'$
**move** $\ell$ from $\mathcal{T}^d(n^{\ell})$ (thus $\mathcal{O}^d$) **to** $\mathcal{P}^d(n^{\ell})$ (thus $\mathcal{C}^d$)
**Go to** 2

Algorithm 2 written this way implements a fronts-to-ends stopping criterion. As in BHPA, it induces a so long post-phase that this algorithm unefficient compared to unidirectional search. The cartesian front-to-front stopping criterion that builds $\mathcal{O}^{\triangleright} \times \mathcal{O}^{\triangleleft}$, is also time-prohibitive. For this reason, it is worth investigating how to implement the **"stopping criterion"** at step 2.

In the next sections, we show how to adapt the balanced heuristic functions, to the preference-based multiobjective case; then we propose a variant of PBMOA$^*$ allowing to implement nipping in a $\succsim = \succsim_P$ setting.

## Stopping criteria in bidirectional PBMOA*

In this section we design efficient stopping criteria for bidirectional PBMOA*. They rely on comparisons of a *scalar upper bound* and a *scalar lower bound*. According to the definition of $\succsim \in \{\succsim_L, \succsim_\psi\}$, the upper bound $\mathbf{UB}_{\succsim}(M)$ is defined as in previous section.

The main difficulty is to conciliate the use of a heuristic, and an efficient meaningful lower bound. If $H^{\triangleright} \equiv H^{\triangleleft} \equiv \{(0,\dots,0)\}$, then Algorithm 2 is a bidirectional multiobjective Dijkstra. Recall that $\mathbf{LB}_{\Sigma g^d}(\mathcal{O}^d)$ is such that: for all labels $\ell$ in $\mathcal{O}^d$, for all solution-paths $P$ obtained by extending $P^\ell$ to the proper goal in the proper direction, $\mathbf{LB}_{\Sigma g^d}(\mathcal{O}^d) \leq \sum_i c_i(P)$ (where $c_i(P)$ denotes the $i^{th}$ component of $c(P)$). By the use of appropriate heaps, we can then access the scalar linear lower bounds $\mathbf{LB}_{\Sigma g^{\triangleright}}(\mathcal{O}^{\triangleright})$ and $\mathbf{LB}_{\Sigma g^{\triangleleft}}(\mathcal{O}^{\triangleleft})$ in constant time. Further label-couplings are linearly lower bounded by the scalar $\mathbf{LB}_{\Sigma g^{\triangleright}}(\mathcal{O}^{\triangleright}) + \mathbf{LB}_{\Sigma g^{\triangleleft}}(\mathcal{O}^{\triangleleft})$. We can then use $\mathbf{LB}_{\Sigma g^{\triangleright}}(\mathcal{O}^{\triangleright}) + \mathbf{LB}_{\Sigma g^{\triangleleft}}(\mathcal{O}^{\triangleleft}) > \mathbf{UB}_{\succsim}(M)$ as a stopping criterion for this Dijkstra version of Algorithm 2.

We now focus on using the ideal point heuristics: $h^{\triangleright,I}(n) = (h_1^I(n,t),\dots,h_p^I(n,t))$ and $h^{\triangleleft,I}(n) = (h_1^I(s,n),\dots,h_p^I(s,n))$, where $h_i^I(n,n')$ is the value of an optimal single objective shortest path from $n$ to $n'$ according to objective $i$. We recall that the addition of $\mathbf{LB}_{\Sigma g^{\triangleright}+h^{\triangleright,I}}(\mathcal{O}^{\triangleright})$ and $\mathbf{LB}_{\Sigma g^{\triangleleft}+h^{\triangleleft,I}}(\mathcal{O}^{\triangleleft})$ loses its meaning, because of the two incompatible heuristics.

Similarly to the single objective case, for $h \equiv h^I$, PBMOA*is equivalent to multiobjective Dijkstra over the modified cost-vectors $\underline{c}(n,m) = c(n,m) + h^I(m) - h^I(n)$. Then for a direction $d \in \{\triangleright,\triangleleft\}$ and a $d$-label $\ell$, the modified cost-vector $\underline{c}^d(P^\ell)$ is $g^{d,\ell} + h^{d,I}(n^l) - \overrightarrow{I}$, where $\overrightarrow{I} = h^I(s,t)$. By resorting to a balanced heuristic vector, here $h^{\triangleright}(n) + h^{\triangleleft}(n) = \overrightarrow{I}$ for any node $n$, the modified cost-vectors in both searches become consistent. Let us define, for $d \in \{\triangleright,\triangleleft\}$, such multiobjective vector-valued balanced heuristics $h^{d,J}$ (recall that $\bar{d}$ is the opposite direction):

$$h^{d,J}(n) = \frac{1}{2}(h^{d,I}(n) - h^{\bar{d},I}(n)) + \frac{1}{2}\overrightarrow{I}$$

