ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008
© 2008 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-891-5-490

Near Admissible Algorithms for Multiobjective Search

Patrice Perny and **Olivier Spanjaard**[1]

Abstract. In this paper, we propose near admissible multiobjective search algorithms to approximate, with performance guarantee, the set of Pareto optimal solution paths in a state space graph. Approximation of Pareto optimality relies on the use of an epsilon-dominance relation between vectors, significantly narrowing the set of non-dominated solutions. We establish correctness of the proposed algorithms, and discuss computational complexity issues. We present numerical experimentations, showing that approximation significantly improves resolution times in multiobjective search problems.

1 INTRODUCTION

Heuristic search in state space graphs was initially considered in the framework of single objective optimization. The value of a path is defined as the sum of the costs of its arcs and the problem amounts to finding one path with minimum cost among all paths from a source node to the goal. This problem is solved by constructive search algorithms like A^* [6] which provide the optimal solution-path. In this case preferences are measured by a scalar cost function inducing a complete weak-order over sub-paths. However, preferences are not always representable by a single criterion function. For example, in path planning problems for autonomous agents, the action allowing a transition from a state to another might have an impact in term of time, distance, energy consumption etc, thus leading to different points of views, non-necessarily reducible to a single overall cost [3]. More generally, multiobjective search is very useful in many applications requiring computer-aided problem solving (e.g., engineering design, preference-based configuration). It justifies the interest for search algorithms like MOA* [12], the multiobjective extension of A*, and its recent refinement by Mandow and Pérez-de-la-Cruz [8].

Besides these works on exact algorithms, several ε-admissible variations of the A* algorithm have been proposed in the literature (e.g. [11, 4]). These algorithms guarantee to find a solution that is within a factor of $(1+\varepsilon)$ of the best solution. They realize a compromise between time and space requirements on the one hand, and optimality of the returned solution on the other hand. These variations have proved to perform well, achieving a significant reduction of the number of iterations (up to 90% for $\varepsilon = 0.1$) to solve instances of the traveling salesman problem [11, 4]. Near admissible algorithms might also prove their efficiency in multiobjective search. At least, the introduction of tolerance thresholds in dominance concepts is worth investigating, with possibly a twofold benefit: not only it might simplify the search by increasing pruning possibilities, but it also might possibly reduce the size of the potential output (the set of non-dominated elements). This latter point is crucial as can be seen in the following example derived from Hansen [5].

Example 1. *Consider a simple biobjective state-space graph with a set $N = \{0,\ldots,q\}$ of nodes, 0 being the initial node and q being the goal node. At each node $n \in N\backslash\{q\}$, two actions a_1, a_2 are feasible: action a_1 leads to node $\{n+1\}$ with cost $(2^n, 0)$ whereas action a_2 leads to the same node with cost $(0, 2^n)$. By construction, there exists 2^q distinct solution-paths from 0 to q in this graph with costs $(k, 2^q-1-k)$ for $k = 0,\ldots,2^q-1$. For example the sequence of q times action a_1 yields a solution path with cost $(2^q-1, 0)$ whereas the sequence of q times action a_2 yields a solution path with cost $(0, 2^q-1)$. In that graph, all the paths from 0 to q have the same sum of costs but distinct costs on the first objective (due to the uniqueness of the binary representation of an integer). The images of all these paths in the space of objectives are on the same line (orthogonal to vector (1,1)) and therefore, they are all Pareto-optimal.*

In such family of instances, with q nodes and only 2 actions and 2 objectives, we can see that the number of Pareto-optimal paths grows exponentially with q. For instance, if $q = 16$ we have 65536 Pareto-optimal solution paths. This example shows that the exact determination of the Pareto set might induce prohibitive computation times. Moreover, producing the entire list of Pareto optimal solutions is probably useless for the Decision maker. In such cases, two approaches might be of interest: 1) focusing the search on a specific compromise solution; 2) approximating the Pareto set while keeping a good representation of the various possible tradeoffs in the Pareto set. The first approach requires additional preference information from the Decision Maker concerning, for example, the relative importance of criteria, the compensations allowed, and the type of compromise sought. When this information is not available, the second approach is particularly relevant. In this direction, several studies have been proposed, relying on the concept of *ε-dominance* introduced as an approximation of Pareto dominance in various multiobjective problems [14, 10, 2, 7, 1]. Despite the growing interest for these concepts, the potential of ε-relaxation of dominance concepts has not been investigated, to the best of our knowledge, in the framework of multiobjective search on implicit state space graphs. This is precisely the aim of this paper which is organized as follows. In the first two sections, we recall some useful results. In Section 2 we introduce formal material for the approximation of the Pareto set. In Section 3 we provide a simple reformulation of a multiobjective search algorithm to determine the exact Pareto set, and we prove its pseudopolynomiality. Then, we show how to modify this algorithm to get more efficient and near admissible versions. Finally, we provide numerical experimentations in the last section.

