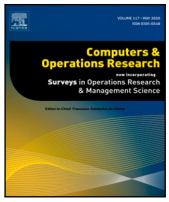
Computers & Operations Research 163 (2024) 106489

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Metaheuristics for the bi-objective resource-constrained project scheduling problem with time-dependent resource costs: An experimental comparison

Sofía Rodríguez-Ballesteros [a], Javier Alcaraz [a,b,*], Laura Anton-Sanchez [a,b]

[a] Centro de Investigación Operativa, Universidad Miguel Hernández, 03202 Elche, Alicante, Spain
[b] Departamento de Estadística, Matemáticas e Informática, Universidad Miguel Hernández, 03202 Elche, Alicante, Spain

ARTICLE INFO

Keywords:
Metaheuristic
Resource-constrained project scheduling problem
Multi-objective optimization
Pareto front
Performance indicator

ABSTRACT

The bi-objective resource-constrained project scheduling problem with time-dependent resource costs was recently introduced and consists of scheduling a set of activities subject to precedence and resource constraints, minimizing the makespan and the total cost for resource usage. Precisely, costs are determined by the resource being considered together with the time it is used. Although this generalization of the traditional resource-constrained project scheduling problem is rather recent, it has garnered substantial interest as it succeeds in meeting a wide range of real-world demands. In such a multi-objective context, solving the aforementioned problem poses a challenge, as both objectives conflict with each other, giving rise to a set of trade-off optimal solutions, commonly known as the Pareto front (PF). Given that many medium or large-sized instances of this problem cannot be solved by exact methods, the development of metaheuristics to find the PF is necessary. So far, only one metaheuristic had been developed to solve this problem. In this work we have implemented six additional multi-objective evolutionary algorithms (MOEAs), representing different paradigms, and subsequently, an exhaustive comparison of their performance has been carried out. In particular, all the compared MOEAs share the same encoding and main operators, focusing the comparison on the general algorithm framework rather than specific versions. Metaheuristic algorithms typically yield an approximation of the optimal PF, prompting the question of how to assess the quality of the obtained approximations. To this end, a computational and statistically supported study is conducted, choosing a benchmark of bi-criteria resource-constrained project scheduling problems and applying a set of performance measures to the solution sets obtained by each methodology. The results show that there are significant differences among the performance of the metaheuristics evaluated.

1. Introduction

Project scheduling is an intrinsic part of project management, wherein teamwork is employed to achieve all project goals within the given constraints. A well-known formalized problem in this field is the resource-constrained project scheduling problem (RCPSP) consisting of scheduling a set of activities subject to precedence and resource constraints. Typically, the objective of the RCPSP is to minimize the makespan, i.e., the completion time of the project in a schedule. A first discussion of the ongoing problem can be found in Blazewicz et al. (1983), where the RCPSP was shown to belong to the strongly NP-hard problems. A pioneer mathematical model was introduced in Pritsker et al. (1969). Since the 1990s, countless generalizations of the standard RCPSP have been developed, aiming to capture manifold realistic situations needed for successful practical applications. A comprehensive review of the literature, including multiple problem variants, is available in Hartmann and Briskorn (2022). This work furnishes an up-to-date account on Hartmann and Briskorn (2010), offering an overview of the current state-of-the-art contributions to the RCPSP literature. Both of the above reviews focus on deterministic strategies. The stochastic approach can be found in Neumann (1990) and Herroelen and Leus (2005). Regarding complementary surveys reported on the RCPSP, Hartmann and Briskorn (2022) discuss the extensive literature on the multi-mode setting (see, e.g., Florez-Perez et al., 2013; Qi et al., 2014; Coughlan et al., 2015).

In order to address more realistic scenarios that may occur in project scheduling, a multi-objective framework is adopted in this work. The existing literature in the presence of multiple objectives is addressed in several survey papers (see, e.g., Ballestín and Blanco, 2015; Habibi et al., 2018; Hartmann and Briskorn, 2022). Specifically, we focus on the resource-constrained project scheduling problem with time-dependent resource costs, henceforth RCPSP_TDRC, introduced

* Corresponding author at: Centro de Investigación Operativa, Universidad Miguel Hernández, 03202 Elche, Alicante, Spain.
E-mail addresses: sofia.rodriguezb@umh.es (S. Rodríguez-Ballesteros), jalcaraz@umh.es (J. Alcaraz), l.anton@umh.es (L. Anton-Sanchez).

https://doi.org/10.1016/j.cor.2023.106489
Received 2 June 2023; Received in revised form 20 November 2023; Accepted 22 November 2023
Available online 25 November 2023

by Alcaraz et al. (2022). This is a bi-objective problem, considering the makespan and the total cost for resource usage as the objectives to minimize. Furthermore, the costs are dependent on both, the resource being utilized and the specific instant at which it is employed, leading to a much more realistic problem. Many practical scenarios can be modeled employing time-dependent resource costs. In the context of production, this holds, for example, with energy costs that may be much cheaper during off-peak times. The costs of some other resources, such as labor, equipment, and raw materials, are subject to variation over time. Likewise, in transportation logistics, the cost of resources such as vehicles, fuel, and drivers can vary significantly depending on the time of day and traffic conditions. For more realistic applications of time-dependent resource costs, we refer to Alcaraz et al. (2022). More specifically, although other authors also consider a bi-objective RCPSP (see, e.g., Wang et al., 2021), to the best of the authors' knowledge, only the aforementioned work has addressed the RCPSP considering time-dependent resource costs from a multi-objective perspective. Frequently, more than one objective emerges as relevant in the context of project management, and this justifies the abundant research and the large number of multicriteria techniques for these problems. In any case, capturing more realistic scenarios, makes it possible to develop tools that can provide a decision-maker with a rich set of alternative solutions from which a better decision can certainly be made.

In contrast to single-objective optimization problems, the concept of optimum is no longer applicable in a multi-objective scenario. Solving multi-objective optimization problems give rise to a set of trade-off optimal solutions, commonly known as the Pareto front (PF), which can be computationally expensive to generate. Moreover, the search space can be too large and too complex to employ exact methods, which has led to the use of heuristic algorithms for approximating the PF. Among these approximate techniques, multi-objective evolutionary algorithms (MOEAs) are widely used (see, e.g., Fonseca and Fleming, 1993; Zitzler, 1999; Emmerich and Deutz, 2018). MOEAs belong to the family of metaheuristic algorithms, which comprises various different methods, such as genetic algorithms (GA), particle swarm optimization (PSO), ant colony optimization, tabu search, differential evolution or scatter search. We refer to Durillo et al. (2010) for an in-depth study of some of the aforementioned techniques.

The unique algorithm developed so far to solve the bi-objective RCPSP_TDRC is the recent metaheuristic proposed by Alcaraz et al. (2022), which is a NSGA-II-based technique (Deb et al., 2002). The aim of this paper is to make several different metaheuristic paradigms available for solving the problem and to conduct a comparative analysis of their performance. To achieve this, we have implemented six additional MOEAs to solve the bi-objective RCPSP_TDRC based on well-known paradigms in the literature: SPEA2 (Zitzler et al., 2001), MOCell (Nebro et al., 2009), PESA-II (Corne et al., 2001), IBEA (Zitzler and Künzli, 2004), SMS-EMOA (Emmerich et al., 2005) and MOEA/D (Zhang and Li, 2007). This selection forms a diverse group of metaheuristics that belong to different categories and make use of very different approaches, as we will describe later. All the new metaheuristics we have implemented make use of the encoding and operators employed by the NSGA-II approach proposed recently. Therefore, the comparison carried out focuses on the performance of the different paradigms to solve the problem rather than the specific versions of the algorithms themselves.

An appropriate comparison of existing techniques to solve a given problem is important to determine which of them offer the best results (see, e.g., Schlünz et al., 2016; Danloup et al., 2018; Govindan et al., 2019). As we are dealing with a bi-objective problem, the results of the algorithms when solving an instance of the problem consists of an approximation of the PF and not a single solution. Zitzler et al. (2003) demonstrate the importance of comparing algorithms quantitatively to obtain the best PF approximation of a given problem and show how challenging this task can be in the context of multi-objective optimization when comparing approximation sets.

The performance of the seven algorithms considered in this work is assessed in a benchmark set of bi-objective RCPSP_TDRC. In detail, the largest instances available in the PSPLIB library (Kolisch and Sprecher, 1997) are employed as a base. Regarding the performance indicators, we select five measures to obtain a detailed description of the characteristics of the PF approximations. For an in-depth understanding of the various metrics available in the field of multi-objective optimization, along with their classification, we refer to Audet et al. (2021). In addition, a statistical analysis of the results obtained is carried out so that they can be compared with a certain level of confidence.