Moreover, those $h^{d,J}$ heuristics are optimistic (thus admissible) and monotonic (with respect to summation). As usual, we will denote by $f^{d,J} = g^d + h^{d,J}$ the corresponding evaluations. From consistency, by denoting $\mathbf{LB}_{\Sigma f^{\triangleright,J}}(\mathcal{O}^{\triangleright})$ and $\mathbf{LB}_{\Sigma f^{\triangleleft,J}}(\mathcal{O}^{\triangleleft})$ the scalar linear minima of $\mathcal{O}^{\triangleright}$ and $\mathcal{O}^{\triangleleft}$, with respect to $f^{d,J}$, we can write a scalar linear lower bound for the *modified costs* of all further label-couplings: $(\mathbf{LB}_{\Sigma f^{\triangleright,J}}(\mathcal{O}^{\triangleright}) - \Sigma_i \overrightarrow{I_i}) + (\mathbf{LB}_{\Sigma f^{\triangleleft,J}}(\mathcal{O}^{\triangleleft}) - \Sigma_i \overrightarrow{I_i})$. It can be written to lower bound the *non*-modified costs of all further solutions $P$: $\mathbf{LB}_{\Sigma f^{\triangleright,J}} + \mathbf{LB}_{\Sigma f^{\triangleleft,J}}(\mathcal{O}^{\triangleleft}) - \Sigma_i \overrightarrow{I_i} \leq \sum_i c_i(P)$. We can therefore recover this stopping criterion:

$$\mathbf{UB}_{\succsim}(M) < \mathbf{LB}_{\Sigma f^{\triangleright,J}}(\mathcal{O}^{\triangleright}) + \mathbf{LB}_{\Sigma f^{\triangleleft,J}}(\mathcal{O}^{\triangleleft}) - \Sigma_i \overrightarrow{I_i}$$

Its efficiency for $\succsim \in \{\succsim_\psi, \succsim_L\}$ compared to unidirectional PBMOA*, is attested in section "Numerical experiments".

## Enabling Nipping in bidirectional PBMOA*

In single objective bidirectional search, if a node $n$ is selected for expansion and is already closed in the opposite search direction, *nipping* consists in closing it without expansion. It disables significantly the overlaping of the two searches. W.lo.g., assume that node $n$ is selected for expansion in the forward search while it is already closed in the backward search. Nipping is made possible because we already know the value of the optimal path from $n$ to $t$. In order to extend this technique to the multiobjective case, and to dispense with the forward expansion of a label at node $n$, we need to know *all* Pareto optimal paths (more precisely, all *images* in the objective space) from $n$ to $t$. Consequently, we propose here a variant of (unidirectional) multiobjective search where each iteration selects a nodes and aims at computing all pareto-optimal paths to this node being "developed". In PBMOA*, the presence of closed labels at node $n$ in the backward search means that we know *some* Pareto-optimal paths from $n$ to $t$, but we are not certain to know *all* Pareto optimal path from $n$ to $t$. Our definition of node developments is the following: developing a node $n$ in our variant can be seen as setting $n$ as a temporary goal node.

For simplicity, we present this Algorithm 3 in an unidirectional and $\succsim = \succsim_P$ framework. We assume that the data structures for $\{\mathcal{L}(n) = \mathcal{T}(n) \cup \mathcal{P}(n)\}_{n\in N}$, $\mathcal{O} = \bigcup_{n\in N} \mathcal{T}(n)$, and $\mathcal{C} = \bigcup_{n\in N} \mathcal{P}(n)$, are created when needed. The set $\mathcal{L}(t)$ fulfills the role of $M$. The set $K$ denotes the set of closed *nodes*. The temporary open labels $\mathcal{O}^{(n)}$ are ordered with respect to $\Sigma_i g_i(.) + h_i(.,n)$. The operation SELECT-LABELS$(\mathcal{O}, n)$ selects from $\mathcal{O}$ the labels $\ell$ such that $g^\ell + h(n^\ell, n)$ is not P-dominated by any label of $\mathcal{L}(n)$. Primitive TEMP-DIS$_{\succsim_P}(\mathcal{L}(n^\ell), \ell)$ returns true if there exists $\ell' \in \mathcal{L}(n^\ell)$ such that $g^{\ell'} \succsim_P g^\ell + h^\ell$; otherwise false.