2 PARETO SET AND ITS APPROXIMATION

Considering a finite set of objectives $\{1,\ldots,m\}$ any solution-path can be characterized by a cost-vector $(c_1,\ldots,c_m) \in \mathbb{Z}_+^m$ where c_i represents the cost of the path with respect to objective i. Hence, the

[1] LIP6, Univ. Pierre and Marie Curie, 104 av. du Président Kennedy 75016 Paris, France, email: firstname.lastname@lip6.fr. This work has been supported by the ANR project PHAC which is gratefully acknowledged.

comparison of paths reduces to the comparison of their cost-vectors. The set of all cost-vectors attached to solution-paths is denoted X. We recall now some definitions linked to dominance concepts:

Definition 1. *The weak Pareto dominance relation ($\preccurlyeq_p$-dominance for short) on cost-vectors of $\mathbb{Z}_+^m$ is defined by:*

$$x \preccurlyeq_p y \iff [\forall i \in \{1,\dots,m\}, x_i \leq y_i]$$

Thus x dominates y, which is denoted by $x \preccurlyeq_p y$, when x is at least as good as y with respect to all objectives. For any dominance relation $\preccurlyeq$ defined on a set X, we will use the following definitions:

Definition 2. *Any element $x \in X$ is said to be $\preccurlyeq$-optimal in X if, for all $y \in X$, $y \preccurlyeq x \Rightarrow x \preccurlyeq y$. If x is not $\preccurlyeq$-optimal then it is said to be $\preccurlyeq$-dominated.*

Definition 3. *A subset $Y \subseteq X$ is said to be a $\preccurlyeq$-covering of X if for all $x \in X$ there exists $y \in Y$ such that $y \preccurlyeq x$. Whenever no proper subset of Y is a $\preccurlyeq$-covering of X, then Y is said to be a* minimal *$\preccurlyeq$-covering of X.*

The aim of multiobjective search is to find a $\preccurlyeq_p$-covering of the set of solution-paths. As shown in Example 1, such a set can be very large. This difficulty can be overcome by resorting to an approximate dominance concept called *ε-dominance relation* [5, 14]:

Definition 4. *The ε-dominance relation on cost-vectors of $\mathbb{Z}_+^m$ is defined by: $x \preccurlyeq_\varepsilon y \iff x \preccurlyeq_p (1+\varepsilon)y$*

As an illustration, consider the left part of Figure 1 concerning a bi-objective problem where every feasible solution is represented by a point $x = (x_1, x_2)$ in the bi-objective space. Within this space, point p^1 (resp. p^2) $\preccurlyeq_\varepsilon$-dominates all the points within cone C^1 (resp. C^2). The notion of $\preccurlyeq_\varepsilon$-covering arises then naturally. Indeed, the set $\{p^1, p^2\}$ is a 2-points $\preccurlyeq_\varepsilon$-covering of X since $X \subseteq C^1 \cup C^2$. Note that a smaller ε yields a finer $\preccurlyeq_\varepsilon$-covering of X, as illustrated on the right part of Figure 1, where a 5-points $\preccurlyeq_\varepsilon$-covering of the same set X is given.

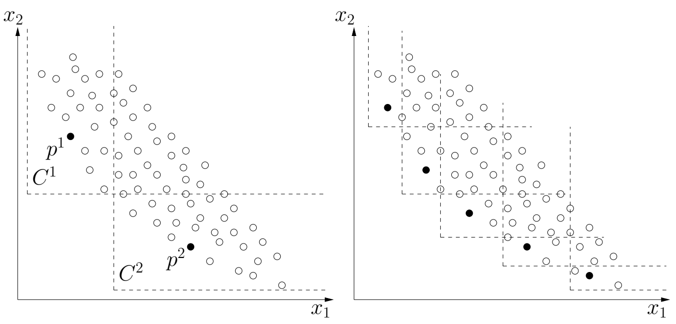

Figure 1. ε-coverings for two values of ε.

Note that, for a given ε, several minimal $\preccurlyeq_\varepsilon$-covering subsets of different sizes exist. For example, consider $X = \{x, y, z\}$ with $x = (800, 950)$, $y = (880, 880)$ and $z = (950, 800)$ and set $\varepsilon = 0.1$. The set $\{x, z\}$ is an $\preccurlyeq_\varepsilon$-covering subset of X since $800 \leq 968 = (1 + 0.1) \times 880$ and $950 \leq 968$ and thereby $x \preccurlyeq_\varepsilon y$. Furthermore, neither $x \preccurlyeq_\varepsilon z$ nor $z \preccurlyeq_\varepsilon x$, and therefore $\{x, z\}$ is minimal. Note that $\{y\}$ is also a minimal $\preccurlyeq_\varepsilon$-covering subset. On the one hand we have indeed $y \preccurlyeq_\varepsilon x$ since $880 \leq 880 = (1 + 0.1) \times 800$ and $880 \leq 1045 = (1 + 0.1) \times 950$, on the other hand $y \preccurlyeq_\varepsilon z$ for the same reasons. The very interest of ε-dominance lies in the following property: for any fixed number $m > 1$ of objectives, for any finite $\varepsilon > 0$ and any set X of bounded vectors x such that $1 \leq x_i \leq M$ for all $i \in \{1,\dots,m\}$, there exists a $\preccurlyeq_\varepsilon$-covering subset of X the size of which is polynomial in $\log M$ and $1/(\log(1+\varepsilon))$, see [10, 7]. This can simply be explained by considering a logarithmic scaling function $\varphi : \mathbb{Z}_+^m \to \mathbb{Z}_+^m$ on the objective space, defined as follows:

$$\varphi(x) = (\varphi_1(x), \dots, \varphi_m(x)) \text{ with } \varphi_i(x) = \left\lfloor \frac{\log x_i}{\log(1+\varepsilon)} \right\rfloor$$

For every component x_i, it returns an integer k such that $(1+\varepsilon)^k \leq x_i < (1+\varepsilon)^{k+1}$. Using φ we can define a φ-dominance relation:

Definition 5. *The φ-dominance relation on cost-vectors of $\mathbb{Z}_+^m$ is defined by: $x \preccurlyeq_\varphi y \iff \varphi(x) \preccurlyeq_p \varphi(y)$*

This relation satisfies the following properties:

Proposition 1. *For all vectors $x, y, z \in \mathbb{Z}_+^m$, we have:*
(i) $x \preccurlyeq_\varphi y$ and $y \preccurlyeq_\varphi z \Rightarrow x \preccurlyeq_\varphi z$ (transitivity)
(ii) $x \preccurlyeq_\varphi y \Rightarrow x \preccurlyeq_\varepsilon z$.