In short, the contribution of this paper is twofold:

1. We implement six different metaheuristics following the main features of the NSGA-II previously developed in Alcaraz et al. (2022), to solve the bi-objective RCPSP_TDRC.
2. We analyze the performance assessment of the considered paradigms when solving a large number of instances of the problem. A previous calibration step allows the algorithms to be configured in order to make a comparison on a equal footing. Furthermore, statistically supported conclusions are derived from the obtained results.

The rest of the paper is organized as follows. In Section 2, the basic notions of multi-objective optimization are discussed and the recently introduced bi-criteria RCPSP_TDRC is formally described. In Section 3, we outline the general framework and classification of MOEAs, along with the shared specific features of all the metaheuristics, before describing the seven algorithms subject to comparison. In Section 4, the five performance measures and the statistical analysis are described. Section 5 is dedicated to presenting and analyzing the experiments conducted and their results. Finally, conclusions and future research directions are given in Section 6.

2. Motivation and problem description

As we have stated previously, this work aims at showing a comprehensive comparison of several metaheuristic algorithms in the context of multi-objective optimization. In this section, we commence with some background on multi-objective optimization. Likewise, we present the problem on which we center our study.

Multi-objective optimization refers to an area of multi-criteria decision-making which deals with optimization problems having more than one objective function to be optimized at a time. It can be applied to many domains of science, where it is necessary to take optimal decisions with trade-offs between two or more conflicting objectives. Given a decision space X, a multi-objective optimization problem is defined as the process of finding a decision vector $\mathbf{x}^* \in X$ which satisfies the set of constraints of the problem, as well as minimizing (or maximizing) the vector of objective functions $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))^T$. Furthermore, the set of values satisfying the constraints of the problem, known as the feasible region, is denoted as Ω. A point $\mathbf{x} \in \Omega$ is said to be a feasible solution.

In the context of multi-objective optimization, there is no longer a unique optimal solution to the problem. We address this issue by finding the set of non-dominated solutions or Pareto optimal solutions, the so-called Pareto optimal set, for which one objective cannot be improved without deteriorating at least one of the other objective functions. The image under the vector function $\mathbf{f}(\mathbf{x})$ of the Pareto optimal set is referred to as the PF. For detailed information on Pareto dominance and other important definitions associated with it, we refer to Emmerich and Deutz (2018).

The RCPSP is a well-known combinatorial optimization problem consisting of a set $V = \{0, 1, \ldots, n, n+1\}$ of activities subject to precedence and resource constraints. Activity 0 represents the start of the schedule, while activity $n+1$ corresponds to the end of it. Both are "dummy" activities with null processing times (or duration) and no

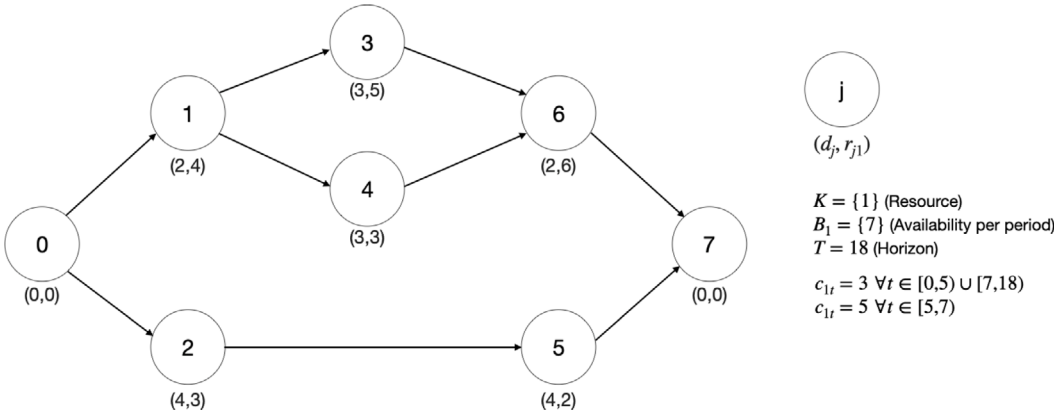


Fig. 1. An example of the RCPSP with time-dependent resource costs.

resource consumption. The processing time of an activity $j \in V$ is denoted as d_j. Preemption is not allowed, i.e., activity j must be executed during d_j time periods from its start, without interruption. Precedence relations are given by sets of immediate predecessors P_j, indicating that an activity j may not be started before each of its predecessors in P_j is completed. On the other hand, renewable resources are formalized by set K. Each activity $j \in V$ demands r_{jk} units of resource $k \in K$ per time unit. Lastly, availability of resource k in each time unit is given by B_k.

Formally, a schedule S is an assignment of a non-negative start time S_j to each activity $j \in V$. Traditionally, the objective of the RCPSP is to find a schedule which results in the earliest possible end of the project, i.e., the completion time of end activity (makespan). In their work, Pritsker et al. (1969) propose a model for this problem based on the maximum time required to complete all the activities, the so-called planning horizon T, which should represent a sharp upper bound of the project optimal makespan.

Taking the RCPSP and the model by Pritsker et al. (1969) as a starting point, Alcaraz et al. (2022) introduce time-dependent resource costs, i.e., the cost depends on the resource being considered as well as the time period in which it is being used. This type of costs are very common in projects when scarce resources are considered. Formally, the authors denote by c_{kt} the cost of employing one unit of resource k during the interval of time $[t, t+1)$, $\forall k \in K$ and $t \in \{0, \ldots, T-1\}$. In this context, they propose adding a second objective to be minimized in the problem, the total cost of the resource usage. In this way, the authors define a new variant of the RCPSP, a multi-objective problem with two objectives to minimize, the so-called resource-constrained project scheduling problem with time-dependent resource costs (RCPSP_TDRC). The mathematical model of this problem is shown in Appendix A.

An example of the above problem is given in Fig. 1. The project comprises 6 activities that utilize a single renewable resource with an availability of 7 units per period. Each activity is paired with its duration and the resource requirement. Dummy activities 0 and 7 are included to represent project start and completion, and the planning horizon is determined by summing the processing times of all activities in the project. Finally, the time-dependent resource costs are shown in the figure with two possible values.

In the presence of two objectives, makespan and total cost of resource usage, we can obtain different solutions to the problem, depending on the objective we are prioritizing. In Fig. 2, we present two feasible solutions and calculate their respective objective values. The schedule shown in Fig. 2(a) achieves a lower makespan value compared to the one in Fig. 2(b), while the cost of resource usage exhibits the opposite trend. Hence, both solutions are non-dominated, in the sense that no objective can be improved without deteriorating the other.

We refer to Alcaraz et al. (2022) for a detailed example of this problem, as well as an illustration of the main differences with respect to the traditional single-objective problem. In that work, the authors demonstrate that exact techniques fail when solving medium or large-sized instances of the problem and propose a metaheuristic as alternative. Specifically, the authors compare the fronts given by the metaheuristic approach with the exact PF given by the AUGMECON method (Mavrotas, 2009). In small and medium-sized instances (30 and 60 activities respectively), where the exact techniques can solve up to proven optimality the optimization models, the metaheuristic demonstrates its efficiency in much lower computation times. In the set of instances where the exact techniques fail, the metaheuristic gives approximations of the PF with very good characteristics. Taking these experiments as a starting point and aiming to compare several metaheuristic paradigms, large-sized projects with 120 activities are mainly considered in this work. However, in order to better support the conclusions, experiments with medium-sized projects with 60 activities are also included in an appendix. The general template of the algorithm proposed by Alcaraz et al. (2022), which is the basis of the algorithms implemented in this work and that will be later compared, is described in the following section. It should be highlighted that we have chosen seminal variants of the metaheuristics selected for the comparison, allowing us to focus on the performance of the different paradigms to solve the problem rather than the specific versions of the algorithms themselves.