**Algorithm 3**: unidirectional node-PBMOA*

**Input**: State Space Graph $G$,
starting node $s$, terminal node $t$

1. INITIALIZATION
**insert** label $[s, \vec{0}, \langle s \rangle]$ into $\mathcal{T}(s)$ and $\mathcal{O}$

2. CHECK TERMINATION
**if** $\mathcal{O} = \emptyset$ or $t \in K$ **then return** $\mathcal{L}(t)$

3. NODE SELECTION
$n \leftarrow \text{top}_{\Sigma f}(\mathcal{O})$
$\mathcal{O}^{(n)} \leftarrow$SELECT-LABELS$(\mathcal{O}, n)$

4. CHECK NODE TERMINATION
**if** $\mathcal{O}^{(n)} = \emptyset$ **then** $K \leftarrow K \cup \{n\}$ **and Go to** 2

5. LABEL EXPANSION
$\ell \leftarrow \text{top}_{\Sigma g(.)+h(.,n)}(\mathcal{O}^{(n)})$
**for** *each node* $n' \in S(n^\ell)$ **do**
  **create** $\ell' = [n', g^\ell + c(n^\ell, n'), \langle P^\ell, n' \rangle]$
  **if** GLOBAL-DIS$_{\succsim_P}(\mathcal{L}(t), \ell')$ **then discard** $\ell'$ **else if** LOCAL-DIS$_{\succsim_P}(\mathcal{L}(n^{\ell'}), \ell')$ **then discard** $\ell'$ **else**
    **update** $\mathcal{T}(n')$ (thus $\mathcal{O}$) with $\ell'$
    **if** not TEMP-DIS$(\mathcal{L}(n), \succsim_P, g^{\ell'} + h(\ell', n))$
    **then update** $\mathcal{O}^{(n)}$ with $\ell'$
**move** $\ell$ **from** $\mathcal{T}(n^\ell)$ (thus $\mathcal{O}$) **to** $\mathcal{P}(n^\ell)$ (thus $\mathcal{C}$)
**Go to** 4

We generalize Algorithm 3 to bidirectional search by performing at the same time a forward and a backward node-PBMOA$^*$. When the two searches meet, new solution-paths are found and the set $M$ is updated. When a label $\ell$ is selected for expansion and his *node* $n^\ell$ is *closed* in the opposite direction, it is no more necessary to expand $\ell$.

## Numerical experiments

We implemented using C++, and a standard 1.6Ghz PC.

**Instances.** Experiments were done on random unoriented graphs $\mathcal{G} = (V, E, c)$ generated as follows: First, $|V|$ vertices are drawn uniformly in $\{1, \dots, 1024\} \times \{1, \dots, 1024\}$, except for a source $v_s$ and a sink $v_t$ which are positioned to $(128, 512)$ and $(896, 512)$. Each vertex is linked by unoriented edges to the four closest vertices. For the easy instances, the cost vectors $c(e)$ of edges $e \in E$ are drawn uniformly in $\{0, \dots, 255\}^p$. Figure 2 displays such a random graph with $|V| = 400$ vertices. In the cost space, the convexity (resp. concavity) of the Pareto-set is a well-known cause of easyness (resp. hardness) for multiobjective problems. In problems with summations over randomly drawn edges costs, this convexity comes from the central limit theorem. To obtain the hard instances we disable this theorem both locally and globally as follows. Locally, to create the concavity of a $||.||_2$ sphere, the cost vectors $c$ of edges are first drawn uniformly in $[0, 1]^p$, and then normalized to an euclidian norm $||c||_2$ drawn in $\{2pM, \dots, 3pM\}$ with $M = 256$. Globally, to create bigger concavities, the edges adjacent to $v_s$ and $v_t$ are drawn in this same way but for $M = 256 \times 2\sqrt{|V|}$.

**Problems and State Spaces.** We tested two problems on these random unoriented graphs: MO shortest $s$-$t$-paths and MO minimal spanning trees.