The symmetric part of $\preccurlyeq_\varphi$ defined by $x \equiv_\varphi y$ if and only if $\varphi(x) = \varphi(y)$ is therefore an equivalence relation (by transitivity). Clearly, by keeping one element of X for each equivalence class of $\equiv_\varphi$, one obtains an $\preccurlyeq_\varphi$-covering of X [10]. The left part of Figure 2 illustrates this point on the bi-objective example introduced for Figure 1. The dotted lines form a logarithmic grid in which each square represents an equivalence class for $\equiv_\varphi$. Hence the set of black points (one per non-empty square) represents a $\preccurlyeq_\varphi$-covering of all points. Interestingly enough, the resulting $\preccurlyeq_\varphi$-covering is also a $\preccurlyeq_\varepsilon$-covering set by Proposition 1 (ii). Moreover, the size of this $\preccurlyeq_\varepsilon$-covering is upper bounded by the number of equivalence classes of relation $\equiv_\varphi$, which is not greater than: $(1 + \lfloor \log M / \log(1+\varepsilon) \rfloor)^m$ [10]. A refined $\preccurlyeq_\varphi$-covering (which is also an $\preccurlyeq_\varepsilon$-covering) can easily be derived by removing $\preccurlyeq_\varphi$-dominated elements (we keep only the black points on the right part of Figure 2) which improves the bound to $(1 + \lfloor \log M / \log(1+\varepsilon) \rfloor)^{m-1}$, see [7]. Coming back to Example 1 with $q = 16$, a $\preccurlyeq_p$-covering requires 65536 solution-paths whereas a $\preccurlyeq_\varepsilon$-covering of this set constructed with φ as indicated above (for $\varepsilon = 0.1$) contains at most $\left\lfloor \frac{\log 65536}{\log 1.1} \right\rfloor + 1 = 117$ elements. More generally, it is important to note that, for fixed values of ε and m, the size of the $\preccurlyeq_\varepsilon$-covering grows only polynomially with the size of the instance, even when the Pareto set grows exponentially. In addition, if a set $Y \subseteq X$ is an $\preccurlyeq_\varepsilon$-covering of X, we know (by Definition 3) that any feasible tradeoff achieved in X is approximated with performance garantee, i.e it is $\preccurlyeq_\varepsilon$-dominated by at least one element in Y. This enables a more concise and yet representative description of possible tradeoffs in the Pareto set. The question is whether an $\preccurlyeq_\varepsilon$-covering is computable in polynomial time or not.

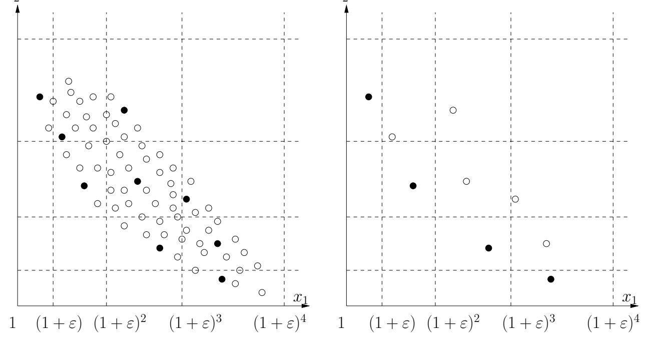

Figure 2. Logarithmic grid.

3 MULTIOBJECTIVE SEARCH ALGORITHM

We now present a multiobjective extension of A* (reformulation of the label-expanding version of MOA* [8]), and we prove its pseudopolynomiality, which is directly related to that of the size of the Pareto set. In the following sections, we will then use a logarithmic grid to derive near-admissible algorithms: a first one the complexity of which is polynomial in the number of states, a second one more efficient in practice in spite of a higher theoretical complexity. To our knowledge, this is the first attempt to devise near admissible algorithms for multiobjective search in *implicit* graphs (the existing near admissible multiobjective algorithms work in *explicit* graphs).

A* algorithm and its multiobjective extensions explore a state space graph $G = (N, A)$ where N is a finite set of nodes (possible states), and A is a set of arcs representing transitions. Formally, we have $A = \{(n, n') : n \in N, n' \in S(n)\}$ where $S(n) \subseteq N$ is the set of all successors of node n. A cost-vector $v(n, n')$ is attached to each arc $(n, n') \in A$, and the cost-vector of a path P is defined by $v(P) = \sum_{(n,n') \in P} v(n, n')$. In the sequel, we assume that $v(P) \in [1, M]$ for every solution path, where M is a known constant. Then $s \in N$ denotes the source of the graph (the initial state), $\Gamma \subseteq N$ the subset of goal nodes, $\mathcal{P}(s, \Gamma)$ the set of all paths from s to a goal node $\gamma \in \Gamma$ (solution-paths), and $\mathcal{P}(n, n')$ the set of all paths linking n to n', characterized by a list $\langle n, \dots, n' \rangle$ of nodes.

