Hyper-GRASP: A Hypervolume-Based Constructive Heuristic

Gonçalo Lopes*
galopes@dei.uc.pt
University of Coimbra, CISUC/LASI, DEI
Coimbra, Portugal

Luís Paquete
paquete@dei.uc.pt
University of Coimbra, CISUC/LASI, DEI
Coimbra, Portugal

Carlos Fonseca
cmfonsec@dei.uc.pt
University of Coimbra, CISUC/LASI, DEI
Coimbra, Portugal

Abstract

This article proposes a novel constructive heuristic approach, Hyper-GRASP, that integrates the hypervolume indicator and GRASP principles to solve multiobjective combinatorial optimization problems. The approach constructs solutions iteratively by generating and evaluating candidate extensions of the current partial solution, guided by an optimistic bound on the hypervolume contribution of the future complete solution. The most promising extension is selected from a pool according to a parameter α, which balances greediness and exploration during the solution construction phase. Throughout the procedure, only feasible and nondominated solutions are retained. The proposed approach is tested on the multiobjective knapsack problem and the biobjective minimum spanning tree problem to assess its performance. The results show that the best-performing Hyper-GRASP variant effectively approximates the Pareto front across various instances. Moreover, it can also outperform state-of-the-art exact algorithms for the multiobjective knapsack problem as the problem size increases and across various time budgets.

CCS Concepts

• **Computing methodologies** → **Randomized search**; • **Theory of computation** → **Algorithm design techniques**.

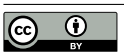
Keywords

multiobjective combinatorial optimization, constructive algorithm, randomized search, hypervolume indicator, GRASP

ACM Reference Format:
Gonçalo Lopes, Luís Paquete, and Carlos Fonseca. 2025. Hyper-GRASP: A Hypervolume-Based Constructive Heuristic. In *Foundations of Genetic Algorithms XVIII (FOGA '25), August 27–29, 2025, Leiden, Netherlands.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3729878.3746617

1 Introduction

In this work, we propose a new constructive heuristic approach, Hyper-GRASP, to solve Multiobjective Combinatorial Optimization (MCO) problems based on Greedy Randomized Adaptive Search Procedure (GRASP) principles and using the hypervolume indicator to guide the construction of solutions.



FOGA '25, Leiden, Netherlands

ACM ISBN 979-8-4007-1859-5/2025/08
https://doi.org/10.1145/3729878.3746617

GRASP has been widely used to solve single-objective optimization problems [18, 19]. It consists of two main phases, the *construction phase* and the *local search phase*, repeated over several iterations until a stopping condition is met. In the construction phase, a solution is built incrementally, where, at each step, the impact of adding an available element to the solution is evaluated according to a certain quality measure. The best possible solution extensions are stored in a *list of restricted candidates (RCL)* of size determined by a parameter α, defined *a priori*. From this list, one extension is randomly selected for the next constructive step. Once the solution is constructed, a local search starts, inducing several local modifications to the solution to find improvements. This process terminates when no further improvement is possible.

The Hyper-GRASP uses the same principles as GRASP, but extends it to MCO problems by using hypervolume information and taking into account the *best* set of nondominated points found so far. In particular, during the construction phase of Hyper-GRASP, the quality of a solution is evaluated in terms of the hypervolume contribution of its optimistic bound relative to the best nondominated solutions found so far.

To study the proposed approach, we performed an in-depth empirical study on two classical MCO problems: the Multiobjective Knapsack and the Biobjective Minimum Spanning Tree problem. We analysed the impact of different time budgets on its performance measured in terms of hypervolume indicator. Moreover, we compare it against exact algorithms that are based on similar principles, each terminating at the same time limits. Our results indicate that Hyper-GRASP consistently returns better approximations beyond a certain problem size, for the various time budgets considered in the experiments.

The article is organized as follows. Section 2 provides an overview of relevant concepts and definitions in multiobjective optimization, including the hypervolume indicator, hypervolume-based solution approaches and constructive heuristics for MCO problems. In Section 3, we introduce the proposed Hyper-GRASP approach under a more general framework for MCO problems. Sections 4 and 5 report the empirical study on Hyper-GRASP for the Multiobjective Knapsack problem and the Biobjective Minimum Spanning Tree problem, respectively. In Section 6, we present an empirical comparison of Hyper-GRASP with state-of-the-art algorithms for the Multiobjective Knapsack Problem under several time budgets. To conclude, in Section 7, we present our conclusions and discuss possible directions for further work.

2 Background

In the following, we introduce fundamental concepts in MCO and constructive heuristic methods, followed by an overview of the hypervolume indicator and hypervolume-based algorithms.

2.1 Multiobjective Combinatorial Optimization

In this article, we consider optimization problems of a combinatorial nature (see Ehrgott [16]), for which a solution can be seen as a subset of components chosen from a finite set. Let $E = \{e_1, \ldots, e_n\}$ be a finite set of n solution components and let $c : E \mapsto \mathbb{R}^m$ be an m-dimensional cost function of the elements in E. The meaning of E and c is problem-dependent. For example, in a knapsack problem, E represents the set of available objects and $c(e)$, for each $e \in E$, is a vector of profits. In a minimum spanning tree problem, E is the set of edges in a graph, and $c(e)$ is the vector of weights associated with an edge $e \in E$. A MCO problem is given by a feasible set $X \subset 2^E$ and a m-dimensional objective function $f : X \mapsto \mathbb{R}^m$. A *feasible* solution $x \in X$ is a subset of E. In this work, we consider sum objectives of the form

$$f^k(x) = \sum_{e \in X} c^k(e)$$

for $k = 1, \ldots, m$. In the context of our work, a *candidate* solution x' is also a subset of E, but it does not necessarily belong to X. It may be possible to reach a feasible solution from x' by adding further solution components from E.

In general, there exists no solution that is optimal for all functions f^k at once, since there is no canonical order of $\mathbb{R}^m$ for $m \geq 2$. We consider the Pareto concept of optimality, which is based on the component-wise order relations in $\mathbb{R}^m$ [17]. A solution $x \in X$ is called *efficient* if there exists no other solution $\bar{x} \in X$ that dominates it in the objective space. The set of all efficient solutions is called the *efficient set* and its image in the objective space is called the *Pareto front*. The approaches described in this article return a set of feasible solutions for a given MCO problem, whose image in the objective space approximates the Pareto front. We require that this approximation consists of *pairwise* nondominated points in the objective space, that is, every point is nondominated with respect to every other point in the set.

2.2 Constructive Heuristics for MCO

Certain problem-specific heuristics can provide optimal solutions to specific combinatorial optimization problems. For instance, Prim's algorithm solves the Minimum Spanning Tree problem to optimality in a greedy manner [40]. However, the multi-objective variants of such problems do not allow straightforward extensions of these constructive heuristics. A natural multi-objective extension of Prim's algorithm would involve selecting edges in a topologically ordered manner based on their cost vectors, provided no cycles are formed. However, depending on the topological order used, this approach may not return an efficient spanning tree [16]. In fact, optimal constructive heuristics for MCO problems are only known for very particular cases [23].

Even if optimality is relaxed, very few constructive approaches for MCO problems can be found in the literature, unlike the more extensively studied population and local search-based methods (see an overview in [22]). This is primarily due to the inherent difficulty of adequately covering diverse regions of the objective space. In the following, we briefly review some of the existing approaches.

GRASP-based approaches have been proposed for MCO problems. The most common technique in previous GRASP methods is the weighted combination of objectives to determine a search direction towards the Pareto front and to introduce randomness to diversify the construction of the solutions, which are later refined using local search techniques [3, 41, 45]. Furthermore, recent works have combined different GRASP and local search variants with other search strategies, such as path relinking [2, 36]. In particular, in [36], the authors consider multiple variants of the construction phase, such as optimizing each objective individually, sequentially, alternating or aggregating them in terms of weighted-sums. Then, each solution constructed is optimized using multiple local search variants, similarly to the construction phase. At last, for each pair of nondominated solutions found, a path-relinking step is applied.

ACO algorithms construct solutions through probabilistic decisions guided by pheromone information [34]. This pheromone information can be represented either by a single matrix encompassing all objectives or by multiple matrices, one for each objective, with decisions based on one matrix at a time or on an aggregation of the matrices using weights. Additionally, ACO algorithms can employ multiple colonies [27], where ants are divided into disjoint groups, each targeting a specific region of the Pareto front, or adopt population-based approaches [26, 30], where a set of candidate solutions is maintained and evolved, allowing for simultaneous exploration of multiple solutions. See [14] for an overview of ACO approaches to MCO problems.

Beam Search has also been used for MCO problems. Following the principles of Branch and Bound using a breadth-first search strategy, it maintains a set of candidate solutions at each tree level, selecting a subset of a given size based on the ϵ-indicator [39]. An extension of this approach is discussed in [5].