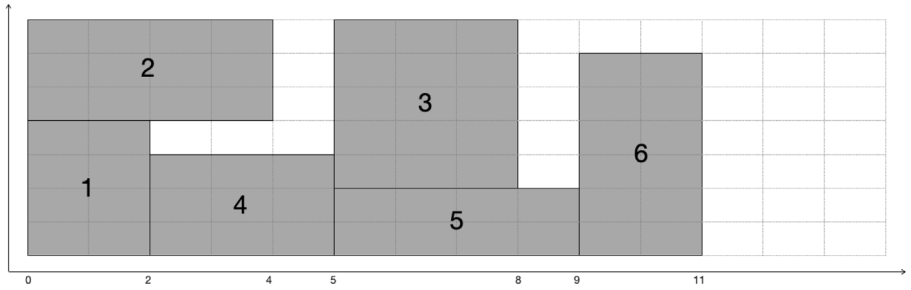
3. Multi-objective evolutionary algorithms

In this section, we first describe the general template of MOEAs, as well as the specific algorithm-features needed to solve the bi-objective RCPSP_TDRC. Next, we briefly discuss the seven metaheuristics considered to carry out the experimental studies.

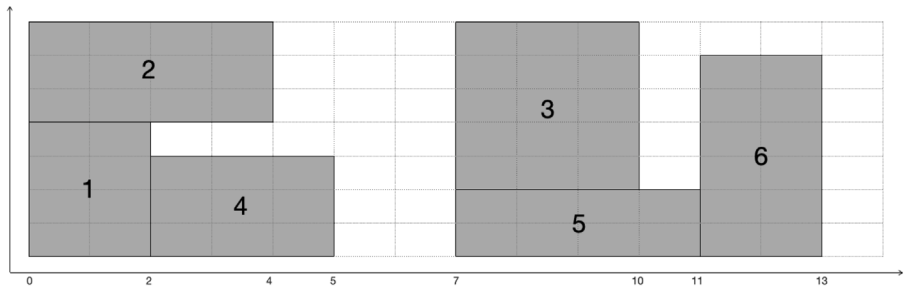
3.1. General framework and taxonomy for MOEAs

In the context of multi-objective optimization, multi-objective evolutionary algorithms, based on paradigms from natural evolution arise. In general, MOEAs employ a variety of procedures, such as selection, crossover and mutation operators that are applied to a set of individuals, the so-called population, in order to guide the process towards the set of Pareto optimal solutions or, at least, a good approximation of it.

Following Emmerich and Deutz (2018), MOEAs can be classified into three main categories based on the paradigms used to define the selection operators:



(a) Schedule 1: Makespan 11, total cost 220.



(b) Schedule 2. Makespan 13, total cost 192

Fig. 2. Different solutions to project in Fig. 1.

1. Pareto-based MOEAs. Individuals are ranked according to two different criteria: the Pareto dominance relation and the contribution of each point to diversity. The first criterion groups points which do not dominate each other and prioritize the non-dominated solutions, that is, those not dominated by any other population member. Next, the second ranking is performed on the groups created using the previous dominance relation.
2. Indicator-based MOEAs. The main idea in this category is the use of indicator functions to distinguish when one approximation set outperforms another. Fitness values obtained by means of these indicators are assigned to the population members, guiding the selection procedure towards the set of Pareto optimal solutions.
3. Decomposition-based MOEAs. The last approach opts to decompose the original problem into several subproblems, through which a specific part of the PF is addressed. To construct each subproblem, a certain weighted scalarization method is used.

Most of the MOEAs we can find in the literature to solve very different optimization problems fall within the first category. Specifically, the most popular MOEAs, those that have been more widely used, are also Pareto-based MOEAs. Therefore, the majority of the seven MOEAs selected for the comparison belong to this category, which includes the NSGA-II used in Alcaraz et al. (2022) to solve the bi-objective RCPSP_TDRC. Additionally, SPEA2, MOCell and PESA-II have also been chosen. To extend the comparison with MOEAs of the three categories, we have also selected two of the best known indicator-based algorithms, IBEA and SMS-EMOA, and one of the most popular decomposition-based algorithms, MOEA/D.

As stated above, MOEAs shared different structures and procedures that do not depend on the specific problem being considered. Algorithm 1 shows the general template of MOEAs, which is common to all the algorithms of this type. The procedure begins by creating an initial population and evaluating all its individuals. The main loop represents the evolution process and will continue until the stopping criterion is satisfied. Common criteria include a maximum number of function evaluations or a CPU time limit. During this phase, the current population undergoes the three genetic operators to generate the offspring population, which is subsequently evaluated. At this stage, individuals from both populations participate in the replacement procedure, where the best solutions are chosen to form the population of the next generation.

Algorithm 1 Evolutionary Algorithm Template

```
1: i ← 0;
2: P_i ← create_initial_population(N);
3: P_i ← evaluate_population(P_i);
4: while not stopping_criterion do
5:     R_i ← selection(P_i);
6:     R_i ← crossover(R_i);
7:     R_i ← mutation(R_i);
8:     R_i ← evaluate_population(R_i);
9:     P_{i+1} ← replacement(P_i, R_i);
10:    i ← i + 1;
11: end while
```

Although the general template described above gives an idea of how MOEAs work, implementing these algorithms for solving a particular problem implies the design of several features, which necessarily incorporate problem-specific knowledge that will guarantee the efficiency of the technique. In particular, we refer to the way to encode the solutions, and the genetic operators, crossover and mutation, to manage this encoding.

3.2. Specific encoding and operators for the RCPSP_TDRC

As stated above, Alcaraz et al. (2022) propose a metaheuristic to solve the bi-criteria RCPSP_TDRC. Specifically, the authors design a solution encoding and genetic operators in order to implement the

algorithm and solve the problem being considered. These features, that are described in detail by Alcaraz et al. (2022), are summarized below:

- **Activity list with scheduling objective.** The solutions are encoded by a double-list, an activity list where each activity must appear after all its predecessors and a binary list representing, for each activity, one of the two objectives considered: the makespan or the total cost for resource usage. This encoding is an extension of the one proposed by Alcaraz and Maroto (2006) for dealing with the RCPSP with only one objective. The order given by the activity list is followed when scheduling the activities in order to build the schedule. Nevertheless, the binary list determines which scheduling objective we prioritize for each activity. Thus, when prioritizing the makespan, activities must be executed the moment there is availability of resources, whereas when considering the second scheduling objective, i.e., the cost, the start time of an activity can be delayed a maximum number of time periods until its processing cost is cheaper, provided that there are sufficient resources to execute it.
  The design of the solution encoding is crucial, since the good performance of the algorithm depends, to a great extent, on it. Fig. 3 shows a solution encoded with the activity list with scheduling objective representation for the project example presented in Fig. 1. We observe that the first list orders activities according to their precedence relationships in the project, i.e., an activity will always appear after all its predecessors. In the second list, activities 1, 2, 4, and 6 prioritize minimizing the makespan when they are scheduled, as their scheduling objective is 0. Meanwhile, activities 3 and 5 have a scheduling objective of 1, indicating a priority on the cost when they are going to be placed in the schedule. More examples of this double-list encoding and its subsequent transformation into a schedule are available in the work by Alcaraz et al. (2022).
- **Double-list crossover.** The crossover operator is applied to a pair of solutions, where information from each double-list is combined to create an offspring. To achieve this goal, a two-step procedure is carried out. In a first stage, the activity lists of the parents undergo the two-point crossover (Hartmann, 1998). Next, the activities conforming the offsprings inherit the scheduling objectives from their parents, taking into account the provenance of each activity.
  Fig. 4 illustrates an example of the double-list crossover procedure. In this example, we have two solutions derived from the project shown in Fig. 1: the mother (M) and the father (F). The crossover operation begins by choosing two random crossover points k_1 and k_2. Let us suppose that $k_1 = 2$ and $k_2 = 4$ are selected. The double lists of the parents are divided into three sections, resulting in the daughter (D) inheriting the first two positions in the activity list from the mother. The next two positions are inherited from the father, following a specific rule: we identify the first two activities in the father that are not already present in the daughter, maintaining their relative order in the father's list. Finally, the last two activities are taken from the mother, ensuring that their relative order on the mother is maintained. The procedure for generating the son's activity list (S) is analogous to that of the daughter but interchanging the role of the parents. After the crossover of the activity lists of the parents, the activities in the offspring inherit the scheduling mode from the respective parent they were copied from.
- **Double-list mutation.** As for the crossover operator, the mutation mechanism is performed in two steps. First, each of the activities in the sequence is moved to a random position (between the last of its predecessors and the first of its successors, generating a feasible solution) with a determined mutation probability (Boctor, 1996; Alcaraz and Maroto, 2001). The scheduling objective list of all the activities is preserved after this procedure. Afterwards, the binary list is altered with certain probability, changing from 0 to 1 or vice versa.