Unlike the scalar case, there possibly exists several $\preccurlyeq_P$-optimal paths with distinct cost-vectors to reach a given node in a multiobjective problem. Hence, one expands labels $\ell = [n_\ell, P_\ell, g_\ell]$ (attached to subpaths) rather than nodes, where n_ℓ indicates the labeled node, P_ℓ the corresponding subpath in $\mathcal{P}(s, n_\ell)$, and g_ℓ the cost-vector of P_ℓ. As in A*, the set of generated labels is divided into two disjoint sets: a set OPEN of not yet expanded labels and a set CLOSED of already expanded labels. Besides, the $\preccurlyeq_P$-optimal expanded labels in $\{\ell : n_\ell \in \Gamma\}$ are stored in a set SOL. Since a node n may be on the path of more than one $\preccurlyeq_P$-optimal solution, a *set* $H(n)$ of heuristic cost-vectors is given for each node n, estimating the set $\{v(P) : P \in \mathcal{P}(n, \Gamma)\}$. For each generated label ℓ_0, a *set* $F(\ell_0)$ of evaluation vectors is computed from all possible combinations $\{g_{\ell_0} + h, h \in H(n_{\ell_0})\}$. It estimates the set of $\preccurlyeq_P$-optimal values of solution-paths extending P_{ℓ_0}. Initially, OPEN contains only label $[s, \langle s \rangle, \vec{0}]$, while CLOSED and SOL are empty. At each subsequent step, one expands a label ℓ^* in OPEN such that $F(\ell^*)$ contains at least one $\preccurlyeq_P$-optimal vector in $\cup_{\ell \in \text{OPEN}} F(\ell)$. The process is kept running until OPEN becomes empty. Two pruning rules are used:

Rule R_1: discard label ℓ when there exists $\ell' \in$ OPEN $\bigcup$ CLOSED s.t. $n_{\ell'} = n_\ell$ and $g_{\ell'} \preccurlyeq_P g_\ell$.

Rule R_2: discard label ℓ when $\forall f \in F(\ell), \exists \ell' \in$ SOL s.t. $g_{\ell'} \preccurlyeq_P f$.

These rules ensure to generate all $\preccurlyeq_P$-optimal paths in $\mathcal{P}(s, \Gamma)$ provided heuristic H is admissible, i.e. $\forall n \in N, \forall P \in \mathcal{P}(n, \Gamma), \exists h \in H(n)$ s.t. $h \preccurlyeq_P v(P)$. The algorithm is outlined below:

```
MULTIOBJECTIVE SEARCH ALGORITHM (MOA*)
Input: $G$, OPEN, CLOSED, SOL
while OPEN $\neq \emptyset$
01  move a label $\ell^*$ from OPEN to CLOSED
02  if $n_{\ell^*} \in \Gamma$
03    then UPDATE(SOL, $\emptyset$, $\ell^*$)
04    else for each node $n' \in S(n_{\ell^*})$ do
05           create $\ell_0 = [n', \langle P_{\ell^*}, n' \rangle, g_{\ell^*} + v(n, n')]$
06           if $\exists f_0 \in F(\ell_0)$ s.t. $\forall \ell \in$ SOL not($g_\ell \preccurlyeq_P f_0$)
07             then UPDATE(OPEN($n'$), CLOSED($n'$), $\ell_0$)
08             else discard $\ell_0$
Output: SOL
```

This algorithm calls procedure UPDATE which applies to L$_1$, a list of open labels, and L$_2$, a list of closed labels. It possibly updates list L$_1$ with label ℓ as follows:

```
UPDATE(L$_1$, L$_2$, $\ell$)
01 If $\forall \ell' \in$ L$_1 \cup$ L$_2$ not($g_{\ell'} \preccurlyeq_P g_\ell$) then L$_1 \leftarrow$ L$_1 \cup \{l\}$
02 Remove $\preccurlyeq_P$-dominated labels from L$_1$
```

We now show that this multiobjective search algorithm is pseudopolynomial for integer costs (for a fixed number m of objectives), with the following worst case complexity analysis. The "while" loop in the main procedure is iterated at most $|N|\,(M+1)^m$ times since this is the maximum number of distinct labels. Indeed there are $|N|$ nodes, and for each of them, the number of different cost vectors is upper bounded by $(M+1)^m$. Furthermore, at each iteration of the loop the main computational cost is due to line 06 which requires binary comparisons of labels from $F(\ell_0)$ and SOL. With a naive method, this represents $(M+1)^{2m}$ comparisons. Hence the algorithm executes less than $|N|\,(M+1)^m$ loops of cost $(M+1)^{2m}$. Therefore the overall complexity is within $O(|N|^2\, M^{3m})$.

4 APPROXIMATION ALGORITHMS

We consider now two ways of relaxing the exact version of the multiobjective search algorithm so as get a better efficiency, either by modification of R_1 or R_2 (both modification cannot be performed together without losing the performance guarantee).