2.3 Hypervolume Indicator

The *hypervolume indicator* was first introduced in [50] as an indicator for the evaluation of the performance of EMO approaches. In the context of this work, the hypervolume indicator of a set $R \subset \mathbb{R}^m$ of nondominated points is understood as follows (w.l.o.g. we assume m maximizing objective functions):

$$H(R) = vol\left(\left\{p \in \mathbb{Z}^m \mid \exists\, z \in R : r \leqq p \leqq z\right\}\right)$$

where $\leqq$ denotes the weak-componentwise ordering, *vol* denotes the Lebesgue measure and r is a reference point, with $r \in \mathbb{R}^m$ and $r \leqq z$ for every $z \in R$. This indicator has interesting properties: it is monotonically increasing as nondominated points are added to R and the Pareto front of a given MCO problem has maximal hypervolume value [51].

For this work, it is important to measure the specific contribution of a point $z \in \mathbb{R}^m$ to the overall hypervolume of a set R of nondominated points, known as the *hypervolume contribution* of z with respect to R, which is defined as follows:

$$H(z, R) = H(\{z\} \cup R) - H(R).$$

Figure 1 illustrates the hypervolume indicator of a representation set $R = \{z_1, z_2, z_3, z_5\}$ given a reference point r (grey area) and the hypervolume contribution of point z_4 with respect to R (red region).

The hypervolume indicator is commonly used in the selection process of EMO algorithms. Most hypervolume-based EMO approaches, including SMS-EMOA [10], HypE [4], IBEA [49], and SIBEA [48] use hypervolume to discard (accept) solutions from (to)

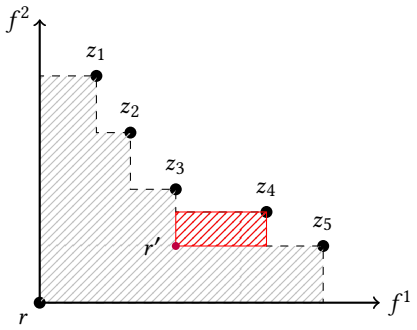


Figure 1: Illustration of the hypervolume indicator of a set $R = \{z_1, z_2, z_3, z_5\}$ and the contribution of a point z_4 with respect to R and r'.

Algorithm 1: Hyper-GRASP Algorithm

Input: α
$S \leftarrow \emptyset$
while *stopping condition is not met* **do**
  $x \leftarrow emptySolution()$
  $C \leftarrow \{x\}$
  while C *is not empty* **do**
    $RCL \leftarrow selectCandidates(C, \alpha)$
    $x' \leftarrow chooseCandidate(RCL)$
    $x \leftarrow x'$
    $C \leftarrow generateCandidates(x, S)$
  if $isFeasible(x) = True$ **then**
    if $computeContribution(x, S) > 0$ **then**
      $S \leftarrow S \cup \{x\}$
      $S \leftarrow update(S)$
return S

the population. In the context of MCO problems, the search process is typically based on modifying solution components through specific operators, such as mutation and crossover. In [1], the hypervolume indicator is used in the selection step of a population-based ACO approach, but not as heuristic information to guide the construction of new individuals. An overview on hypervolume-based EMO approaches can be found in [43].

More recently, the hypervolume indicator has been used in the context of exact objective-based approaches [38], as well as in branch-and-bound methods [7, 29] for MCO problems. In fact, some aspects of our approach are inspired by the latter, particularly the use of optimistic bounds to guide the search process. Furthermore, the hypervolume indicator has also been used in the selection step of anytime behaviour of the Pareto Local Search variants [15], and fitness assignment within local search and variants [6, 47].

3 Hyper-GRASP

The Hyper-GRASP is a new constructive heuristic designed to solve MCO problems using hypervolume to guide the construction of solutions. Similarly to GRASP, each solution is constructed iteratively by selecting solution components associated with the problem definition. The algorithm requires a parameter α, which controls the greediness of the construction phase.

Algorithm 1 describes Hyper-GRASP, which outputs a set S containing the *best* feasible solutions found until the stopping condition is met. During a run of Hyper-GRASP, only feasible solutions that are pairwise nondominated in the objective space are maintained in S. The inner loop of Algorithm 1 constructs a candidate solution x, which is initially empty, meaning it contains no solution components. At each iteration, the *generateCandidates* procedure generates extensions of x by adding each available solution component without violating any problem constraints. These extensions are stored in a set C of candidate solutions. If C is empty – indicating that no extensions of x were generated – the inner loop terminates, and x is tested for potential insertion into S. Otherwise, the procedure *selectCandidates* identifies a pool of *promising* candidates from C based on the current archive, S, and on the parameter α, forming the restricted candidate list (*RCL*). Then, the *chooseCandidate* procedure selects the next candidate solution x' from *RCL* for exploration.

Once the inner loop terminates, if the constructed solution x is feasible and its hypervolume contribution relative to S is positive – meaning x is pairwise nondominated with respect to all elements in S in the objective space – then x is added to S. After adding x to S, the set S is updated, that is, the dominated solutions present in S are removed. The approach restarts to generate a new solution using the information stored in S, until the stopping condition is met.

Procedures *generateCandidates* and *selectCandidates* require access to solution quality information for each candidate solution, even if the solution is not feasible. In our approach, a candidate solution x is evaluated based on an optimistic bound $b(x)$ in the objective space, which is guaranteed to be *at least as good as* any feasible extension of x. The *solution quality* of x is then determined by the hypervolume contribution of its optimistic bound relative to S. If S is empty, the hypervolume contribution is computed relative to a *reference point* that is guaranteed to be dominated by any element of the Pareto front. A larger contribution indicates a more promising candidate solution.

Figure 2 shows three candidate solutions evaluated in terms of the hypervolume contribution of their optimistic bound ($b(x_1)$, $b(x_2)$ and $b(x_3)$) with respect to a set of four nondominated points (z_1, z_2, z_3, and z_4) for two maximizing objectives. The optimistic bound $b(x_1)$ has a larger hypervolume contribution than $b(x_2)$ and $b(x_3)$ and, therefore, solution x_1 is more promising. Note that $b(x_3)$ has no positive hypervolume contribution, unlike $b(x_1)$ and $b(x_2)$. Therefore, there is no need to consider extensions of x_3.

In our approach, the optimistic bounds and their corresponding hypervolume contributions are calculated for each extension of a candidate solution x generated within the procedure *generateCandidates*, and only extensions with positive hypervolume contribution are inserted into C. Therefore, the method for computing the hypervolume contribution must be carefully chosen. For a given set R of pairwise nondominated points, the hypervolume contribution

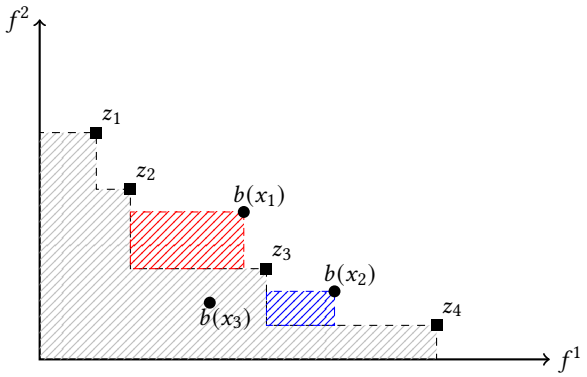


Figure 2: Hypervolume contribution of the optimistic bound of three candidate solutions x_1, x_2 and x_3 with respect to a set of nondominated points.

can be computed in $O(\log |R|)$ time for $m = 2$ using a dimensional sweep approach by maintaining an archive of the solutions sorted based on one of the objective values and in $O(|R|)$ time using HV3D$^+$-U algorithm [24] for $m = 3$. For $m > 3$, the WFG algorithm [46] can be used, which takes $O(|R|^{m/2} \log |R|)$ time. See [25] for a discussion about algorithms for computing hypervolume contribution.

Procedure *selectCandidates* uses parameter α to define which solutions should be considered in *RCL*. Let h_{min} and h_{max} represent the lowest and highest hypervolume contributions of the optimistic bounds with respect to S among all elements in C. For a given α, the candidate solutions in C that fall into the interval $[h_{max} - \alpha \cdot (h_{max} - h_{min}), h_{max}]$ are selected to *RCL*. Parameter α defines how greedy the optimization process should be. Hyper-GRASP can adapt its behaviour by favouring greedier selections with low α, or opting for higher α to encourage diversity and avoid premature convergence in suboptimal regions.

We finally note that Hyper-GRASP shares similar principles with indicator-based Branch-and-Bound approaches presented in [7, 29], as both use an optimistic bound based on the hypervolume indicator to guide the selection of the next candidate solution to explore.