1	2	4	3	5	6
0	0	0	1	1	0

Fig. 3. Example of the activity list with scheduling objective representation.

The features described above make the algorithm unique and its performance highly depends on them. These features will be employed as a common basis in all the metaheuristics compared in this work. Furthermore, in all the metaheuristics, the procedure begins by creating an initial population through the same random mechanism. First, an activity list is generated by randomly selecting activities from an eligible list while maintaining the precedence relationships. After selecting an activity, the eligible list is updated to ensure that no activity is chosen more than once. Second, the scheduling objective for each activity is randomly determined from the two possible values with a 50% probability for each. Then, each solution from the initial population is evaluated, i.e., we obtain the values of makespan and cost for every individual. Therefore, if the different metaheuristics show different performance, these differences cannot be attributed to the encoding or the operators used but to the general scheme of the algorithm. The specific characteristics of all the metaheuristics considered in this work are described in the following subsections.

3.3. NSGA-II

Within the Pareto-based MOEAs, we begin by describing the non-dominated sorting genetic algorithm II (NSGA-II) introduced by Deb et al. (2002), which has been successfully applied to a wide range of problems in very different contexts. Firstly, the random mechanism previously described is applied to obtain the initial population, that is, the set of initial solutions. Next, the first ranking scheme is performed by assigning a fitness (or rank) to each solution, thus sorting the population into different fronts. To elaborate, the first front consists of those solutions that are not dominated by any other individual in the population, while the second front comprises population members dominated by one or more solutions from the first front. This non-domination ranking process continues until all the fronts are identified.

Hereafter, the typical genetic operators, i.e., binary tournament selection, crossover and mutation, are used to obtain an offspring population of the same size as the initial one. Both populations are combined resulting in a double-sized population, on which the non-domination ranking process previously outlined is performed. This procedure yields several sets, so that the best solutions in the combined population are the ones belonging to the first front. In order to build a new population with the original size, individuals of the different fronts are chosen in order, starting with the first front. If the size of the first front is less than the original population size, all the individuals are chosen for the new population. As for the remaining members, they are selected from the subsequent non-dominated fronts, until no more sets can be fully accommodated. When all the individuals of a front cannot be placed in the new population, a procedure is employed to determine the individuals in that front to be copied in the new population. The crowding distance of all the solutions in that front are calculated, representing the distance between a given solution and the rest of solutions in the front, and a higher value is desirable, i.e., solutions located in a lesser crowded area of the objective space are preferred. Then, solutions are sorted in descending order with respect to this distance, and copied one by one, until the new population is fulfilled. Thereby, the first rank of fronts enable us to identify which solutions are preferable based on the non-domination relationship, while the second sorting procedure allows us to choose between the solutions belonging to the same front. Further details on this metaheuristic are described in Deb et al. (2002).

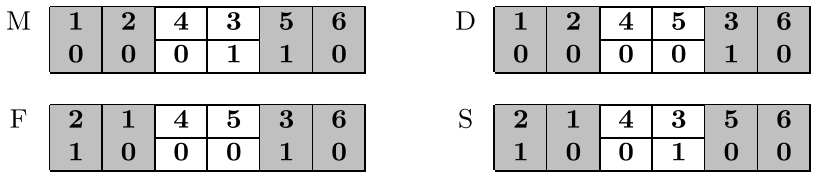


Fig. 4. Example of the double-list crossover.

3.4. SPEA2

We shall now present the improved version of the strength pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1999), namely SPEA2, which is a Pareto-based MOEA for finding or approximating the Pareto optimal set of solutions. Attempting to overcome the shortcomings of its predecessor, SPEA2 incorporates a fitness assignment procedure, which considers for each solution the individuals dominated by the solution as well as the ones that dominate it. Moreover, SPEA2 employs a regular population and an archive. The archive is an external population which stores the non-dominated solutions attained by the algorithm during the optimization process. Precisely, it serves as a representation of the non-dominated front among all the evaluated solutions considered so far. Following the ideas presented in Zitzler et al. (2001), let $\mathbf{x}$ be an individual belonging to either the archive, $\overline{P}$, or the population, P. We denote by $S(\mathbf{x})$ the strength value of $\mathbf{x}$, and it can be calculated as follows:

$$S(\mathbf{x}) = |\{\mathbf{y} : \mathbf{y} \in P + \overline{P} \wedge \mathbf{x} \prec_{Pareto} \mathbf{y}\}|,$$

where $|\cdot|$ denotes the cardinality of a set, $+$ represents the multiset union and $\prec_{Pareto}$ stands for the Pareto dominance relation. Thus, the strength value of an individual represents the number of population members it dominates. Subsequent to the above, the raw fitness $R(\mathbf{x})$ of an individual $\mathbf{x}$ is obtained:

$$R(\mathbf{x}) = \sum_{\mathbf{y} \in P + \overline{P}, \mathbf{y} \prec_{Pareto} \mathbf{x}} S(\mathbf{y}).$$

Hence, the raw fitness of a solution $\mathbf{x}$ is given by the strength values of the individuals which it is dominated by. Although the raw fitness value indicates the extent to which an individual is dominated, it may fail when most of the individuals in a population do not dominate each other. For that reason, a density estimation based on the nearest neighbor technique is combined with the raw fitness in order to discriminate between population members with the same raw fitness. As a result, selection process is biased towards minimizing the fitness values, thus preferring the exploration of less populated regions of the objective space. A detailed description on how to obtain the fitness of an individual $\mathbf{x}$, as the sum of the two values included above, can be found in Zitzler et al. (2001).

Starting with an initial population and an empty archive, we now describe the main loop of this algorithm. First of all, the fitness values of the individuals in both, the original population and the archive, are calculated. We refer to this step as the fitness assignment. Then, the non-dominated individuals are copied into the archive of the next generation. At this point, the environmental selection procedure takes place. If the number of non-dominated solutions is greater than the archive size, a truncation operator based on calculating the distances to the kth nearest neighbor is employed. Otherwise, the archive of the next generation is filled with dominated solutions. Finally, similarly to other MOEAs, the algorithm continues to the mating selection phase, where the mating pool is created by selecting individuals belonging to the previous archive. Afterwards, the variation procedure takes place, and the crossover and mutation operators are applied to the mating pool in order to obtain the resulting population. If a stopping criterion is satisfied, the process stops and the obtained archive which contains the non-dominated individuals will represent the approximation of the Pareto optimal set, that is, the solution of our problem. Again, we refer to Zitzler et al. (2001) for a more detailed explanation of this methodology.

3.5. MOCell

Belonging to the Pareto-based MOEAs class, we now discuss a cellular genetic algorithm (cGA) called MOCell (Nebro et al., 2009), which is based on two chief components: a bounded external archive to store the non-dominated solutions and the small neighborhoods. Here, we concentrate on the concept of small neighborhood, as it constitutes a fundamental element of this technique. When we refer to a neighborhood, we address how individuals interact with each other and, in particular, which kind of cooperation is allowed and which is not. Hence, genetic operators may only be applied to an individual and its immediate neighbors, i.e., within the neighborhood. A neighborhood, in this sense, is a subset of solutions or individuals within the overall population that are considered "close" to each other in terms of their objective values. In this context, we can assume population size is greater or even much greater than the size of a neighborhood. Moreover, this algorithm works with overlapping small neighborhoods within which solutions are shared, inducing a gradual exploration of the search space that guarantees diversification. Besides, exploitation (intensification) occurs inside each neighborhood by means of genetic operators (Alba and Dorronsoro, 2008).

Regarding the breeding loop of MOCell, it starts by considering a bounded and empty external archive and by arranging the population members in a two-dimensional toroidal grid. Next, for each individual, two parents are selected from its neighborhood and undergo the crossover and mutation operators to obtain an offspring. At this stage, the resulting individual is compared to the current one, making use of the crowded-comparison operator introduced in NSGA-II. Hence, the current individual is replaced if it is dominated by the offspring. On the other hand, if both individuals are not comparable according to the Pareto dominance relation, a population made up of nine neighbors, using the $Compact9$ criterion (Whitley, 1993; Alba and Dorronsoro, 2008; Jie et al., 2017) is considered. Therefore, the crowding distance of both individuals in the reduced population is computed, whereby the individual with the worst value is rejected. Lastly, the altered population replaces the old one after undergoing a feedback process where several individuals are selected from the archive with the purpose of replacing the corresponding randomly selected ones in the new population.