Given a random graph $\mathcal{G} = (V, E, c)$, the state space $G = (N, A)$ for shortest $s$-$t$-paths is defined by: $N = V$, $s = v_s$, $t = v_t$, $A = \bigcup_{\{v,w\} \in E} \{(v, w)\} \cup \{(w, v)\}$. The ideal point heuristics $h(n, n')$ are precomputed for all pairs of nodes.

The state space $G = (N, A)$ for minimal spanning trees are defined by: the nodes $N = 2^V$ are the subsets covered by a tree, $s = \emptyset$, $t = V$ and $S(n) = \{n' \in 2^V : n \subset n' \wedge |n| + 1 = |n'| \wedge \exists \{v, v'\} \in E \, [n : (n' \setminus n)] \, , n \cup \{v'\} = n'\}$. When two trees cover the exact same subset $n \in 2^V$ of vertices with cost vectors $x, y \in \mathbb{N}^p$, each completion (with cost vector $z \in \mathbb{N}^p$) to a tree of one is also a completion to a tree of the other. Hence, if $x \succsim_P y$, then for all completions $x + z \succsim_P y + z$, and $y$ can be pruned. Therefore, optimisation can be done on these substructures. The ideal point heuristics are computed inline, using a variant of Prim's algorithm. This state space holds $|N| = 2^{|V|}$ nodes, and each state $n$ holds $|n|^{|n|-2}$ possible trees (Cayley's formula). Although this search approach is not the most convenient for this problem, it gives us an example of exponential size state space, with heuristics computed inline.

**Preferences and Algorithms.** We tested $\succsim \in \{\succsim_P, \succsim_L, \succsim_\psi\}$ for $w = (p, p-1, \dots, 2, 1)$ normalized to $\sum_i w_i = 1$. The algorithms used for Tables 1, 2, 3 are the bi (resp. uni) directional variants of label expanding PBMOA$^*$with $J$-heuristics (resp. $I$-heuristics) for the linear lower bounds.

**Notations** are the followings:
- $p$ is the number of objectives in the problem.
- $|N|$ is the number of nodes in the state space graph. Recall that it is $|V|$ for path problems and $2^{|V|}$ for tree problems.
- $|L|$ is the number of L-optimal solutions.
- $|E|$ (resp. $|E_0|$) is the median over 25 executions of the number of expanded labels in the bi (resp. uni ) directional algorithms.
- $\frac{|E|}{|E_0|}$ is the median over 25 instances (thus executions) of the ratios between the number of expanded labels of bidirectional over unidirectional algorithms.
- $T$ (resp. $T_0$) is the median over 25 executions of the cpu time (seconds) used by the bidirectional (resp. uni) algorithms.
- $\frac{T}{T_0}$ is the median over 25 instances (thus executions) of the ratios between the cpu time of bi- over unidirectional algorithms (median of ratios).

**About bidirectional PBMOA$^*$** Tables 1, 2, 3 summarize the numerical results obtained. The execution times are up to twenty times faster, and it goes even faster as the size of the state space (i.e., the number of states) increases, for all studied problems (see columns $\frac{|T|}{|T_0|}$). This can be easily explained by observing that considerably less labels are expanded in the bi-directional variant than in the unidirectional variant (see columns $\frac{|E|}{|E_0|}$). The relative behavior of both variants (uni- and bi-) is not influenced by the hardness of costs. Indeed, the hardness of costs have two consequences. First, it delays the triggering of the stopping criteria (which is very dependent on the shape of the Pareto front). Second, it multiplies the number of Lorenz optima (see column $L$). Whatever the variant used, these two factors impact similarly the number of label expansions, and therefore the ratios of running times do not change.

**About bidirectional node PBMOA$^*$.** Figure 2 represents the graph of a MO shortest path problem in blue, a forward search in light-gray (or green) right-oriented triangles, and a backward search in dark-gray (or red) left-oriented triangles. On each node, *areas* of the triangles are *proportional* to the *numbers of labels* expanded on this node. As one can see in bidirectional label expanding, forward and backward search intersect deeply. Moreover, the number of labels on a node grows with the distance to the source node. Bidirectional node expanding, by implementing MO nipping, successfully disables this deep intersection and enables the saving of many label expansions. Unfortunately, bidirectional node-PBMOA$^*$(despite half less labels to expand) uses two times more cpu-time than PBMOA$^*$.