4 Empirical Study on the Multiobjective Knapsack Problem

This section presents an in-depth experimental analysis of Hyper-GRASP on the Multiobjective Knapsack problem, with two, three, and four objectives. Specifically, it analyses the impact of parameter α on the performance of Hyper-GRASP and its relationship with particular instance features, such as problem size and degree of correlation between the objectives.

In our experimental analysis, two distinct variants of Hyper-GRASP are also explored to assess different aspects of the approach. The first variant, `Hyper-GRASP-OUT`, calculates the hypervolume contribution of the optimistic bound solely with respect to the reference point of each instance. This would allow us to measure the impact of using set S in the search process of Hyper-GRASP. The second variant, `Hyper-GRASP-GREEDY`, adopts a deterministic greedy approach by setting the parameter α to 0, selecting the candidate that maximizes the hypervolume contribution with respect to S.

4.1 Multiobjective Knapsack Problem

Consider a set of items, $\{1, \ldots, n\}$, each item j with a weight w_j, a m-dimensional profit $p_j = (p_j^1, p_j^2, \ldots, p_j^m)$, and a finite capacity W. The Multiobjective Knapsack (MOK) problem aims to determine which items to include in the knapsack such that the capacity constraint is not exceeded and the m objective profits are *maximized*. Several exact algorithms based on dynamic programming have been proposed to solve the bi-objective version of this problem [9, 21].

In the context of Hyper-GRASP, the solution components for the MOK problem correspond to the items, and a solution is defined by a subset of these items. A candidate solution is feasible if its total weight does not exceed capacity W.

An optimistic (upper) bound for a candidate solution x can be obtained by applying Dantzig's greedy algorithm [13] to each objective function, which allows fractional items to be selected. For each objective i, the items not already included in solution x are processed in descending order of their profit-to-weight ratio. Then, Dantzig's algorithm iteratively adds items into x following this ordering, and stopping when adding the next item j would exceed the total weight capacity W. In that case, only the fraction of item j that fits within the remaining capacity is included in x. Note that the resulting solution is not feasible, in general. Let ub^i denote the total profit of the resulting solution for objective i. By repeating the procedure above for each objective, we obtain an upper bound:

$$b(x) = \left(ub^1, \ldots, ub^m\right).$$

The upper bound computation can be a bottleneck. To address this, an efficient implementation precomputes item orderings and positions for each objective based on profit-to-weight ratios. For each objective, the algorithm greedily simulates filling the knapsack, summing values of fully included items up to a break point and adding a fractional value for the next item. When updating the bound due to adding an item to x, the remaining capacity is adjusted, and the partial bound is recalculated, only if the item is after the break point. Similarly, when an item is skipped, the algorithm checks if it was included before the break point; if so, its value is subtracted, the remaining capacity is adjusted, and the break point shifts to include the next eligible items. By manipulating precomputed data and performing incremental updates, this approach minimizes recomputation, reducing the computational overhead.

For the evaluation of the hypervolume contribution, a reference point is needed e.g. when S is empty. For the MOK problem, it can be set to $(0, \ldots, 0)$.

4.2 Experimental Setup

The algorithms were implemented in C++ and compiled with g++ version 11.4.0, with flag `-O3`. The implementation of Hyper-GRASP is available at [33]. Each variant was executed 30 times with different random seeds on the same instance, with a cut-off running time limit of 300 seconds. The results reported represent the aggregated performance across these independent runs. The experiments were

conducted on a machine with Linux Debian 6.1.76-1 Operating System (64-bit), an Intel Xeon(R) Silver(R) 4210R processor running at 2.4 GHz (nominal frequency) with 10 cores, 20 threads each, and 256GB RAM. In order to understand the effect of parameter α on the Hyper-GRASP variants, we considered $\alpha \in \{0.05, 0.1, 0.2, 0.5\}$.

For the MOK problem with $m = 2$, the instances used are those described by Bazgan et al. [9]. Each weight and profit value of each instance is generated randomly according to a uniform distribution within the range $[1, 1000]$. Three types of instances are defined: Type A, with weights and profits generated independently of each other and uniformly distributed; Types B and C, which feature a positive and negative correlation between profits, respectively, with uniformly distributed weights. Only instances that require a much larger computational effort than our cut-off limit by exact algorithms reported in [21] are considered in our experimental analysis. We consider instances with 500 and 700 items for Type A, 1000, 2000, and 4000 items for Type B, and 300 and 500 items for Type C. For $m \in \{3, 4\}$ the instances correspond to Type A considering a range $[1, 300]$ for both weight and profit. The range is smaller than for $m = 2$ in order to avoid overflow in the hypervolume computations. For $m = 3$, we considered 50, 100 and 150 items, whereas for $m = 4$, we considered 50 and 80 items. For each instance, the capacity constraint is set to half of the sum of all the weights. Moreover, the Pareto front is known for all instances.

4.3 Experimental Results

The performance of each Hyper-GRASP variant is assessed based on: (i) the percentage of returned solutions that belong to the Pareto front, and (ii) the *hypervolume* approximation ratio with respect to the Pareto front. The latter is computed by dividing the hypervolume of the returned solution set by the hypervolume of the region dominated by the Pareto front, taking the nadir point as the reference point.

The results obtained on the MOK problem are shown in Tables 1, 2, and 3, for 2, 3, and 4 objectives, respectively. For each variant and α value, and for each instance type (column *Type*) and size (column N), the hypervolume approximation ratio (column HA_R), the number of solutions found (column $|S|$), and the percentage of solutions in the Pareto front (column $|N_S|_{\%}$), averaged over all runs and all instances, is reported. Moreover, for each variant and every instance type and size, the highest HA_R value is underlined, and the highest HA_R value across all variants for each instance type and size can be found in bold. The time taken by `Hyper-GRASP-GREEDY` is reported in column t. Results for $\alpha = 0.5$ are not reported in the tables since they consistently yielded low approximation ratios, making them less relevant for the discussion.

Table 1 indicates that `Hyper-GRASP-GREEDY` is very computationally efficient but finds few solutions, which rarely belong to the Pareto front, and with a low approximation ratio. Comparing `Hyper-GRASP-GREEDY` and `Hyper-GRASP-OUT`, it is noticeable that the former achieves better or comparable approximation ratio values. `Hyper-GRASP` presents the best overall performance, suggesting that the hypervolume information provided by S significantly improves the search process.

For all bi-objective instances, the best performance is achieved with a small α value, except for `Hyper-GRASP-OUT` in Type A and C instances, where a higher α value shows to be more effective. For three- and four-objective cases (see Tables 2 and 3), all variants achieve the best performance with a higher α value. Notably, a larger proportion of the solutions found belong to the Pareto front compared to the bi-objective case, likely due to the smaller instance sizes. This has also been observed on smaller bi-objective instances, not reported here.

Figure 3 presents the median of the *empirical attainment function* (EAF) [12] [35] obtained by `Hyper-GRASP` variant for different values of α, for one instance of each type in the bi-objective case. We include the results obtained with $\alpha = 0.5$ for reference. A worse performance can be observed for $\alpha = 0.5$ (green line). For the other values of α, the differences are generally small, except for the Type B instance, which corroborates the results shown in Table 1 for the bi-objective case.

5 Empirical Study on the Multiobjective Minimum Spanning Tree Problem

In the section, we describe an analysis of Hyper-GRASP to the Multiobjective Minimum Spanning Tree problem. Only two objectives were considered since no Pareto front is available for more than two objectives. We used the same Hyper-GRASP variants and computational environment as presented in the previous section for the MOK problem.

5.1 The Multiobjective Spanning Tree Problem

Consider a graph $G = (V, E)$, where V is a set of n vertices and E is a set of edges, each edge $e \in E$ has an m-dimensional cost $c_e = (c_e^1, c_e^2, \ldots, c_e^m)$ representing costs for m different objectives. The Multiobjective Minimum Spanning Tree (MOMST) Problem aims to find a spanning tree $T \subseteq E$ that connects all vertices in V while *minimizing* the m cost objectives. See an in-depth experimental analysis of several exact algorithms for this problem with $m = 2$ in [20].

In our approach, the solution components for the MOMST problem correspond to the set of edges, and a candidate solution is a subset of these edges. A candidate solution is feasible only if it forms a spanning tree. It is important to note that any candidate solution containing a cycle cannot lead to a feasible solution. This can be efficiently verified within the *generateCandidates* procedure by using a union-find data structure tied to x.

An optimistic (lower) bound for each candidate solution x can be computed using Kruskal's Algorithm [32] with the edges that do not belong to x. For each objective, the remaining edges are processed in non-decreasing order of their costs. Then, iteratively add edges to x ensuring that no cycles are created until a spanning tree is obtained. Let lb^i denote the total cost of the resulting solution for objective i. By applying the procedure above for each objective, we obtain the following optimistic bound:

$$b(x) = \left(lb^1, \ldots, lb^m\right).$$

For the evaluation of the hypervolume contribution, the reference point used is the *nadir point*, which can be obtained by solving lexicographically each objective independently, for the case $m = 2$.