With regard to the Pareto set, i.e., the external archive, a ranking based on the crowding distance is performed to decide whether a non-dominated solution is inserted into the archive once it is already full. Specifically, the individual with the worst crowding distance value is discarded, ensuring that the diversity of solutions on the PF is preserved.

3.6. PESA-II

To conclude with the Pareto-based MOEAs we present PESA-II (Corne et al., 2001), a version of the Pareto envelope-based selection algorithm (PESA) (Knowles and Corne, 2000), which outperforms its predecessor providing significantly superior results. Both algorithms share the habitual structure of MOEAs, already presented in previous techniques, in which genetic operators act to gradually approximate the Pareto optimal set. Nevertheless, differences arise when it comes to the selection procedure followed by each of them. In this section, we firstly provide an overview of PESA as it represents the basis of PESA-II, and subsequently describe the latter.

Individual-based selection employed in PESA relies on the adaptive hyper-grid division adopted in Knowles and Corne (1999), i.e., a subdivision of the objective space into hyperboxes within which the number of individuals is computed. Notice that the adaptive grid depends on the number of bi-sections considered, S, which conforms a configurable parameter of this algorithm. For each individual, this method keeps a track of the number of other solutions inside the same hyperbox, the so-called selective fitness or squeeze factor. After this fitness assignment procedure, a general selection scheme is used, through which the parents for genetic operators are chosen among the individuals with small squeeze factors. Furthermore, an external archive stores non-dominated solutions and utilizes the adaptive grid discussed earlier as a density estimator for the selection process. Notice the difference between this algorithm and previously described ones, in which the selection is conducted on the entire population.

On the contrary, PESA-II employs an internal population from which parents are selected to create an offspring, and an external population to store non-dominated solutions. This external population also uses the adaptive hyper-grid division of the objective space adopted by PESA. Nonetheless, a region-based selection scheme is adopted. In region-based selection, the unit of selection is a hyperbox rather than an individual. Therefore, the procedure consists of selecting a hyperbox by any traditional selection method, within which an individual is randomly chosen. This alteration in the selection procedure leads the algorithm to choose isolated individuals instead of non-isolated ones with higher probability than in PESA (Corne et al., 2001.) The resulting individuals are subjected to the crossover and mutation operators. Finally, in the environmental selection process, several non-dominated individuals from the current population are asked to enter in the archive one by one. Like in other MOEAs, if an individual is not dominated by any solution in the archive, then it is included in the external set and all the archive members dominated by the new one are removed from the set. Finally, if the archive is full, the density of the hyperboxes (number of solutions within it) is used so the individual in the most crowded hyperbox is removed.

3.7. IBEA

The indicator-based evolutionary algorithm (IBEA) (Zitzler and Künzli, 2004) is an indicator-based MOEA which utilizes binary quality indicators adaptable to arbitrary decision maker's preferences. A binary quality indicator is defined as a function that maps two approximations of the Pareto optimal set to a real number. Therefore, these indicator functions can be considered as performance measures that allow us to compare the quality of two approximation sets. As an example of a binary indicator function we refer to the binary additive ϵ-indicator, I_{ϵ^+} (see Zitzler et al., 2003 and Section 4.1), which returns the minimum distance that a Pareto optimal set approximation must be translated in the objective space in order to be weakly dominated by another approximation set. In particular, binary quality indicators can be used for the fitness assignment procedure directly as they constitute a natural extension of the Pareto dominance relation. In IBEA, the fitness assignment of an individual $\mathbf{x}$ is performed as follows:

$$F(\mathbf{x}) = \sum_{\mathbf{y} \in P \setminus \{\mathbf{x}\}} -e^{-I(\{\mathbf{y}\},\{\mathbf{x}\})/(c \cdot \kappa)},$$

where I represents the binary quality indicator, P refers to the population, the parameter κ is a scaling factor and c represents the maximum absolute indicator value. Note that in this work we focus on the adaptive IBEA template described in Zitzler and Künzli (2004), where indicator values are scaled to lie in the interval $[-1, 1]$ for all individuals. Further considerations should be taken into account when determining whether an indicator is "appropriate" or not. In this context, the notion of preserving dominance emerges as a requirement for indicators to be compliant with Pareto dominance. We refer to Zitzler and Künzli (2004) for the proof of the compliance with Pareto dominance of the previously introduced fitness scheme.

Next, we shall briefly describe the main loop of the IBEA algorithm. Firstly, the initialization phase generates an initial population, followed by the fitness assignment scheme procedure, where the objective and indicator values are scaled. Hence, a fitness value is assigned to each population member and the individual with the worst value is removed from the population. Afterwards, the fitness values of the remaining individuals are updated. The former environmental selection step is iterated until a fixed population size is reached. At this stage, the algorithm must decide whether the procedure ends or whether it continues with the mating selection and variation steps. If no stopping criterion satisfied, the selection operator is applied to the population in order to fill the mating pool. Hereafter, this mating pool undergoes the crossover and mutation operators in order to obtain an offspring population. Again, we refer to Zitzler and Künzli (2004) for further details of the algorithm.

3.8. SMS-EMOA

Regarding the indicator-based MOEAs, we conclude with the S-metric selection evolutionary multi-objective optimization algorithm (SMS-EMOA) (Emmerich et al., 2005). As previously stated, binary quality indicators allow us to compare the quality of two approximation sets of the PF. Nevertheless, there are performance indicators which compute the quality of a PF approximation, providing an absolute measure of its quality. We refer to the latter as unary quality indicators or just unary indicators. In particular, the hypervolume measure or S-metric introduced in Zitzler (1999) is a function of such type. SMS-EMOA evolves around this unary indicator, which determines the selection procedure biasing the search towards the set of non-dominated solutions or Pareto optimal solutions. Precisely, the hypervolume indicator calculates the dimensions of the dominated space regarding a reference point, which bounds it and must be fixed in advance. This measure is described in detail in Section 4.1. Obviously, a set of non-dominated solutions with a large number of points that are well-distributed along the PF would attain a vaster dominated area.

SMS-EMOA aims at maximizing the hypervolume indicator of a population of size N, in which the employment of an external archive is not necessary. As we contend with a multi-objective scenario, the proposed algorithm adopts a ranking criterion based on the non-domination relation, as in NSGA-II. Therefore, the selection procedure, which is governed by the S-metric, takes place at the worst-ranked layer derived from the ranking scheme. Moreover, it must be emphasized that we are facing a steady-state $(N + 1)$-EMOA, meaning only one offspring results from the genetic operators per iteration. Hence, population size is increased to $N+1$, so one individual must be removed from the set by the end of the loop. In a first scenario, if all the population members are non-dominated, the individual with the smallest hypervolume contribution is removed from the set of solutions. Nonetheless, in the presence of dominated solutions, SMS-EMOA employs the aforementioned ranking scheme and selects the solution with the smallest hypervolume contribution from the worst-ranked layer. We refer to Emmerich et al. (2005) for a detailed explanation of this metaheuristic, including its pseudocode.

3.9. MOEA/D

So far, we have described several MOEAs, including Pareto and indicator-based ones. Henceforth, we comment on an algorithm of the remaining category, that is, a multi-objective evolutionary algorithm based on decomposition (MOEA/D) (Zhang and Li, 2007). This metaheuristic consists in decomposing a multi-objective optimization problem into N scalar optimization subproblems to be optimized simultaneously. Ideally, each of these subproblems attaches a specific region of the objective space and, all at once, aim at approximating the PF of the problem. In particular, this procedure reports less computational issues compared to other methods, such as NSGA-II. Among the different decomposition approaches, the Tchebycheff approach is the one adopted by the current algorithm. Following the ideas introduced in Miettinen (1998), the jth subproblem of this kind can be expressed as follows:

$$\text{minimize } g^{te}(\mathbf{x} : \boldsymbol{\lambda}^j, \mathbf{p}^*) = \max_{1\leq i\leq n}\{\lambda_i^j : f_i(\mathbf{x}) - p_i^*\}$$

$$\text{subject to } \mathbf{x} \in \Omega$$

where $\mathbf{p}^* = (p_1^*, \dots, p_n^*)^T$ is the reference point defined as $p_i^* = \max\{f_i(\mathbf{x}) : \mathbf{x} \in \Omega\}$ for each $i = 1, \dots, n$ and $\boldsymbol{\lambda}^j = (\lambda_1^j, \dots, \lambda_n^j)^T$ is named the jth weight vector. As usual, Ω refers to the feasible region, i.e., the set of values satisfying the constraints of the problem. In this context, given a multi-objective optimization problem and a non-dominated solution $\mathbf{x}^*$, there exists an appropriate $\boldsymbol{\lambda}$ such that $\mathbf{x}^*$ is also an optimal solution of $g^{te}(\mathbf{x} : \boldsymbol{\lambda}, \mathbf{p}^*)$. Likewise, each optimal solution of this Tchebycheff scalarization subproblem provides a point in the optimal PF of the original problem. Hence, varying the weight vector allows for obtaining different non-dominated solutions. In particular, the continuity of g^{te} at $\boldsymbol{\lambda}$ ensures the closeness of the optimal solutions of the jth and the ith scalar optimization subproblems. Therefore, given the jth subproblem, information from the remaining subproblems whose weight vectors are close to $\boldsymbol{\lambda}^j$, is of interest. At this point, the concept of neighborhood of the jth subproblem arises, and it is defined as the set of subproblems whose weight vectors are the closest to $\boldsymbol{\lambda}^j$ in the Euclidian distance. Clearly, the number of weight vectors in the neighborhood is to be fixed beforehand.