## Conclusion

We have proposed in this paper several bidirectional variants of multiobjective search in state space graphs, for the OWA, Lorenz and Pareto preference models. For OWA and Lorenz, the introduction of bidirectional searches very significantly improves the running times on the two state spaces we studied, thanks to the existence of linear lower bounds that enable the design of good stopping criteria. Concern-

Table 1: Results for Lorenz path problems: monodirectional VS bidirectional label expanding.

| Lorenz path problems with easy costs | | | | | | |
|---|---|---|---|---|---|---|
| p | $\|N\|$ | $\|L\|$ | $\|E\|$ | $\frac{\|E\|}{\|E_0\|}$ | T | $\frac{T}{T_0}$ |
| 3 | 800 | 7 | 1605 | 0.19 | 0.12 | 0.19 |
| | 1600 | 6 | 4190 | 0.15 | 0.31 | 0.09 |
| | 2400 | 11 | 9578 | 0.14 | 0.86 | 0.08 |
| 5 | 400 | 6 | 814 | 0.12 | 0.13 | 0.10 |
| | 800 | 11 | 3989 | 0.08 | 0.7 | 0.03 |
| | 1200 | 14 | 7254 | 0.07 | 1.78 | 0.01 |
| 7 | 200 | 4 | 397 | 0.15 | 0.07 | 0.17 |
| | 400 | 6 | 1489 | 0.08 | 0.45 | 0.03 |
| | 600 | 8 | 3609 | 0.05 | 0.79 | 0.01 |
| Lorenz path problems with hard costs | | | | | | |
| p | $\|N\|$ | $\|L\|$ | $\|E\|$ | $\frac{\|E\|}{\|E_0\|}$ | T | $\frac{T}{T_0}$ |
| 3 | 80 | 7 | 1297 | 0.27 | 0.22 | 0.30 |
| | 160 | 9 | 1201 | 0.14 | 0.21 | 0.10 |
| | 240 | 18 | 3095 | 0.12 | 0.68 | 0.07 |
| 5 | 40 | 6 | 475 | 0.18 | 0.11 | 0.16 |
| | 80 | 13 | 4794 | 0.24 | 2.26 | 0.08 |
| | 120 | 18 | 13725 | 0.26 | 8.77 | 0.06 |
| 7 | 20 | 5 | 429 | 0.32 | 0.1 | 0.42 |
| | 40 | 7 | 1278 | 0.24 | 0.62 | 0.16 |
| | 60 | 7 | 1558 | 0.11 | 0.71 | 0.04 |

Table 2: Results for Lorenz tree problems: monodirectional VS bidirectional label expanding.

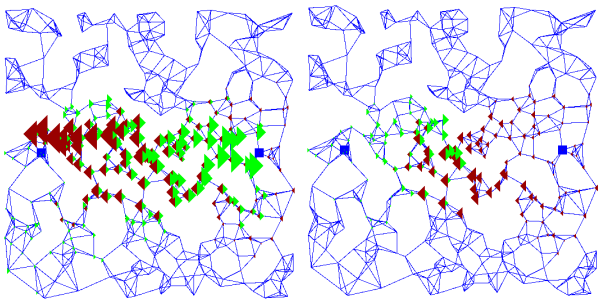
| Lorenz tree problems with easy costs | | | | | | |
|---|---|---|---|---|---|---|
| p | $\|N\|$ | $\|L\|$ | $\|E\|$ | $\frac{\|E\|}{\|E_0\|}$ | T | $\frac{T}{T_0}$ |
| 3 | $2^6$ | 3 | 123 | 0.7 | 0.11 | 3 |
| | $2^9$ | 6 | 638 | 0.21 | 1.93 | 1.11 |
| | $2^{12}$ | 6 | 2177 | 0.09 | 10.41 | 0.46 |
| 5 | $2^6$ | 4 | 216 | 0.45 | 0.46 | 2.8 |
| | $2^9$ | 8 | 1819 | 0.12 | 18.9 | 0.79 |
| | $2^{12}$ | 8 | 6621 | 0.04 | 180.44 | 0.21 |
| 7 | $2^6$ | 9 | 333 | 0.35 | 0.91 | 2.37 |
| | $2^9$ | 14 | 2434 | 0.08 | 33.01 | 0.18 |
| | $2^{12}$ | 29 | 16052 | 0.03 | 639.51 | 0.04 |
| Lorenz tree problems with hard costs | | | | | | |
| p | $\|N\|$ | $\|L\|$ | $\|E\|$ | $\frac{\|E\|}{\|E_0\|}$ | T | $\frac{T}{T_0}$ |
| 3 | $2^6$ | 7 | 186 | 0.62 | 0.18 | 3 |
| | $2^8$ | 13 | 829 | 0.35 | 1.94 | 1.62 |
| | $2^{10}$ | 12 | 3293 | 0.19 | 12.78 | 0.84 |
| 5 | $2^6$ | 11 | 300 | 0.57 | 0.38 | 2.83 |
| | $2^8$ | 21 | 1744 | 0.23 | 5.88 | 0.91 |
| | $2^{10}$ | 43 | 7491 | 0.12 | 66.66 | 0.11 |
| 7 | $2^6$ | 11 | 372 | 0.56 | 0.66 | 3.15 |
| | $2^8$ | 20 | 2309 | 0.29 | 11.23 | 0.79 |
| | $2^{10}$ | 40 | 9560 | 0.13 | 86.75 | 0.07 |