4.1 An FPTAS for multiobjective search

In this subsection, we assume that a finite upper bound L on the lengths (numbers of arcs) of all solution-paths in $\mathcal{P}(n, \Gamma)$ is known. Under this assumption, we provide a *Fully Polynomial Time Approximation Scheme* (FPTAS) for computing an approximation of the Pareto set. For simplicity, we assume throughout this section that the input is a finite graph on $|N|$ nodes. Several FPTAS to compute $\preccurlyeq_\varepsilon$-coverings in multiobjective shortest path problems (MSP) have been proposed in the literature; that is, algorithms that, given an encoding of the graph and an accuracy level $\varepsilon > 0$, yield an $\preccurlyeq_\varepsilon$-covering in time and space bounded by a polynomial in $|N|$ and $\frac{1}{\varepsilon}$. Hansen [5] and Warburton [14] have proposed methods combining *rounding and scaling* techniques (i.e., approximating data elements before the execution of an algorithm) with *pseudopolynomial* exact algorithms (i.e., algorithms that operate in time and space bounded by a polynomial in $|N|$ and the largest data element), in order to keep polynomially bounded the size of the auxiliary data computed during the execution. These methods are particular to biobjective problems and acyclic graphs respectively. Another algorithm is due to Papadimitriou and Yannakakis [10]. It is less specific to MSP, and its interest resides mainly in its generality: it proceeds by computing one solution (if it exists) inside every box of the logarithmic grid of Section 2. The authors show that this can be polynomially performed in a problem A if there is a pseudopolynomial algorithm for the exact version of A (given an instance of A and an integer B, is there a feasible solution with cost exactly B?). Finally, Tsaggouris and Zaroliagis [13] have recently proposed an FPTAS based on a generalized Bellman-Ford algorithm. Except for [10], all the other approaches rely on dynamic programming. We now show how to obtain an FPTAS by applying trimming techniques to the multiobjective search algorithm. The idea is to keep polynomially bounded the number of possible labels at each node, by using a logarithmic grid. Nevertheless, it is not possible to work directly with $\preccurlyeq_\varphi$ in place of $\preccurlyeq_P$ within procedure UPDATE because we might exceed the desired error threshold $(1+\varepsilon)$ due to error propagations, as shown in the following example.

Example 2. *Consider the graph with nodes* $\{s, n, n', \gamma\}$ *and costs:* $v(s, n) = (2, 2)$, $v(s, n') = (1, 1.1)$, $v(n', n) = (0.9, 1)$, $v(n, \gamma) = (1, 1)$, $v(n', \gamma) = (2.3, 1.8)$. *We set* $\varepsilon = 0.1$. *We get two labels at node n,* $\ell_1 = [n, \langle s, n\rangle, (2, 2)]$ *and* $\ell_2 = [n, \langle s, n', n\rangle, (1.9, 2.1)]$. *Since* $\ell_1 \preccurlyeq_\varepsilon \ell_2$, *assume that* ℓ_2 *is discarded. We get two labels* $\ell_3 = [\gamma, \langle s, n', \gamma\rangle, (3.3, 2.9)$ *and* $\ell_4 = [\gamma, \langle s, n, \gamma\rangle, (3, 3)]$ *at* γ. *At this point* ℓ_4 *might be discarded since* $\ell_3 \preccurlyeq_\varepsilon \ell_4$. *In this case the unique returned solution path would be* $\langle s, n', \gamma\rangle$ *with cost* $(3.3, 2.9)]$. *However it is clear that path* $\langle s, n', n, \gamma\rangle$ *with cost* $(2.9, 3.1)$ *is not* ε*-covered by* $(3.3, 2.9)$. *Actually we have:* $(3.3, 2.9) \preccurlyeq_P 1.1(3, 3)$ *and* $(3, 3) \preccurlyeq_P 1.1(2.9, 3.1)$ *but not* $(3.3, 2.9) \preccurlyeq_P 1.1(2.9, 3.1)$. *We only have* $(3.3, 2.9) \preccurlyeq_P 1.1^2(2.9, 3.1)$.

This example suggests a possible solution relying on the assumption that solution-paths contain at most L arcs. We might replace $(1 + \varepsilon)$ by $(1 + \varepsilon)^{\frac{1}{L}}$ to remain below $(1 + \varepsilon)$ by propagation of errors. This idea is implemented in the following revised pruning rule.

Rule R'_1: discard label ℓ when there exists $\ell' \in$ OPEN $\bigcup$ CLOSED s.t. $n_{\ell'} = n_\ell$ and $\psi(g_{\ell'}) \preccurlyeq_P \psi(g_\ell)$

where ψ is a logarithmic scaling function $\psi : \mathbb{Z}^m_+ \to \mathbb{Z}^m_+$ on the objective space, defined as follows:

$$\psi(x) = (\psi_1(x), \ldots, \psi_m(x)), \psi_i(x) = \left\lfloor \log x_i / \log(1 + \varepsilon)^{1/L} \right\rfloor$$

This lead to replace procedure UPDATE by:

```
ψ−UPDATE(L1, L2, ℓ)
01 If $\forall \ell' \in L_1 \cup L_2$ not($\psi(g_{\ell'}) \preccurlyeq_P \psi(g_\ell)$)
02     then $L_1 \leftarrow L_1 \cup \{l\}$
03 Remove $\preccurlyeq_\psi$-dominated labels from $L_1$
```

With $\psi-$UPDATE the multiobjective search algorithm becomes polynomial in $|N|$ and $\frac{1}{\varepsilon}$, provided $1 \leq M \leq 2^{p(|N|)}$ where p denotes some polynomial. Indeed, the cost of every solution path on the logarithmic scale is upper bounded by $\left\lfloor \log M / \log(1 + \varepsilon)^{\frac{1}{L}} \right\rfloor \in O(L \log M/\varepsilon)$. Hence, the global complexity of the algorithm becomes $O(|N|^2 (L \log M/\varepsilon)^{3m})$. Since $L \leq |N|$ and $\log M \in O(p(|N|))$, it is within $O((\frac{1}{\varepsilon})^{3m} p(|N|))$ for some polynomial p and therefore polynomial in $\frac{1}{\varepsilon}$ and $|N|$. Now, it remains to show that this version of the algorithm yields a $\preccurlyeq_\varepsilon$-covering subset of the solution-paths. To this end we state the following propositions:

Proposition 2. *For all* $i \in \{1, \ldots, L\}$, $\forall x, y, z \in X$ *the following* monotonicity *property hold:*
$x \preccurlyeq_P y(1 + \varepsilon)^{\frac{i}{L}} \Rightarrow (x + z) \preccurlyeq_P (y + z)(1 + \varepsilon)^{\frac{i}{L}}$

Note that this monotonicity property does not hold for the ψ-dominance relation induced by $\psi(x) \preccurlyeq_P \psi(y)$.