As for the main loop of MOEA/D, a population of N solutions $\mathbf{x}^1, \dots, \mathbf{x}^N$ is considered, where each of the individuals corresponds to the best solution found so far by the scalar subproblems obtained by decomposition. Moreover, an external population or archive is maintained to store non-dominated solutions. In the first place, Euclidean distances between pairs of weight vectors are computed, and so neighborhoods are established. Next, for each individual $\mathbf{x}^j$, we consider the neighborhood of the jth subproblem and randomly select two other solutions within it. The chosen parents undergo genetic operators to obtain an offspring $\mathbf{y}$. At this stage, a problem-specific improvement procedure is applied resulting in a new individual $\mathbf{y}'$, which is compared to the current solution and replaces it in case of obtaining an improvement in the value of the objective function of the jth subproblem. As an improvement procedure, we have employed a basic local search based on the double-list mutation described in Section 3.2. Lastly, neighborhoods are conveniently updated, as well as the external archive, regarding the dominance relation. As usual, further details on this algorithm can be found in Zhang and Li (2007).

4. Proposed methodology

In this section, we describe the different metrics used to carry out the comparison, together with the statistical tests considered in the study.

4.1. Performance indicators in multi-objective optimization

When solving a multi-objective optimization problem it is desirable to obtain the set of non-dominated solutions that forms the PF. Comparing the performance of different algorithms when solving a specific multi-objective problem implies determining which of the fronts have the best characteristics. The most important desirable characteristics of a front make reference to distribution, convergence and spread and they can be measured by different metrics (see, e.g., Fonseca and Fleming, 1996; Zitzler, 1999; Deb et al., 2002). Following Audet et al. (2021), the distribution metrics measure how well each region of the objective space is represented. Convergence quantifies how close a set of non-dominated solutions is from the PF in the objective space. Spread focuses on the aspect that points should be far away from each other.

There are several different metrics to measure these characteristics and for a complete review of them we refer to Audet et al. (2021). We have selected five performance indicators that cover the above desirable characteristics. The selection has been carried out taking into account that these indicators have been widely used in the literature and, in addition, they are easy to interpret. Capturing both the properties of convergence and distribution, two measures are used to evaluate the fronts produced by the proposed methodologies:

- **Hypervolume** (HV) (Zitzler, 1999). Named also S-metric, the hypervolume metric was previously introduced along with the SMS-EMOA metaheuristic. In further detail, let $A = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k) \subset X$ be a set of k decision vectors. Considering a bi-objective optimization problem, we denote by $S(A)$ the volume enclosed by the union of the rectangles $p_1, p_2, \dots, p_k$, where each p_i is defined by the points $(0, 0)$ and $(f_1(\mathbf{x}_i), f_2(\mathbf{x}_i))$. This measure was first proposed for a maximization problem, so the obtained front must be inverted to apply the original metric implementation. Moreover, the decision vectors are normalized, restricting the performance indicator to the interval $[0, 1]$. Note that a value closer to 1 indicates a better approximation.
- **Modified Inverted Generational Distance** (IGD^+) (Ishibuchi et al., 2015). Overcoming the drawbacks of its predecessors, GD and IGD measures (see Audet et al., 2021), this second convergence and distribution performance indicator computes the distance between two fronts integrating the dominance relation. In particular, it takes into account the dominance relation when computing the Euclidean distance within elements of both fronts. For this indicator, a lower value is desirable.

As for the distribution and spread indicators, another two measures are considered in the computational study carried out in this work:

- **Spread** (Δ) (Deb et al., 2002). The spread measure takes into account the dispersion of the points within the front. Hence, isolated solutions are preferred. Lower values correspond to better fronts and, in particular, a spread equal to zero represents the most widely and evenly distributed set of non-dominated solutions.
- **μ-indicator** (μ) (Alcaraz, 2022). This metric is defined as the ratio of two well-known performance measures, the Γ indicator (Custódio et al., 2011) and Zitzler's $\mathcal{M}_3^*$ metric (Zitzler et al., 2000). The Γ indicator computes the maximum distance in the infinity norm between two consecutive points in the considered front. Hence, a lower value of this metric is preferred. Likewise, when considering two objective functions, the $\mathcal{M}_3^*$ metric equals to the distance of the two outer solutions. Therefore, a higher value of this measure is desired. By combining both performance indicators, the μ-indicator provides a more dependable measure of how well the points are distributed along the front. In particular, μ is to be minimized.

Finally, the convergence to the PF is measured by the following performance indicator:

- **The additive ϵ-indicator** ($I_{\epsilon+}$) (Zitzler et al., 2003). This metric has been already mentioned when discussing the IBEA metaheuristic. Formally, given two fronts F_1 and F_2 it is defined, for the two-objective case, as follows:

$$I_{\epsilon+}(F_1, F_2) = \min_{\epsilon}\{\forall \mathbf{y} \in F_2\ \exists \mathbf{x} \in F_1 : f_i(\mathbf{x}) - \epsilon \leq f_i(\mathbf{y}) \text{ for } i \in \{1,2\}\}.$$

  The function returns the minimum additive factor by which one front must be translated in the objective space to weakly dominate the other front. Thus, we lean toward lower values of this measure.

4.2. Statistical test

As is stated above, the main goal of this work is to compare the behavior of the considered metaheuristics when solving the bi-criteria RCPSP by measuring the quality of the PF approximations in terms of several performance indicator values. Notice that we are dealing with evolutionary techniques, and therefore, with stochastic algorithms, whose mechanism relies on certain probabilistic operations. Under these circumstances, performance assessment is attained by using an appropriate statistical test, which allows for comparing the obtained results with a certain level of confidence. In this section, we present a two-way standard analysis of variance (ANOVA), which considers two independent categorical variables that can affect the response and interact with each other. In particular, the response variables for the two-way ANOVA are the different performance indicators presented in Section 4.1.

Consider A and B as two distinct factors. For factor A we can find I possible values, while factor B admits J possibilities. We are interested in determining the effect of each of these two factors, and also their interaction. Therefore, we assume that besides the effect that each one can have separately, an interaction can be produced by adding certain values of both factors and combining them to produce a mixed effect. Moreover, for each of the $I \times J$ possibilities, let n_{ij} be the number of samples for level i of factor A and level j of factor B. Overall, a parametrization of the described model can be defined as follows:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}, \tag{1}$$

where $i \in \{1, \ldots, I\}$, $j \in \{1, \ldots, J\}$, $k \in \{1, \ldots, n_{ij}\}$, $\epsilon_{ijk} \in N(0, \sigma^2)$ and are independent. Parameter μ represents the overall mean, parameters α_i represent the main effect of factor A, parameters β_j represent the main effect of factor B, and parameters γ_{ij} represent the interaction between the two factors. Notice that all necessary ANOVA hypotheses should be checked. In particular, errors are assumed to follow a normal (Gaussian) distribution. Nonetheless, the error term can be thought of as the sum of minor influences of unpredictable factors and their levels. Hence, the Central Limit Theorem guarantees that the distribution of the error term approximates normal distribution, due to the large volume of data.

Lastly, we employ Tukey's Honest Significant Difference (HSD) test to determine whether or not the means values of the different performance indicators are significantly different from each other. As for both, the two-way ANOVA and Tukey's HSD test, a confidence level 99% (i.e., significance level of 1% or p-value under 0.01) is used. Note that normality of errors is verified following the same argument as previously mentioned, whereas all the remaining necessary hypotheses are also checked.