ing the determination of the Pareto set, the results are more mixed, since we do not know good stopping criteria based on scalar linear bounds. Nevertheless, by providing a new "node-oriented" version of multiobjective search, we managed to extend the nipping technique to the multiobjective case, thus avoiding almost completely that the two searches overlap and expanding much less labels than MOA*. Unfortunately, the running times are not yet satisfactory.

Table 3: Results for OWA path and tree problems: monodirectional VS bidirectional label expanding.

| OWA path problems | | | | | | |
|---|---|---|---|---|---|---|
| | Easy costs | | | Hard costs | | |
| p | $\|N\|$ | T | $\frac{T}{T_0}$ | $\|N\|$ | T | $\frac{T}{T_0}$ |
| 3 | 800 | 0.08 | 0.17 | 80 | 0.04 | 0.23 |
| | 1600 | 0.25 | 0.11 | 160 | 0.04 | 0.05 |
| | 2400 | 0.59 | 0.09 | 240 | 0.12 | 0.06 |
| 5 | 400 | 0.03 | 0.15 | 40 | 0.01 | 0.14 |
| | 800 | 0.16 | 0.06 | 80 | 0.19 | 0.09 |
| | 1200 | 0.57 | 0.03 | 120 | 0.22 | 0.04 |
| 7 | 200 | 0.01 | 0.21 | 20 | 0.01 | 0.14 |
| | 400 | 0.06 | 0.08 | 40 | 0.02 | 0.11 |
| | 600 | 0.15 | 0.03 | 60 | 0.03 | 0.03 |
| OWA tree problems | | | | | | |
| | Easy costs | | | Hard costs | | |
| p | $\|N\|$ | T | $\frac{T}{T_0}$ | $\|N\|$ | T | $\frac{T}{T_0}$ |
| 3 | $2^6$ | 0.09 | 2.44 | $2^6$ | 0.19 | 3.14 |
| | $2^9$ | 1.57 | 1.07 | $2^9$ | 5.32 | 1.54 |
| | $2^{12}$ | 7.63 | 0.37 | $2^{12}$ | 70.77 | 0.74 |
| 5 | $2^6$ | 0.41 | 2.73 | $2^6$ | 0.4 | 3.07 |
| | $2^9$ | 13.02 | 1.12 | $2^9$ | 14.89 | 1.12 |
| | $2^{12}$ | 132.1 | 0.49 | $2^{12}$ | 314.47 | 0.22 |
| 7 | $2^6$ | 0.6 | 2.74 | $2^6$ | 0.54 | 3.32 |
| | $2^9$ | 24.28 | 0.92 | $2^9$ | 29.44 | 0.87 |
| | $2^{12}$ | 444.39 | 0.29 | $2^{12}$ | 505.42 | 0.1 |

Figure 2: Searching the multiobjective Pareto-set: Bidirectional label expanding VS Bidirectional node expanding.



For future works, it is worth investigating further how to design an efficient bi-directional search algorithm to compute the Pareto set by optimizing the bidirectional "node-oriented" variant. Other research directions are to generalize the use of landmarks (Goldberg and Harrelson 2005) to the multiobjective case, or to make use of bounding sets (Ehrgott and Gandibleux 2007; Sourd and Spanjaard 2008), or to investigate approximation algorithms making use of relaxed termination criteria (Rice and Tsotras 2012) or approximate dominance relations (Perny and Spanjaard 2008).