Proposition 3. *Let* $P \in \mathcal{P}(s, \Gamma)$. *At any time before termination, if* $\forall \ell \in$ SOL *not*$(g_\ell \preccurlyeq_\varepsilon v(P))$, *then there exists* $\ell' \in$ OPEN *and* P' *extending* $P_{\ell'}$ *such that* $v(P') \preccurlyeq_\varepsilon v(P)$.

Proof. Consider a solution-path $P = \langle s, n_1, \ldots, n_k \in \Gamma\rangle$. By contraposition, assuming that for all $\ell' \in$ OPEN no solution-path P' extending $P_{\ell'}$ is such that $v(P') \preccurlyeq_\varepsilon v(P)$, we show that there exists a label $\ell \in$ SOL for which $g_\ell \preccurlyeq_\varepsilon v(P)$. For that purpose, we exhibit a finite sequence (ℓ_i) of closed labels generated during the search, such that $g_{\ell_i} \preccurlyeq_P (1 + \varepsilon)^{\frac{i}{L}} v(P_i)$ (1), where $P_i = \langle s, n_1, \ldots, n_i\rangle$. We proceed as follows: for $i = 0$, we set $\ell_0 = [s, \langle s\rangle, \vec{0}]$ and we clearly have $g_{\ell_0} \preccurlyeq_P (1 + \varepsilon)^{\frac{0}{L}} v(P_0)$. Inductively, assume now that labels $\ell_0, \ldots, \ell_j$ have been generated and closed $(j < k)$, such that Equation 1 holds for $i = 0, \ldots, j$. Let $\ell = [n_{j+1}, \langle P_{\ell_j}, n_{j+1}\rangle, g_{\ell_j} + v(n_j, n_{j+1})]$ be the label of the path from s to n_{j+1} extending P_{ℓ_j}. This label has been generated since ℓ_j has been expanded and $n_{j+1} \in S(n_j)$. There are two cases:
Case 1. If $\ell \in$ CLOSED then we set $\ell_{j+1} = \ell$.
Case 2. If $\ell \notin$ CLOSED, we cannot have $\ell \in$ OPEN since it would contradict the initial assumption. Indeed, consider solution-path $P' = \langle P_{\ell_j}, n_{j+1}, \ldots, n_k\rangle$. We would have: $v(P') = g_{\ell_j} + v(\langle n_j, \ldots, n_k\rangle) \preccurlyeq_P (1 + \varepsilon)^{\frac{j}{L}} v(P_j) + v(\langle n_j, \ldots, n_k\rangle) \preccurlyeq_P (1 + \varepsilon)v(P_j) + (1 + \varepsilon)v(\langle n_j, \ldots, n_k\rangle) \preccurlyeq_P (1 + \varepsilon)v(P)$. Hence, ℓ has been generated, but $\ell \notin$ OPEN $\cup$ CLOSED. Therefore ℓ has been discarded using pruning rule R'_1 or R_2:
Case 2.1. If ℓ is discarded by R'_1, then there exists $\ell' \in$ OPEN $\cup$ CLOSED such that $n_{\ell'} = n_\ell$ and $\psi(g_{\ell'}) \preccurlyeq_P \psi(g_\ell)$, which implies $g_{\ell'} \preccurlyeq_P (1 + \varepsilon)^{\frac{1}{L}} g_\ell$. We have $g_{\ell'} \preccurlyeq_P (1 + \varepsilon)^{\frac{1}{L}} g_\ell = (1 + \varepsilon)^{\frac{1}{L}} (g_{\ell_j} + v(n_j, n_{j+1})) \preccurlyeq_P (1 + \varepsilon)^{\frac{1}{L}} ((1 + \varepsilon)^{\frac{j}{L}} v(P_j) + v(n_j, n_{j+1})) \preccurlyeq_P (1 + \varepsilon)^{\frac{j+1}{L}} v(P_{j+1})$. Moreover, by the same reasoning as above with $P' = \langle P_{\ell'}, n_{j+2}, \ldots, n_k\rangle$, we have $v(P') \preccurlyeq_\varepsilon v(P)$, and therefore ℓ' cannot be in OPEN. Hence, $\ell' \in$ CLOSED and we set $\ell_{j+1} = \ell'$.
Case 2.2. If R_2 prunes ℓ, then sequence (ℓ_i) is stopped.
Whenever case 2.2 stops the sequence $\ell_0, \ldots, \ell_j$ by discarding label ℓ, then for all $f \in F(\ell)$, there exists $\ell' \in$ SOL s.t. $g_{\ell'} \preccurlyeq_P f$ (Eq. 1). Moreover, there exists $f \in F(\ell)$ such that $f \preccurlyeq_\varepsilon v(P)$, as we now show. By admissibility of H, there exists $h \in H(n_{j+1})$ such that $h \preccurlyeq_P v(\langle n_{j+1}, \ldots, n_k\rangle)$. Then, there exists $f = g_\ell + h \in F(\ell)$ such that $f \preccurlyeq_P g_\ell + v(\langle n_{j+1}, \ldots, n_k\rangle) = g_{\ell_j} + v(\langle n_j, \ldots, n_k\rangle) \preccurlyeq_P (1 + \varepsilon)^{\frac{j}{L}} v(P_j) + v(\langle n_j, \ldots, n_k\rangle) \preccurlyeq_P (1 + \varepsilon)v(P_j) + (1 + \varepsilon)v(\langle n_j, \ldots, n_k\rangle) = (1 + \varepsilon)v(P)$. Hence $f \preccurlyeq_P (1 + \varepsilon)v(P)$ (Eq. 2). From Eq. 1 and Eq. 2, we get $g_{\ell'} \preccurlyeq_\varepsilon v(P)$ (by transitivity of $\preccurlyeq_P$).
Whenever case 2.2 does not occur, the sequence continues until $j = k$. Once label ℓ_k has been expanded at n_k, solution-path P_k has been discovered and SOL includes a label ℓ such that $g_\ell \preccurlyeq_\varepsilon v(P)$.
In all cases, the existence of $\ell \in$ SOL with $g_\ell \preccurlyeq_\varepsilon v(P)$ is proved. □