So far, a general formulation of the statistical model has been presented. In the following section, we will provide further details on the specific elements involved at each stage of the two-way ANOVA.

5. Computational results

5.1. Experimental setup

The instances to test the performance of the proposed methodologies are based on the available instances in the PSPLIB library (http://www.om-db.wi.tum.de/psplib/ for the RCPS problem). This library comprises the J30, J60, J90 and J120 data sets, where the number indicates the activity count of the projects belonging to each set. In particular, we consider the J120 data set, which includes the largest projects, with 120 activities per project. A total of 600 instances make up this data set, which were generated by a procedure which combines three different factors: Network complexity (NC), resource factor (RF), and resource strength (RS) (Kolisch and Sprecher, 1997). The first factor relates to the average number of direct activity successors, while the second evaluates the proportion of resource requirements per activity. This value falls between 0 and 1, with values close to 1 implying high resource demands. Resource strength assesses the average tightness of resource constraints, also on a scale from 0 to 1. A value approaching 1 indicates ample resources for activities to start at their earliest starting time, whereas values nearing zero indicate resource limitations.

Each factor has different levels and, therefore, their combination provides specific parameter settings from which instances were generated. In the J120 data set, we can find three different levels for NC (1.5, 1.8, and 2.1), and four levels for RF (0.25, 0.5, 0.75, and 1). Besides, there are five levels for RS (0.1, 0.2, 0.3, 0.4, and 0.5). All combinations of the previous instance factors lead to a total of 60 possibilities. Lastly, 10 instance replicates were generated for each possible combination, yielding to a total of 600 projects, as stated above. We refer to these problems as J120Y_Z, where Y ranges from 1 to 60, and Z ranges from 1 to 10. Note that none of the PSPLIB instances considered include time-dependent costs for the resources, so we have generated them following the procedure proposed by Alcaraz et al. (2022). Finally, the planning horizon, T, has been set to the sum of the processing times of all activities.

To evaluate the seven algorithms, we adopt the standard procedure in experimental design. Firstly, a calibration phase is carried out on a calibration set of instances, deciding which is the best configuration for each of the considered metaheuristics. Subsequently, all the algorithms, employing the configuration chosen in the first phase, are compared on an evaluation set of instances, determining whether an algorithm (or some of them) outperforms the others. The benchmark consists of the previously presented J120 data set. In particular, we employ 2 instance replicates from each of the 60 possibilities, considering a total of 120 instances for the calibration set. Hence, 8 instance replicates of each problem remain, yielding a total of 480 instances for the evaluation set. Given that obtaining the optimal PF of the instances considered in this work is not possible, a reference PF for each instance must be computed instead. In each one of the phases, the reference front of an instance is built by collecting all the non-dominated solutions generated by all the runs performed over the instance.

All the executions reported in this work have been performed on the Dantzig Cluster of the Miguel Hernandez University (UMH). This computer equipment uses Rocky Linux release 8.7. In particular, only node g-0 was used. This node is a Supermicro SYS-120GQ-TNRT model with 2 Intel(R) Xeon(R) Silver 4316 CPU at 2.30 GHz (80 cores in total) and 768 Gigabyte of RAM. On the other hand, the considered metaheuristics, defined in Section 3, have been coded in Java (Arnold et al., 2005). Precisely, the open-source jMetal framework (Durillo and Nebro, 2011; Nebro et al., 2015) has been used to implement them.

5.2. Comparison criterion

Before explaining the calibration and the comparison phases in further detail, we must decide on how to compare the results obtained. We are dealing with several performance indicators to assess the quality

of results provided by the different metaheuristics and their respective configurations, depending on whether we are in the comparison or calibration phase, respectively. Hence, a preference order among the metrics must be established in order to rank the best approaches considered so far. At this point, we adopt a discarded criterion. Suppose, for example, that we are in the calibration phase. Thus, we start off by selecting the best configurations with regard to a specific measure, i.e., those configurations belonging to the first group given by Tukey's HSD test. If there is just one element in the group the process finishes. Otherwise a second metric is consulted, checking Tukey's HSD test results and deciding which of the previous best configurations has a better performance. Once again, the procedure continues unless a unique best configuration can be chosen.

The measures chosen for the discarded criterion are ordered as follows: hypervolume (HV), spread (Δ), additive ϵ-indicator (I_{ϵ^+}), modified inverted generational distance (IGD^+) and μ-indicator (μ). We prioritize the first three metrics as they are widely used in the literature (see, e.g., Deb et al., 2002; Durillo et al., 2010; Yepes-Borrero et al., 2021). HV measures both convergence to the PF and diversity of the obtained solutions. Similarly, the additive ϵ-indicator also measures convergence to the optimal front. In contrast to the latter two metrics, and providing a wider range of information, Δ aims at measuring the extents of the spread achieved among the obtained solutions. As will be seen in the comparison phase (Section 5.4), the three previous metrics are insufficient to determine the best metaheuristic, so we also consider the IGD^+ measure, which is also a widely used convergence and distribution performance indicator. On the other hand, the μ-indicator combines two well-known performance indicators, providing a distribution and spread measure that summarizes the information from both. Zitzler et al. (2003) prove that the number of criteria that determine what a good approximation set is, is infinite. Thus, an optimal order cannot be found and we can only achieve a convenient one, summarizing various aspects of the obtained approximation sets.

5.3. Calibration of the algorithms

In this work, we deal with algorithms that depend on different sets of parameters. To compare them, an extensive computational and statistical study is carried out. Nonetheless, a first stage, where those methodologies are calibrated, must be performed. As mentioned above, this first stage is referred to as the calibration phase where each algorithm is calibrated separately, considering a full factorial design of experiments (DOE), with all combinations of factors and levels. In particular, we have chosen a set of parameter values in order to allow a fair comparison between all the metaheuristics.

All the MOEAs considered share these three parameters: population size, N, crossover probability, p_c, and mutation probability, p_m. Note that the archive size is not included in the DOE since it has been set to match the population size. After some preliminary experiments, we have set three different values for N, 50,100 and 200 individuals. On the other hand, we have set two possible levels for the crossover probability, 0.7 and 0.9, and for the mutation probability, $1.0/n$ and $2.0/n$, being n the number of activities in the project. Regarding the last parameter, mutation probabilities that depend on n are commonly used in the literature (see, e.g., Durillo et al., 2010, Deb, 2011, Salto and Alba, 2019). In the PESA-II metaheuristic, we consider two possible number of bi-sections, S, 5 or 10. Likewise, in the IBEA algorithm, we have used three possible values for κ: $0.025, 0.05, 0.1$. Finally, the neighborhood selection probability, p_n, in MOEA/D has two levels: $0.7, 0.9$. An overview of the number of tuning parameter values involved in the calibration phase is included in Table 1, together with the number of possible configurations for each technique.

Since all the metaheuristics employ the same solution encoding and also the same operators, crossover and mutation, to manage this type of encoding (Section 3.2), differences in behavior of the algorithms cannot be due to the use of a better or worse encoding or the use of specific procedures that could be more efficient. Differences between two different metaheuristics should indicate that the basic scheme of one outperforms the other to solve the problem considered.

Table 1
Number of different parameter values and possible configurations.

	Possible values						Number of configurations
	N	p_c	p_m	S	κ	p_n	
NSGA-II	3	2	2	–	–	–	12
SPEA2	3	2	2	–	–	–	12
MOCell	3	2	2	–	–	–	12
PESA-II	3	2	2	2	–	–	24
IBEA	3	2	2	–	3	–	36
SMS-EMOA	3	2	2	–	–	–	12
MOEA/D	3	2	2	–	–	2	24

Once we have presented the different elements regarding the parametrization of the metaheuristics, we should focus on the calibration procedure. At this point, we consider, for example, the SPEA2 algorithm and we elaborate the DOE for this specific technique. We will not go into detail about the remaining metaheuristics, since the steps to follow and the statistical models are similar to those presented for the current algorithm. Therefore, taking a close look to Table 1, we check that the factorial design for SPEA2 gives a total of $3 \times 2 \times 2 = 12$ possible configurations. On the other hand, each configuration is tested in the 120-instance calibration set, performing 3 independent runs by problem. Hence, this first phase results in 4320 runs of SPEA2. Regarding the stopping criterion, it has been set to 2 minutes per run (for all the metaheuristics). Thus, each one of the twelve configurations requires 43,200 s, i.e., to configure SPEA2 we employ a total of 6 days of CPU time to complete the calibration phase.