Type	N	Hyper-GRASP-GREEDY				α	Hyper-GRASP-OUT			Hyper-GRASP		
		t	HA_R	$\|S\|$	$\|N_S\|_\%$		HA_R	$\|S\|$	$\|N_S\|_\%$	HA_R	$\|S\|$	$\|N_S\|_\%$
A	500	0.2	0.786	14.6	<0.1	0.05	0.669	31.2	<0.1	**0.961**	152.7	<0.1
						0.1	0.688	37.3	<0.1	0.958	161.8	<0.1
						0.2	0.697	33.7	<0.1	0.947	161.1	<0.1
	700	0.6	0.757	19.5	<0.1	0.05	0.653	31.4	<0.1	**0.959**	153.9	<0.1
						0.1	0.666	30.5	<0.1	0.954	166.2	<0.1
						0.2	0.671	29.6	<0.1	0.919	147.4	<0.1
B	1000	0.1	0.467	1.3	<0.1	0.05	0.503	6.6	<0.1	**0.855**	17.3	0.4
						0.1	<0.1	4.0	<0.1	0.800	19.6	0.9
						0.2	<0.1	1.6	<0.1	0.719	19.7	1.5
	2000	0.5	0.547	2.0	<0.1	0.05	0.337	5.8	<0.1	**0.828**	26.7	<0.1
						0.1	<0.1	3.0	<0.1	0.688	25.4	0.2
						0.2	<0.1	1.7	<0.1	<0.1	7.5	<0.1
	4000	2.5	0.571	2.1	<0.1	0.05	0.261	4.5	<0.1	**0.755**	36.1	<0.1
						0.1	<0.1	2.6	<0.1	<0.1	7.3	<0.1
						0.2	<0.1	1.5	<0.1	<0.1	4.5	<0.1
C	300	<0.1	0.696	4.6	<0.1	0.05	0.670	36.6	<0.1	0.983	210.4	0.2
						0.1	0.693	85.4	0.2	**0.983**	269.8	0.3
						0.2	0.719	119.9	0.1	0.980	306.2	0.2
	500	0.1	0.705	6.8	<0.1	0.05	0.666	60.8	<0.1	**0.985**	291.9	<0.1
						0.1	0.677	98.4	<0.1	0.983	312.9	<0.1
						0.2	0.693	104.1	<0.1	0.978	320.8	<0.1

Table 1: Experimental results obtained on the MOK problem with two objectives (Type A, B and C)

Type	N	Hyper-GRASP-GREEDY				α	Hyper-GRASP-OUT			Hyper-GRASP		
		t	HA_R	$\|S\|$	$\|N_S\|_\%$		HA_R	$\|S\|$	$\|N_S\|_\%$	HA_R	$\|S\|$	$\|N_S\|_\%$
A	50	<0.1	0.529	2.9	0.2	0.05	0.493	3.1	0.2	0.746	12.3	1.2
						0.1	0.583	14.2	0.8	0.888	45.2	4.9
						0.2	0.752	77.4	7.9	**0.943**	112.6	10.7
	100	<0.1	0.500	3.6	<0.1	0.05	0.516	19.2	<0.1	0.861	104.5	0.3
						0.1	0.608	91.8	0.4	0.902	226.7	0.6
						0.2	0.702	342.4	1.1	**0.931**	508.1	1.5
	150	<0.1	0.511	4.7	<0.1	0.05	0.494	29.1	<0.1	0.886	271.4	0.1
						0.1	0.568	135.2	0.1	0.902	456.3	0.2
						0.2	0.654	480.4	0.3	**0.915**	738.5	0.2

Table 2: Experimental results obtained on the MOK problem with three objectives (Type A)

Type	N	Hyper-GRASP-GREEDY				α	Hyper-GRASP-OUT			Hyper-GRASP		
		t	HA_R	$\|S\|$	$\|N_S\|_\%$		HA_R	$\|S\|$	$\|N_S\|_\%$	HA_R	$\|S\|$	$\|N_S\|_\%$
A	50	<0.1	0.418	2.9	0.1	0.05	0.393	4.7	0.1	0.626	21.3	0.4
						0.1	0.473	19.6	0.2	0.769	51.1	1.0
						0.2	0.632	133.8	2.1	**0.843**	95.2	2.0
	80	<0.1	0.375	2.9	<0.1	0.05	0.385	7.4	<0.1	0.685	52.0	0.1
						0.1	0.466	41.7	0.1	0.777	91.8	0.2
						0.2	0.612	452.4	0.8	**0.796**	133.6	0.3

Table 3: Experimental results obtained on the MOK problem with four objectives (Type A)

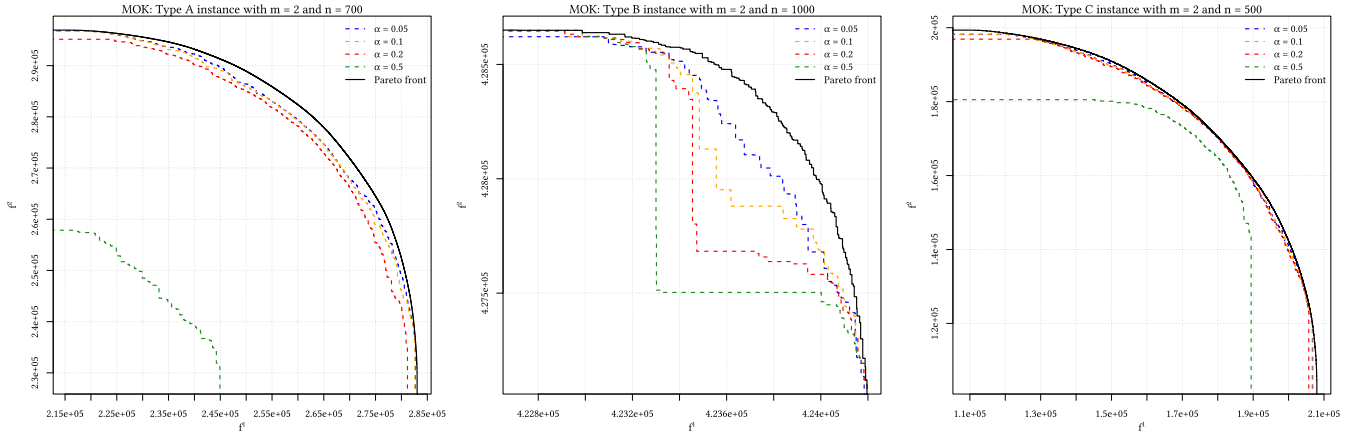


Figure 3: Pareto front and median of the empirical attainment function for $\alpha \in \{0.05, 0.1, 0.2, 0.5\}$ for individual instances of Type A (left), B (middle) and C (right) for the MOK problem.

Type	N	Hyper-GRASP-GREEDY				α	Hyper-GRASP-OUT			Hyper-GRASP		
		t	HA_R	$\lvert S\rvert$	$\lvert N_S\rvert_\%$		HA_R	$\lvert S\rvert$	$\lvert N_S\rvert_\%$	HA_R	$\lvert S\rvert$	$\lvert N_S\rvert_\%$
conc	30	<0.1	0.816	3.0	2.0	0.05	0.746	12.3	0.0	0.936	15.6	5.8
						0.1	0.544	11.2	0.0	**0.944**	20.3	7.4
						0.2	0.394	12.1	0.0	0.926	23.2	2.1
	50	0.2	0.829	5.5	0.6	0.05	0.710	13.1	0.0	**0.956**	41.0	3.3
						0.10	0.485	10.5	0.0	0.951	38.9	0.7
						0.20	0.219	9.1	0.0	0.722	30.3	0.0
	100	3.7	0.838	7.2	0.4	0.05	0.594	6.2	0.0	**0.924**	29.1	0.0
						0.10	0.268	5.3	0.0	0.602	17.4	0.0
						0.20	0.038	3.7	0.0	0.092	11.3	0.0
pos	30	<0.1	0.702	2.8	0.5	0.05	**0.811**	25.7	16.3	0.785	13.1	1.4
						0.10	0.794	19.8	6.1	0.805	16.5	2.2
						0.20	0.589	9.5	0.9	0.788	15.9	1.2
	50	0.2	0.644	2.9	0.1	0.05	0.721	16.9	0.2	**0.830**	25.9	0.8
						0.10	0.573	9.6	0.0	0.801	22.9	0.2
						0.20	0.236	6.2	0.0	0.513	17.0	0.0
	100	4.1	0.681	4.1	<0.1	0.05	0.550	7.1	0.0	**0.765**	21.5	0.0
						0.10	0.306	4.7	0.0	0.487	14.3	0.0
						0.20	0.110	4.0	0.0	0.173	12.0	0.0
neg	30	0.1	0.802	14.0	0.1	0.05	0.653	76.4	5.1	0.964	227.4	4.1
						0.10	0.670	72.0	1.7	**0.966**	230.6	2.6
						0.20	0.669	45.5	<0.1	0.945	169.2	0.1
	50	0.9	0.789	19.1	<0.1	0.05	0.638	62.8	0.2	**0.950**	285.4	0.2
						0.10	0.643	46.9	<0.1	0.944	260.3	<0.1
						0.20	0.621	33.2	0.0	0.887	174.2	0.0
	100	29.2	0.773	36.6	<0.1	0.05	0.615	28.3	0.0	**0.883**	65.8	0.0
						0.10	0.607	22.0	0.0	0.841	58.5	0.0
						0.20	0.566	18.5	0.0	0.726	44.9	0.0