From this proposition, it follows that the algorithm cannot terminate as long as the solution-paths stored in SOL does not constitute a $\preccurlyeq_\varepsilon$-covering of $\mathcal{P}(s, \Gamma)$. Indeed, it would imply that OPEN is non-empty, which contradicts the termination of the algorithm. We can therefore conclude that the algorithm returns a $\preccurlyeq_\varepsilon$-covering subsets of solution-paths. Note that this technique is mainly of theoretical interest, since the complexity is quadratic in the number $|N|$ of states but $|N|$ is usually exponential in the depth of the search. We propose therefore below a simpler technique that also guarantees the approximation, more efficient in practice in spite of a higher complexity.

4.2 A near admissible version of MOA*

We now present the MOA$^*_\varepsilon$ algorithm, which returns a $\preccurlyeq_\varepsilon$-covering of solution-paths without requiring the knowledge of an upper bound on the number of nodes that can be expanded. The basic features of the algorithm are essentially the same as MOA*. The main difference lies in the following pruning rule which uses $\preccurlyeq_\varepsilon$-dominance:

Rule R'_2: discard label ℓ when $\forall f \in F(\ell), \exists \ell' \in$ SOL s.t. $g_{\ell'} \preccurlyeq_\varepsilon f$.

This rule allows an early elimination of uninteresting labels while keeping near admissibility of the algorithm provided heuristic H is admissible. Indeed, if H is admissible, then for all $f^* \in F^*(\ell)$ there exists $f \in F(\ell)$ such that $f = g_\ell + h \preccurlyeq_P g_\ell + h^* = f^*$. Hence $g_{\ell'} \preccurlyeq_\varepsilon f$ implies that $g_{\ell'} \preccurlyeq_\varepsilon f^*$. This pruning rule can be inserted in the multiobjective search algorithm by substituting line 06 by:

```
06'        if $\exists f_0 \in F(\ell_0)$ s.t. not($f(\ell) \preccurlyeq_\varepsilon f_0$) $\forall \ell \in$ SOL
```

Despite MOA$^*_\varepsilon$ does not provide complexity guarantees, it outperforms significantly the exact version.

Remark 1. *The following weaker relaxation of the pruning condition in R_2 can be used in the FPTAS:*

$06''$ if $\exists f_0 \in F(\ell_0)$ s.t. not$(f(\ell) \preccurlyeq_p (1+\varepsilon)^{\frac{k}{L}} f_0)$ $\forall \ell \in$ SOL

where k is an upper bound of the length of the longest path from n_{ℓ_0} to a goal. This is the case in the implemented version.

5 NUMERICAL EXPERIMENTATIONS

To investigate the potential of approximation, we tested our algorithms on two multiobjective combinatorial problems.

Biobjective binary knapsack problem. Given a set $\{1,\ldots,n\}$ of items j, each item having a weight w_j and a profit p_{ij} according to every objective i, one searches a minimal $\preccurlyeq_p$-covering of combinations of items that can be put into a knapsack of capacity b (i.e., the total weight of the items cannot run over b):

$$\max \sum_{j=1}^{n} p_{1j}x_j, \quad \max \sum_{j=1}^{n} p_{2j}x_j$$
$$\text{subject to } \sum_{j=1}^{n} w_j x_j \le b$$
$$x_j \in \{0,1\}\ \forall j \in \{1,\ldots,n\}$$

where $x_j = 1$ iff one chooses to put item j in the knapsack. The state space has been defined such that all solution-paths share the same length n. This enables to apply the FPTAS with $L = n$. The heuristic evaluations used to order and prune the search derive from the upper bound of Martello and Toth [9] for the single objective version. The MOA*, FPTAS and MOA$^*_\varepsilon$ algorithms have been implemented in JAVA and were run on a Pentium 4 3.60GHz PC. Table 1 shows computation times (in sec) obtained on 35 random instances of size n, with profits and weights randomly drawn in $[1, 100]$, and a capacity b bounded to 50% of the total weight of the items. These results show that the relaxation of the optimality condition significantly speeds up the search, with faster results when using MOA$^*_\varepsilon$. We have also studied the behavior of MOA$_\varepsilon$ when setting $\forall j\ p_{1j} = 2^j$, $p_{2j} = 2^n - 2^j$ and $w_j = 1$, which yields instances where all combinations of b items are non-dominated with distinct profits on the first objective. In the first line of Table 2, we indicate the execution times of MOA*, and in the second line the number $\#sol$ of non-dominated solutions (which grows exponentially with n). Both approximation algorithms return a $\preccurlyeq_\varepsilon$-covering in less than one second for all ε in $\{0.005, 0.01, 0.05\}$. For each value of ε we give the size of the returned $\preccurlyeq_\varepsilon$-covering. It shows that the choice of ε allows the size of the output set to be controlled, as well as the computation times.