## Acknowledgments

This research was funded by the French National Research Agency under grant ANR-09-BLAN-0361, within the GUEPARD project (Guarranteed Efficiency for Pareto Optimal Solutions Determination).

## References


Carraway, R.; Morin, L.; and Moskowitz, H. 1990. Generalized dynamic programming for multicriteria optimization. *European Journal of Operational Research* 44:95–104.

Chong, K. 1976. An induction theorem for rearrangements. *Canadian Journal of Mathematics* 28:154–160.

Dasgupta, P.; Chakrabarti, P.; and DeSarkar, S. 1995. Utility of *pathmax* in partial order heuristic search. *J. of algorithms* 55:317–322.

Ehrgott, M., and Gandibleux, X. 2007. Bound sets for biobjective combinatorial optimization problems. *Computers & OR* 34(9):2674–2694.

Felner, A.; Moldenhauer, C.; Sturtevant, N. R.; and Schaeffer, J. 2010. Single-frontier bidirectional search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Galand, L., and Perny, P. 2006. Search for Compromise Solutions in Multiobjective State Space Graphs. In *17th European Conference on Artificial Intelligence*, 93–97.

Galand, L., and Spanjaard, O. 2007. OWA-based search in state space graphs with multiple cost functions. In *FLAIRS Conference*, 86–91.

Galand, L.; Lesca, J.; and Perny, P. 2013. Dominance rules for the choquet integral in multiobjective dynamic programming. In *proceedings of IJCAI*.

Goldberg, A., and Harrelson, C. 2005. Computing the shortest path: *A* search meets graph theory. In *SODA*, 156–165.

Hansen, P. 1980. Bicriterion path problems. In Fandel, G., and Gal, T., eds., *Multiple Criteria Decision Making: Theory and Applications, LNEMS 177*. Springer-Verlag, Berlin. 109–127.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. and Cyb.* SSC-4 (2):100–107.

Ikeda, T.; Hsu, M.; Inai, H.; Nishimura, S.; Shimoura, H.; Hashimoto, T.; Tenmoku, K.; and Mitoh, K. 1994. A fast algorithm for finding better routes by AI search techniques. In *Proc. Vehicle Navigation and Information Systems Conference, IEEE*, 291–296.

Kaindl, H., and Kainz, G. 1997. Bi-directionnal heuristic search reconsidered. *Journal of Artificial Intelligence Research* 7:283–317.

Kwa, J. H. 1989. BS*: An admissible bidirectional staged heuristic search algorithm. *Artif. Intell.* 95–109.

Machuca, E.; Mandow, L.; Cruz, J.; and Ruiz-Sepulveda, A. 2010. An empirical comparison of some multiobjective graph search algorithms. In *KI 2010: Advances in Artificial Intelligence*, volume 6359 of *Lecture Notes in Computer Science*. 238–245.

Mandow, L., and Pérez de la Cruz, J. 2005. A new approach to multiobjective A* search. In *Proceedings of IJCAI-05*, 218–223. Professional Book Center.

Marshall, A., and Olkin, I. 1979. *Inequalities: Theory of Majorization and its Applications*. Academic Press.

Mitten, L. 1964. Composition principles for synthesis of optimal multistage processes. *Operations Research* 12:610–619.

Perny, P., and Spanjaard, O. 2002. On preference-based search in state space graphs. In *Proceedings of AAAI-02*, 751–756.

Perny, P., and Spanjaard, O. 2003. An Axiomatic Approach to Robustness in Search Problems with Multiple Scenarios. In *Proceedings of the 19th conference on Uncertainty in Artificial Intelligence*, 469–476.

Perny, P., and Spanjaard, O. 2008. Near admissible algorithms for multiobjective search. In *European Conference on Artificial Intelligence (ECAI)*, 490–494.

Pohl, I. 1971. Bi-directional search. *Machine Intelligence* 124–140.

Przybylski, A.; Gandibleux, X.; and Ehrgott, M. 2010. A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives. *Discrete Optimization* 7(3):149–165.

Rice, M. N., and Tsotras, V. J. 2012. Bidirectional a search with additive approximation bounds.

Shorrocks, A. 1983. Ranking income distributions. *Economica* 50:3–17.

Sourd, F., and Spanjaard, O. 2008. A multiobjective branch-and-bound framework: Application to the biobjective spanning tree problem. *INFORMS Journal on Computing* 20(3):472–484.

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

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