Table 4: Experimental results obtained on the MOMST problem with two objectives (*conc*, *pos*, *neg*)

5.2 Experimental Setup

The instances used for the MOMST problem are those described in [20] for $m = 2$, which are based on the instances generated in [31] for the constrained version of this problem. The authors define three types of instances: Type *conc*, for which the set of nondominated points forms a *concave* shape in the objective space; type

pos and type *neg* for instances with positive and negative correlations between the weights of the edges, respectively. We considered complete (undirected) graphs with 30, 50, and 100 vertices for all instances types with a cost for each objective generated within the range $[1, 100]$.

5.3 Experimental Results

The results obtained on the MOMST problem with two objectives are shown in Table 4. As observed on the MOK problem, `Hyper-GRASP-GREEDY` finishes very quickly and finds few solutions. Similarly to the MOK problem, the latter achieves higher approximation ratios than `Hyper-GRASP-OUT`, except on *pos* instances. `Hyper-GRASP` deliver the best performance across most cases. However, for smaller *pos* instances, `Hyper-GRASP-OUT` variant produces results comparable to `Hyper-GRASP`. This is due to the limited number of solutions in these instances, unlike in the MOK problem, and the shape of their Pareto front, which concentrates solutions in a small region. As a result, few solutions can lead to high approximation ratios. This behaviour disappear as the problem size increases. In general, the best results are achieved with smaller α values, particularly $\alpha \in \{0.05, 0.1\}$.

Figure 4 presents the median of the EAF obtained from the runs of `Hyper-GRASP` variant, for one instance of each instance type for the MOMST problem. It is possible to observe that this variant is able to produce results that are very close to the Pareto front for all α values, except for $\alpha = 0.5$. As observed in the MOK problem, but more noticeable in this case, the approach exhibits some difficulty in reaching the extreme regions of the Pareto front.

6 Comparison with Exact Approaches Under Different Time Budgets

In this section, we analyse the performance of Hyper-GRASP in comparison with two state-of-the-art exact algorithms for MOK problem, a dynamic programming approach (B-DP) [9] and the Indicator-based Branch-and-Bound (I-BB) [29]. These approaches are very effective for small instance sizes and able to provide approximations to the Pareto front if terminated earlier as they ensure feasibility of stored solutions. In particular, the hypervolume-based I-BB using a Best-First Search strategy has shown to have a favorable anytime performance for this problem [29]. Although we expect the quality of these approximations to decrease as the time budget becomes more constrained, our main interest lies in identifying the problem size – if any– at which the best-performing Hyper-GRASP variant becomes preferable in terms of approximation quality under a given time budget.

6.1 Exact Approaches

B-DP Algorithm. Bazgan et al. [9] proposed a dynamic programming algorithm to solve the MOK problem for any number of objectives. The algorithm operates sequentially over n iterations, where n is the number of items. At each stage k, it generates a set S_k of (feasible) candidate solutions composed exclusively of the first k items. This set is updated iteratively by combining the previous set S_{k-1} with new candidate solutions formed by including the k-th item, while keeping the weight constraint satisfied. The algorithm uses three dominance relations to prune candidate solutions, retaining only non-dominated solutions in S_k. It should be noted that one of the dominance relations uses optimistic bounds for pruning candidate solutions: a candidate solution in S_k is discarded if its optimistic bound is dominated by some feasible extension of other candidate solution. At the end of iteration n, set S_n is guaranteed to contain the Pareto front. For efficiency reasons, only the profits and weight of each candidate solution are stored, rather than the solutions themselves. This approach outperformed the existing methods of its time in experimental comparisons. Some improvements are discussed in [11, 21] but only for the biobjective case.

Since only candidate solutions that do not violate the capacity constraint are stored in S_k, ensuring feasibility, the approach can terminate at an iteration $k < n$ to provide an approximation to the Pareto front. Still, the characterization of its anytime performance is not yet known.

I-BB Algorithm. Jesus et al. [29] proposed a branch-and-bound framework for MCO problems that uses quality indicators to guide the search process. Specifically, the framework selects the next candidate solution to extend based on the quality of its optimistic bound, measured using either the binary hypervolume or the binary ϵ-indicator. The authors conducted an empirical study comparing six selection strategies under two branching orders. Performance was evaluated on the MOK problem in terms of both the time to find the complete Pareto front and the anytime performance, measured by the hypervolume of the best set of feasible solutions found over time. The best results were obtained with the Best-First Selection (BeFS) strategy, particularly when combined with a random branching order. Among the tested configurations, BeFS using the binary hypervolume indicator (*Hypervolume BeFS*) showed excellent anytime performance for problems with two or three objectives. However, its performance degrades for higher-dimensional instances mainly due to the computational cost of hypervolume calculations.

6.2 Experimental Analysis

For this analysis, we followed the experimental setup described in Section 4.2, focusing only on Type A instances of the MOK problem. For the Hyper-GRASP approach, we set $\alpha = 0.2$, which was preferable for more than two objectives based on results presented in Section 4.3. The implementations of B-DP and Hypervolume BeFS are available online at [28]. To ensure a fair and rigorous comparison, all approaches use the same optimistic bounds as those mentioned in Section 4.1, computed in a similar way across methods.

Figure 5 presents the median EAF by Hyper-GRASP, B-DP and Hypervolume BeFS, across three different instances of the MOK problem with $m = 2$ and $n \in \{50, 200, 400\}$ and a time budget of 900 seconds. For $n = 50$, both B-DP and Hypervolume BeFS can compute the complete Pareto front (black line). Still, Hyper-GRASP returns an approximation that is very close to the Pareto front. For $n = 200$, B-DP is still able to find the complete Pareto front, but Hypervolume BeFS (purple dashed line) begins to show degraded performance, leaving some gaps in the objective space to be covered, while Hyper-GRASP provides a better approximation. For the largest instance considered, the contrast in performance becomes

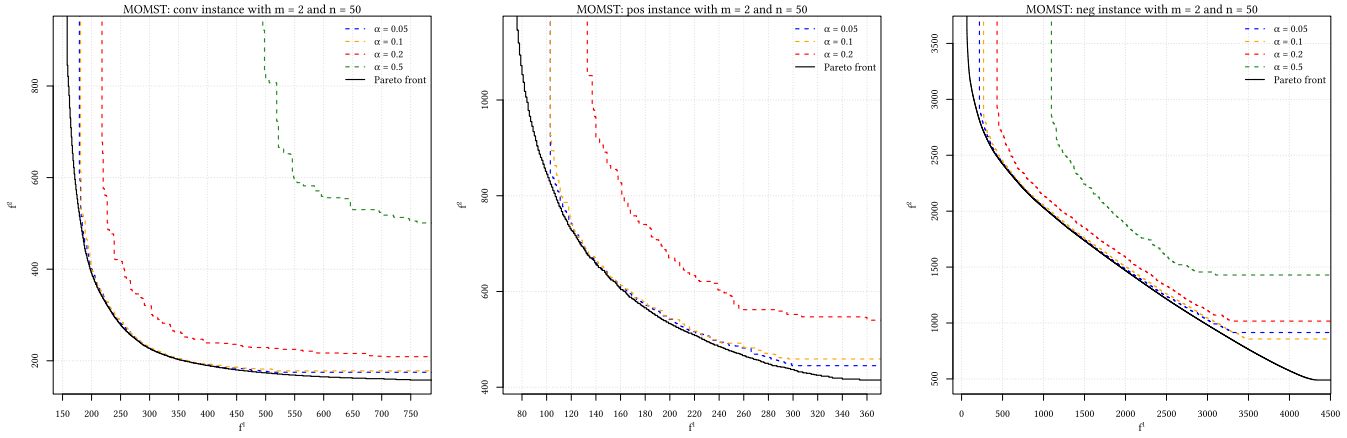


Figure 4: Pareto front and median of the empirical attainment function for $\alpha \in \{0.05, 0.1, 0.2, 0.5\}$ for individual instances of *conc* (left), *pos* (middle) and *neg* (right) for the MOMST problem.