n	30	40	50	60	70	80
			MOA*			
time	0.397	1.879	11.31	43.66	215.2	457.7
ε			FPTAS			
0.005	0.353	1.514	7.922	29.90	127.8	226.5
0.01	0.297	1.077	4.842	18.29	65.91	97.26
0.05	0.046	0.036	0.065	0.331	0.555	0.393
0.1	0.003	0.001	0.001	0.002	0.001	0.001
ε			MOA$^*_\varepsilon$			
0.005	0.315	0.940	4.225	18.58	62.37	110.4
0.01	0.179	0.364	1.389	9.294	19.75	35.11
0.05	0.008	0.007	0.013	0.064	0.065	0.075
0.1	0.001	0.001	0.001	0.001	0.001	0.001

Table 1. Numerical results on the biobjective knapsack.

n	15	16	17	18	19	20
time	0.495	3.328	1.895	44.08	17.55	711.9
# sol	6.10^3	1.10^4	2.10^4	5.10^4	9.10^4	2.10^5
0.005	31	28	28	25	25	22
0.01	16	14	15	13	13	11
0.05	3	3	4	3	3	3

Table 2. Pareto approximation on pathological instances.

Multiobjective shortest path problem. In order to study the interest of approximation when the number of objectives grows, we have performed experimentations of MOA$^*_\varepsilon$ on the multiobjective path problem. We have generated different classes of instances by controlling the number of nodes $|N| = 1000, 2000, 3000$ and the number of objectives $m = 2, 5, 10$. Cost of arcs are randomly generated within $[1, 100]$. The approximations have been computed with $\varepsilon = 0.1$. Table 3 gives, in each class of instances, the average execution time (in sec) obtained on 20 different instances. These performances illustrate that approximation remains powerful when the number of objectives grows. As a comparison, the exact determination (with MOA*) of the Pareto set on instances with 1000 nodes and 10 criteria required more that one hour on the same computer.

$\|N\|$	1000	2000	3000
2 obj	0.078	0.295	0.751
5 obj	0.175	0.761	1.901
10 obj	0.447	2.474	7.268

Table 3. Times for MOA$^*_{0.1}$ on the shortest path problem.

6 CONCLUSION

We have proposed two approximation algorithms for multiobjective search. The first one is an FPTAS which requires that an upper bound on the length of a solution-path is known, while the second one does not provide guarantee on the worst case complexity, but performs better in practice without requiring any information on the length of solution-paths. Both algorithms outperform exact multiobjective search in times. Note that the approximate Pareto set can include dominated solutions (although close to optimality). An interesting research direction is therefore to look for algorithms able to compute approximate Pareto set including only non-dominated solutions. Another possible extension of this work is to study the use of ε-dominance to approximate more involved preference models.

REFERENCES

[1] E. Angel, E. Bampis, and A. Kononov, 'On the approximate tradeoff for bicriteria batching and parallel machine scheduling problems.', *Theor. Comput. Sci.*, **306**(1-3), 319–338, (2003).

[2] T. Erlebach, H. Kellerer, and U. Pferschy, 'Approximating multiobjective knapsack problems', *Manag. Science*, **48**(12), 1603–1612, (2002).

[3] K. Fujimura, 'Path planning with multiple objectives', *IEEE Robotics and Automation Magazine*, **3**(1), 33–38, (1996).

[4] M. Ghallab, 'A$_\varepsilon$: an efficient near admissible heuristic search algorithm', in *Proc. of the 8th IJCAI*, pp. 789–791, (1983).

[5] P. Hansen, 'Bicriterion path problems', in *Multicriteria Decision Making*, eds., G. Fandel and T. Gal, (1980).

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

[7] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, 'Combining convergence and diversity in evolutionary multiobjective optimization', *Evolutionary Computation*, **10**(3), 263–282, (2002).

[8] L. Mandow and J.-L. Pérez-de-la Cruz, 'A new approach to multiobjective A* search.', in *Proc. of the 19th IJCAI*, pp. 218–223, (2005).

[9] S. Martello and P. Toth, 'An upper bound for the zero-one knapsack problem and a branch and bound algorithm', *European J. of Operational Research*, **1**, 169–175, (1975).

[10] C. H. Papadimitriou and M. Yannakakis, 'On the approximability of trade-offs and optimal access of web sources', in *Proc. of the 41th IEEE Symp. on FOCS*, pp. 86–92, (2000).

[11] J. Pearl and J.H. Kim, 'Studies in semi-admissible heuristics', *IEEE Trans. on PAMI*, **4**(4), 392–400, (1982).

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

[13] G. Tsaggouris and C. Zaroliagis, 'Multiobjective optimization: Improved FPTAS for shortest paths and non-linear objectives with applications', in *Proc. of the 17th ISAAC*, pp. 389–398, (2006).

[14] A. Warburton, 'Approximation of pareto optima in multiple-objective shortest-path problems', *Operations Research*, **35**(1), 70–79, (1987).