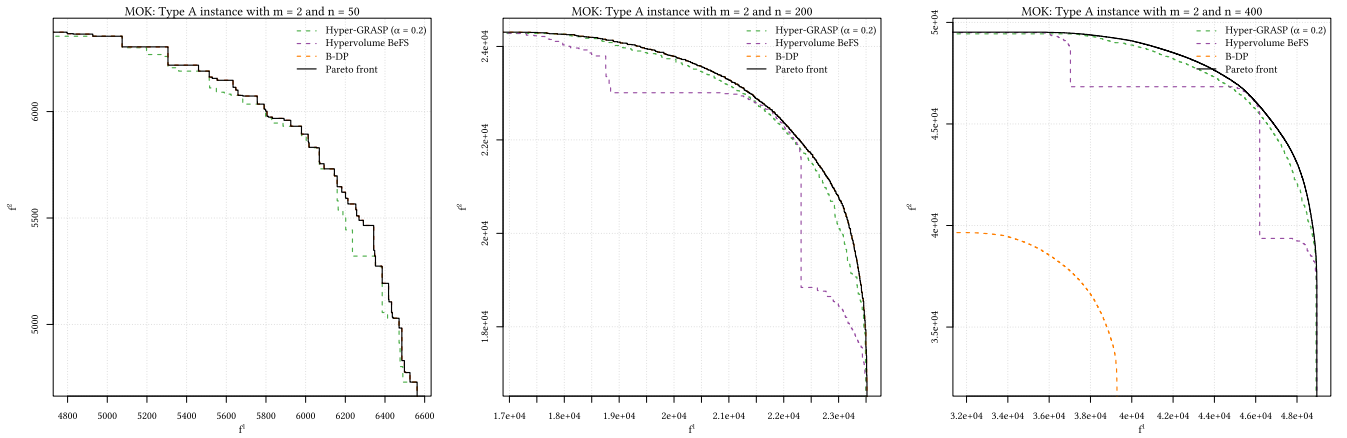


Figure 5: Pareto front and median of the empirical attainment function for the `Hyper-GRASP` ($\alpha = 0.2$), `B-DP` and `Hypervolume BeFS` for individual instances of Type A for the MOK problem.

even more pronounced. Hyper-GRASP maintains a tight approximation to the Pareto front, whereas B-DP can no longer compute the complete front and begins to fall short in approximation quality, even performing worse than Hypervolume BeFS. The results show that as instance size increases, Hyper-GRASP consistently provides high-quality approximations, often outperforming exact algorithms when they are terminated early: Hypervolume BeFS shows a decline in terms of objective space coverage for $n \geq 200$, and B-DP experiences a rapid drop in performance for $n \geq 400$.

To gain further insight into the efficiency of each algorithm, Figure 6 presents the performance of three approaches at three distinct time budgets, $t \in \{10, 100, 300, 900\}$, for different instance sizes and objectives. Each figure plots the hypervolume ratio that evaluates the quality of solution sets generated by each approach against the problem size (n) and different number of objectives (m). Note that the hypervolume of the solution sets is computed using the instance nadir point as the reference point, with a ratio of 1 indicating that the approach found the Pareto front, whereas a ratio of 0 indicates that there is no point in the solution set that dominates the reference point. Moreover, the shaded "fill" around each solid line indicates the maximum and minimum hypervolume ratios for each size n across all the runs.

The plots indicate that both Hyper-GRASP and Hypervolume BeFS are able to maintain high-quality approximations across several time budget scenarios. However, their performance declines for larger instances, with the degradation being more pronounced for Hypervolume BeFS. Hyper-GRASP begins to outperform Hypervolume BeFS at instance sizes between $n = 50$ and $n = 100$, even in the bi-objective case where $\alpha = 0.2$ was not the best choice. We also observe that the variation in the hypervolume ratio increases with the number of objectives. B-DP performs consistently well, surpassing both Hyper-GRASP and Hypervolume BeFS, but only up to a certain instance size, which varies with the number of objectives.

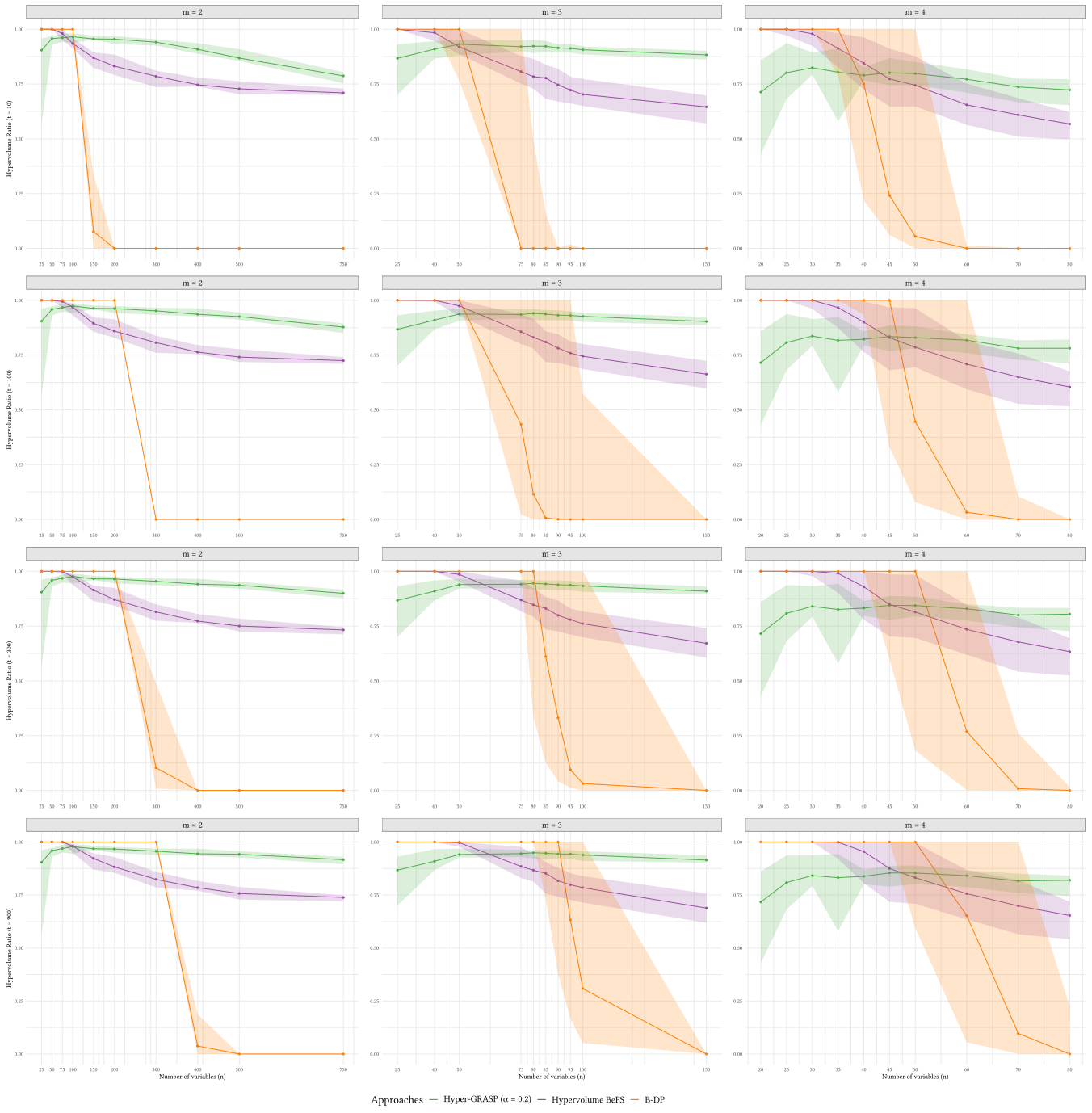


Figure 6: Hypervolume ratio for the Hyper-GRASP, Hypervolume BeFS and B-DP for instances of Type A for the MOK problem with $m \in \{2, 3, 4\}$, left to right, for $t \in \{10, 100, 300, 900\}$, top to bottom.

Beyond this threshold, its performance drops sharply, as shown in the left plot of Figure 5, accompanied by significant variability.

7 Conclusion

This work introduced Hyper-GRASP, a novel constructive heuristic for multiobjective combinatorial optimization problems that integrates GRASP principles with the hypervolume indicator to guide

the search process. Experimental results on the Multiobjective Knapsack problem and Multiobjective Minimum Spanning Tree problem highlighted the important role of using optimistic bounds in the construction phase. Compared to state-of-the-art exact algorithms, Hyper-GRASP showed better solution quality across various time budgets and with increasing instance sizes.

Although the best-performing variants achieved a high hypervolume approximation ratio, there remains room for further improvement. One possible enhancement is to incorporate a local search phase after executing Hyper-GRASP, such as a PLS-based approach [37]. Preliminary experiments, not reported in this work, suggest that this strategy can be highly effective for both MOC problems.

Another possible improvement is to tighten the optimistic bound, for example, by using an *optimistic bound set* [17] instead of a single point. This set can be composed of nondominated points obtained from suitable problem relaxations, when such formulations are computationally tractable. These bound sets have also been used within dynamic programming approaches for the Multiobjective Knapsack Problem to support pruning decisions [21]. Similarly, the set S can be initialized with optimal solutions obtained from a selected number of scalarized formulations of the MCO problem. For example, dichotomic search can be employed to identify all nondominated points that are optimal for some weighted-sum scalarization, across the entire space of possible weight combinations [44]. In the case of the Biobjective Minimum Spanning Tree problem, each scalarization can be efficiently solved using classical algorithms such as Prim's or Kruskal's. Furthermore, the number of such nondominated points is polynomially bounded with respect to the size of the graph [42].

An additional option is to explore dynamic tuning strategies for the parameter α, in order to adapt it throughout the search to balance exploration and exploitation more effectively. Finally, hypervolume contributions may not need to be computed exactly; instead, easily computable upper or lower bounds can be used as approximations. Similar strategies have been already incorporated into branch-and-bound algorithms for MCO problems [8].

A promising direction for future work involves extending the proposed framework to alternative search paradigms, such as Beam Search [39]. Beam Search can be viewed as a variant of Branch and Bound in which only the most promising candidate solutions are selected for extension, while the rest are pruned. Like Branch and Bound, Beam Search is a search tree algorithm that follows a constructive approach, that is, at each level of the tree, a decision is made regarding the inclusion of a solution component. Typically, Beam Search traverses the tree in a breadth-first manner and uses optimistic bounds to guide the selection of candidate solutions for the next level. The algorithm is parameterized by the *beam width* β, which defines the maximum number of candidate solutions retained at each level. In a similar fashion to Hyper-GRASP, the selection of the best β candidate solutions could be guided by their hypervolume contribution with respect to the best solution set found so far. This corresponds to solving a particular Hypervolume Subset Selection Problem, for which several approaches have been proposed (see an overview about approaches for this problem in [25]).

Acknowledgments

We would like to express our gratitude to the authors of [21] and [20] for generously providing the problem instances used in this study. This work was partially supported by the Portuguese Foundation for Science and Technology (FCT). The first author further acknowledges the FCT for Ph.D. fellowship 2021.07381.BD co-funded by the European Social Fund and by the State Budget of the Portuguese Ministry of Education and Science. This work is partially funded through national funds by FCT - Fundação para a Ciência e a Tecnologia, I.P., in the framework of the Project UIDB/00326/2025 and UIDP/00326/2025. This work is based upon work from COST Action Randomised Optimisation Algorithms Research Network (ROAR-NET), CA22137, supported by COST (European Cooperation in Science and Technology).

References

[1] Giacomo Acciarini, Dario Izzo, and Erwin Mooij. 2020. MHACO: a Multi-Objective Hypervolume-Based Ant Colony Optimizer for Space Trajectory Optimization. In *IEEE Congress on Evolutionary Computation, CEC 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 1–8. doi:10.1109/CEC48606.2020.9185694

[2] José Elias Claudio Arroyo, Michele dos Santos Soares, and Paula M. dos Santos. 2010. A GRASP heuristic with Path-Relinking for a bi-objective p-median problem. In *10th International Conference on Hybrid Intelligent Systems (HIS 2010), Atlanta, GA, USA, August 23-25, 2010*, Ashraf Saad and Ajith Abraham (Eds.). IEEE, 97–102. doi:10.1109/HIS.2010.5600091

[3] José Elias Claudio Arroyo, Pedro Sampaio Vieira, and Dalessandro Soares Vianna. 2008. A GRASP algorithm for the multi-criteria minimum spanning tree problem. *Ann. Oper. Res.* 159, 1 (2008), 125–133. doi:10.1007/S10479-007-0263-4

[4] Johannes Bader and Eckart Zitzler. 2011. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evol. Comput.* 19, 1 (2011), 45–76. doi:10.1162/EVCO_A_00009

[5] Thibaut Barthelemy, Sophie N. Parragh, Richard F. Hartl, and Fabien Tricoire. 2015. *Beam Search for integer multi-objective optimization.* Technical Report. Optimization on-line.

[6] Matthieu Basseur, Rong-Qiang Zeng, and Jin-Kao Hao. 2012. Hypervolume-based multi-objective local search. *Neural Comput. Appl.* 21, 8 (2012), 1917–1929. doi:10.1007/S00521-011-0588-4

[7] Julius Bauß, Sophie N. Parragh, and Michael Stiglmayr. 2024. On improvements of multi-objective branch and bound. *EURO J. Comput. Optim.* 12 (2024), 100099. doi:10.1016/J.EJCO.2024.100099

[8] Julius Bauß and Michael Stiglmayr. 2024. Augmenting bi-objective branch and bound by scalarization-based information. *Math. Methods Oper. Res.* 100, 1 (2024), 85–121. doi:10.1007/S00186-024-00854-3

[9] Cristina Bazgan, Hadrien Hugot, and Daniel Vanderpooten. 2009. Solving efficiently the 0-1 multi-objective knapsack problem. *Comput. Oper. Res.* 36, 1 (2009), 260–279. doi:10.1016/J.COR.2007.09.009

[10] Nicola Beume, Boris Naujoks, and Michael T. M. Emmerich. 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *Eur. J. Oper. Res.* 181, 3 (2007), 1653–1669. doi:10.1016/J.EJOR.2006.08.008

[11] Pedro Correia, Luís Paquete, and José Rui Figueira. 2018. Compressed data structures for bi-objective {0, 1}-knapsack problems. *Comput. Oper. Res.* 89 (2018), 82–93. doi:10.1016/J.COR.2017.08.008

[12] Viviane Grunert da Fonseca and Carlos M. Fonseca. 2010. The Attainment-Function Approach to Stochastic Multiobjective Optimizer Assessment and Comparison. In *Experimental Methods for the Analysis of Optimization Algorithms*, Thomas Bartz-Beielstein, Marco Chiarandini, Luís Paquete, and Mike Preuss (Eds.). Springer, 103–130. doi:10.1007/978-3-642-02538-9_5

[13] George B. Dantzig. 1957. Discrete-Variable Extremum Problems. *Operations Research* 5, 2 (1957), 266–288. doi:10.1287/opre.5.2.266

[14] Marco Dorigo and Thomas Stützle. 2004. *Ant colony optimization.* MIT Press.

[15] Jérémie Dubois-Lacoste, Manuel López-Ibáñez, and Thomas Stützle. 2015. Anytime Pareto local search. *Eur. J. Oper. Res.* 243, 2 (2015), 369–385. doi:10.1016/J.EJOR.2014.10.062

[16] Matthias Ehrgott. 2005. *Multicriteria Optimization (2. ed.).* Springer. doi:10.1007/3-540-27659-9

[17] Matthias Ehrgott and Xavier Gandibleux. 2007. Bound sets for biobjective combinatorial optimization problems. *Comput. Oper. Res.* 34, 9 (2007), 2674–2694. doi:10.1016/J.COR.2005.10.003

[18] Thomas A Feo and Mauricio G.C Resende. 1989. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters* 8, 2 (1989), 67–71. doi:10.1016/0167-6377(89)90002-3

[19] Thomas A. Feo and Mauricio G. C. Resende. 1995. Greedy Randomized Adaptive Search Procedures. *J. Glob. Optim.* 6, 2 (1995), 109–133. doi:10.1007/BF01096763

[20] Islame F. C. Fernandes, Elizabeth Ferreira Gouvea Goldbarg, Sílvia M. D. M. Maia, and Marco César Goldbarg. 2020. Empirical study of exact algorithms for the multi-objective spanning tree. *Comput. Optim. Appl.* 75, 2 (2020), 561–605. doi:10.1007/S10589-019-00154-1

[21] José Rui Figueira, Luís Paquete, Marco Simões, and Daniel Vanderpooten. 2013. Algorithmic improvements on dynamic programming for the bi-objective {0, 1} knapsack problem. *Comput. Optim. Appl.* 56, 1 (2013), 97–111. doi:10.1007/S10589-013-9551-X

[22] Xavier Gandibleux and Matthias Ehrgott. 2004. Approximate solution methods for multiobjective combinatorial optimization. *TOP* 12 (2004), 1–63.

[23] Jochen Gorski, Luís Paquete, and Fábio Pedrosa. 2012. Greedy algorithms for a class of knapsack problems with binary weights. *Computers and Operations Research* 39, 3 (2012), 498–511. doi:10.1016/J.COR.2011.02.010

[24] Andreia P. Guerreiro and Carlos M. Fonseca. 2018. Computing and Updating Hypervolume Contributions in Up to Four Dimensions. *IEEE Trans. Evol. Comput.* 22, 3 (2018), 449–463. doi:10.1109/TEVC.2017.2729550

[25] Andreia P. Guerreiro, Carlos M. Fonseca, and Luís Paquete. 2022. The Hypervolume Indicator: Computational Problems and Algorithms. *ACM Comput. Surv.* 54, 6 (2022), 119:1–119:42. doi:10.1145/3453474

[26] Michael Guntsch and Martin Middendorf. 2003. Solving Multi-criteria Optimization Problems with Population-Based ACO. In *Evolutionary Multi-Criterion Optimization, Second International Conference, EMO 2003, Faro, Portugal, April 8-11, 2003, Proceedings (Lecture Notes in Computer Science, Vol. 2632)*, Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele (Eds.). Springer, 464–478. doi:10.1007/3-540-36970-8_33

[27] Steffen Iredi, Daniel Merkle, and Martin Middendorf. 2001. Bi-Criterion Optimization with Multi Colony Ant Algorithms. In *Evolutionary Multi-Criterion Optimization, First International Conference, EMO 2001, Zurich, Switzerland, March 7-9, 2001, Proceedings (Lecture Notes in Computer Science, Vol. 1993)*, Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne (Eds.). Springer, 359–372. doi:10.1007/3-540-44719-9_25

[28] Alexandre D. Jesus. 2022. *mobkp.* doi:10.5281/zenodo.6857821

[29] Alexandre D. Jesus, Luís Paquete, Bilel Derbel, and Arnaud Liefooghe. 2021. On the design and anytime performance of indicator-based branch and bound for multi-objective combinatorial optimization. In *GECCO '21: Genetic and Evolutionary Computation Conference, Lille, France, July 10-14, 2021*, Francisco Chicano and Krzysztof Krawiec (Eds.). ACM, 234–242. doi:10.1145/3449639.3459360

[30] Joshua D. Knowles and David Corne. 2000. A new evolutionary approach to the degree-constrained minimum spanning tree problem. *IEEE Trans. Evol. Comput.* 4, 2 (2000), 125–134. doi:10.1109/4235.850653

[31] Joshua D. Knowles and David W. Corne. 2001. Benchmark problem generators and results for the multiobjective degree-constrained minimum spanning tree problem. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation* (San Francisco, California) *(GECCO'01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 424–431.

[32] Joseph B. Kruskal. 1956. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proc. Amer. Math. Soc.* 7, 1 (1956), 48–50. http://www.jstor.org/stable/2033241

[33] Gonçalo Lopes. 2025. *gaplopes/hyper-grasp: v1.0.0.* doi:10.5281/zenodo.15838546

[34] Manuel López-Ibáñez, Luís Paquete, and Thomas Stützle. 2004. On the Design of ACO for the Biobjective Quadratic Assignment Problem. In *Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5 - 8, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3172)*, Marco Dorigo, Mauro Birattari, Christian Blum, Luca Maria Gambardella, Francesco Mondada, and Thomas Stützle (Eds.). Springer, 214–225. doi:10.1007/978-3-540-28646-2_19

[35] Manuel López-Ibáñez, Luís Paquete, and Thomas Stützle. 2010. Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization. In *Experimental Methods for the Analysis of Optimization Algorithms*, Thomas Bartz-Beielstein, Marco Chiarandini, Luís Paquete, and Mike Preuss (Eds.). Springer, 209–222. doi:10.1007/978-3-642-02538-9_9

[36] Rafael Martí, Vicente Campos, Mauricio G. C. Resende, and Abraham Duarte. 2015. Multiobjective GRASP with Path Relinking. *Eur. J. Oper. Res.* 240, 1 (2015), 54–71. doi:10.1016/J.EJOR.2014.06.042

[37] Luís Paquete, Tommaso Schiavinotto, and Thomas Stützle. 2007. On local optima in multiobjective combinatorial optimization problems. *Ann. Oper. Res.* 156, 1 (2007), 83–97. doi:10.1007/S10479-007-0230-0

[38] Luís Paquete, Britta Schulze, Michael Stiglmayr, and Ana C. Lourenço. 2022. Computing representations using hypervolume scalarizations. *Comput. Oper. Res.* 137 (2022), 105349. doi:10.1016/J.COR.2021.105349

[39] Aníbal Ponte, Luís Paquete, and José Rui Figueira. 2012. On Beam Search for Multicriteria Combinatorial Optimization Problems. In *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems - 9th International Conference, CPAIOR 2012, Nantes, France, May 28 - June1, 2012. Proceedings (Lecture Notes in Computer Science, Vol. 7298)*, Nicolas Beldiceanu, Narendra Jussien, and Eric Pinson (Eds.). Springer, 307–321. doi:10.1007/978-3-642-29828-8_20

[40] R. C. Prim. 1957. Shortest connection networks and some generalizations. *The Bell System Technical Journal* 36, 6 (1957), 1389–1401. doi:10.1002/j.1538-7305.1957.tb01515.x

[41] Alan P. Reynolds, David W. Corne, and Beatriz de la Iglesia. 2009. A multi-objective GRASP for rule selection. In *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009*, Franz Rothlauf (Ed.). ACM, 643–650. doi:10.1145/1569901.1569990

[42] Florian Seipp. 2013. *On Adjacency, Cardinality, and Partial Dominance in Discrete Multiple Objective Optimization.* Ph. D. Dissertation. Technische Universität Kaiserslauern.

[43] Ke Shang, Hisao Ishibuchi, Linjun He, and Lie Meng Pang. 2021. A Survey on the Hypervolume Indicator in Evolutionary Multiobjective Optimization. *IEEE Trans. Evol. Comput.* 25, 1 (2021), 1–20. doi:10.1109/TEVC.2020.3013290

[44] Ralph E. Steuer and Enguo U. Choo. 1983. An Interactive Weighted Tchebycheff Procedure for Multiple Objective Programming. *Mathematical Programming* 26, 3 (1983), 326–344. doi:10.1007/BF02591962

[45] Dalessandro Soares Vianna and José Elias Claudio Arroyo. 2004. A GRASP Algorithm for the Multi-Objective Knapsack Problem. In *XXIV International Conference of the Chilean Computer Science Society (SCCC 2004), 11-12 November 2004, Arica, Chile.* IEEE Computer Society, 69–75. doi:10.1109/QEST.2004.2

[46] Lyndon While, Lucas Bradstreet, and Luigi Barone. 2012. A Fast Way of Calculating Exact Hypervolumes. *IEEE Trans. Evol. Comput.* 16, 1 (2012), 86–95. doi:10.1109/TEVC.2010.2077298

[47] Rong-Qiang Zeng, Matthieu Basseur, and Jin-Kao Hao. 2013. Hypervolume-Based Multi-Objective Path Relinking Algorithm. In *Evolutionary Multi-Criterion Optimization - 7th International Conference, EMO 2013, Sheffield, UK, March 19-22, 2013. Proceedings (Lecture Notes in Computer Science, Vol. 7811)*, Robin C. Purshouse, Peter J. Fleming, Carlos M. Fonseca, Salvatore Greco, and Jane Shaw (Eds.). Springer, 185–199. doi:10.1007/978-3-642-37140-0_17

[48] Eckart Zitzler, Dimo Brockhoff, and Lothar Thiele. 2007. The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration. In *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007, Matsushima, Japan, March 5-8, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4403)*, Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata (Eds.). Springer, 862–876. doi:10.1007/978-3-540-70928-2_64

[49] Eckart Zitzler and Simon Künzli. 2004. Indicator-Based Selection in Multiobjective Search. In *Parallel Problem Solving from Nature - PPSN VIII, 8th International Conference, Birmingham, UK, September 18-22, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3242)*, Xin Yao, Edmund K. Burke, José Antonio Lozano, Jim Smith, Juan Julián Merelo Guervós, John A. Bullinaria, Jonathan E. Rowe, Peter Tiňo, Ata Kabán, and Hans-Paul Schwefel (Eds.). Springer, 832–842. doi:10.1007/978-3-540-30217-9_84

[50] Eckart Zitzler and Lothar Thiele. 1998. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Parallel Problem Solving from Nature - PPSN V, 5th International Conference, Amsterdam, The Netherlands, September 27-30, 1998, Proceedings (Lecture Notes in Computer Science, Vol. 1498)*, A. E. Eiben, Thomas Bäck, Marc Schoenauer, and Hans-Paul Schwefel (Eds.). Springer, 292–304. doi:10.1007/BFb0056872

[51] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. Evol. Comput.* 7, 2 (2003), 117–132. doi:10.1109/TEVC.2003.810758