Next, we present the corresponding two-way ANOVA needed for the statistical study. As one can notice, there is one model for each response variable. Consequently, let us select one of the performance indicators, e.g. I_{ϵ^+}, and introduce the elements implicated in model (1). Hence, we must determine the two independent variables that participate and interact with each other. As we are in the calibration phase, we hold a total of 120 problems to carry out the experiments. Therefore, factor A corresponds to the set of instances, with 120 possible levels. On the other hand, there are a total of 12 different configurations, which are the possible values for factor B. Lastly, for each of the 120×12 possibilities, we select 3 random samples from the response variable. Considering the above-mentioned elements, the current model is as follows:

$$(I_{\epsilon^+})_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}, \tag{2}$$

where $i \in \{1, \ldots, 120\}$, $j \in \{1, \ldots, 12\}$ and $k \in \{1, \ldots, 3\}$. Additional explanation of the present model is omitted as it does not differ from the one offered in model (1). Furthermore, one should note that factor A remains the same for all the algorithms. However, the number of configurations depends on the possible parameter combinations, varying among the metaheuristics and, consequently, modifying the number of levels in factor B of the two-way ANOVA.

We will now look at the computational results derived from this study. First of all, regarding the analysis of variance, we obtain that both factors and their interaction are statistically significant for all the metaheuristics and the different response variables, i.e., performance indicators. As previously stated, significant results show that the differences are unlikely to have occurred by chance with a probability of 99%. In consequence, the aforementioned Tukey's HSD test is used to identify the best configurations found so far, with respect to each metric. In particular, all the configurations are classified by groups (observed means in our case); those belonging to different groups being significantly different.

Following the discarded criterion described in Section 5.2, best configurations of each metaheuristic have been selected. Results are

Table 2
Selected configurations from the calibration phase (n = number of activities).

	Parameter values					
	N	p_c	p_m	S	κ	p_n
NSGA-II	200	0.9	$2.0/n$	–	–	–
SPEA2	100	0.7	$2.0/n$	–	–	–
MOCell	50	0.9	$1.0/n$	–	–	–
PESA-II	200	0.7	$2.0/n$	5	–	–
IBEA	200	0.7	$2.0/n$	–	0.05	–
SMS-EMOA	50	0.9	$2.0/n$	–	–	–
MOEA/D	200	0.7	$1.0/n$	–	–	0.7

Table 3
Solutions evaluated by each metaheuristic in two minutes of computation time.

	NSGA-II	SPEA2	MOCell	PESA-II	IBEA	SMS-EMOA	MOEA/D
Mean	184 205	153 720	170 487	383 550	182 856	166 986	90 139
Median	178 400	148 850	163 328	368 000	182 800	162 306	85 100
IQR	55 450	47 225	55 900	121 650	46 050	47 745	31 850

Table 4
Average number of non-dominated solutions of each metaheuristic.

NSGA-II	SPEA2	MOCell	PESA-II	IBEA	SMS-EMOA	MOEA/D
51.752	65.147	45.441	50.083	8.990	45.201	9.963

shown in Table 2, where each column of the table corresponds to one of the tuning parameters. In particular, parameters not involved in all the metaheuristics, e.g. the number of bi-sections, S, are marked with the "–" symbol.

Finally, regarding the full calibration phase, the 120-instance set has been executed by every single metaheuristic separately, resulting in a total of 7 experiments. As was stated above, SPEA2 requires 6 days of CPU time to complete the calibration phase. Following the same procedure, we can calculate the execution times for all the metaheuristics, resulting in a CPU time of 66 days to complete the full calibration phase.

5.4. Computational comparison among the best configurations

This section presents the numerical results that allow to evaluate the performance of the seven metaheuristics considered in this work. The objective is to compare the obtained best configurations on the 480-instance evaluation set. As in the calibration phase, the same stopping criteria is adopted for every algorithm, with a CPU time limit of 2 minutes per run. Likewise, 3 independent runs are performed per instance, attaining 1440 runs per algorithm. A total of 10,080 fronts are obtained as a result. Thus, each of the seven metaheuristics needs 172,800 s to finish all the executions, for a total of $1,209,600$ seconds, or 14 days of CPU time to complete the comparison phase.

Table 3 shows the average, median and interquartile range of the number of solutions evaluated by the different algorithms within the time limit of 2 min. As one can notice, PESA-II obtains the greater value, resulting in an average of almost 400,000 solutions computed when solving the evaluation set instances. Conversely, the MOEA/D algorithm evaluates near 90,000 in the same time. Nonetheless, performing a larger number of evaluations does not necessarily mean getting better results, as we will observe later. On the other hand, Table 4 contains the average number of non-dominated solutions obtained by each of the seven algorithms. It is remarkable how IBEA and MOEA/D achieve the lowest number of optimal solutions, showing the challenge these metaheuristics face in discovering new non-dominated solutions in the search space.

The boxplots in Fig. 5 show, for each performance indicator, the observed values attached by every considered metaheuristic when solving the 480-instance evaluation set. A higher value is preferred when considering the hypervolume, while a smaller value is desirable for the rest of the metrics. Focusing on one of the graphs, e.g. the boxplot with HV as the response variable, one can observe the poor behavior of IBEA and MOEA/D compared to the five remaining ones. In particular, IBEA shows a median close to 0.30, almost half of the value reached by the rest of the metaheuristics, excluding the MOEA/D algorithm. As for the latter, the median is over 0.5, so it is closer to the 0.75 value attached by the five best algorithms. Nevertheless, we can appreciate some outliers in MOEA/D, the minimum value being similar to the one attained by IBEA. Regarding Δ and the μ-indicator (second and last boxplots), a slightly different behavior can be observed, with MOEA/D reporting the worst results. In detail, IBEA behaves decently when considering the spread measure, outperforming the SPEA2 algorithm. Still, this does not change the fact that IBEA reports unsatisfactory results in terms of convergence and distribution.

To summarize, the graphs report that five of the metaheuristics (NSGA-II, SPEA2, MOCell, PESA-II and SMS-EMOA) attain high-quality results in terms of convergence to the PF, diversity of solutions and extents of the spread achieved within the set of non-dominated solutions. At the same time, we can infer the worst behavior of the IBEA and MOEA/D algorithms when solving the test problems, as the results are consistent. To support statistically the observed performance and decide on the best technique, we proceed to introduce the two-way ANOVA employed in this section, as well as Tukey's HSD test.

Regarding the two-way ANOVA, we consider again five statistical models, one per performance indicator. As in the calibration phase, let us fix the $I_{\epsilon+}$ as the response variable and introduce the elements of model (1). As for the two independent variables, factor A corresponds to the evaluation set, holding a total of 480 possible levels. Likewise, factor B is made up of the 7 best configurations of the metaheuristics derived from the previous phase. Note that both variables remain the same regardless of the performance indicator considered. Lastly, for each of the 480×7 possibilities we select 3 random samples from the output, obtaining the following model:

$$(I_{\epsilon+})_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}, \tag{3}$$

where $i \in \{1, \dots, 480\}$, $j \in \{1, \dots, 7\}$ and $k \in \{1, \dots, 3\}$. Again, further hypotheses from model (1) are omitted. As for the results, the main effects and their interaction are statistically significant for all the performance indicators considered in the study. Thus, the value of the metric is affected, not only by the algorithm considered, but also by the instance being solved and the interaction between these two factors.

Given the above results, we guarantee the effect of the algorithm is significant over the performance indicator value. We now aim at deciding on the best metaheuristic, which is the final purpose of this work. As in the previous section, Tukey's HSD test is employed to perform a single-step multiple comparison procedure to determine which is the best algorithm among the best configurations. Table 5 shows the average values of the five performance indicators presented in Section 4.1, following the order given by the discarded criterion. At the same time, groups given by Tukey's HSD test are displayed in a second column for each of the metrics. Precisely, we now have that algorithms belonging to different HSD groups report significantly different mean values. In the table, algorithms that achieved the best results are marked in bold, whereas italic is used to show that a metaheuristic belonged to the best treatment group in the previous model. Additionally, we include one last row in Table 5, showing the best results found so far following the lexicographic order.

Next, we further comment on the aforementioned discarded criterion, which is necessary to determine the best metaheuristic. Hence, we begin labeling NSGA-II, PESA-II and SMS-EMOA, as they fall into the first group regarding the hypervolume. Next, we consult the second metric, i.e., spread, and we rank the previous metaheuristics based on

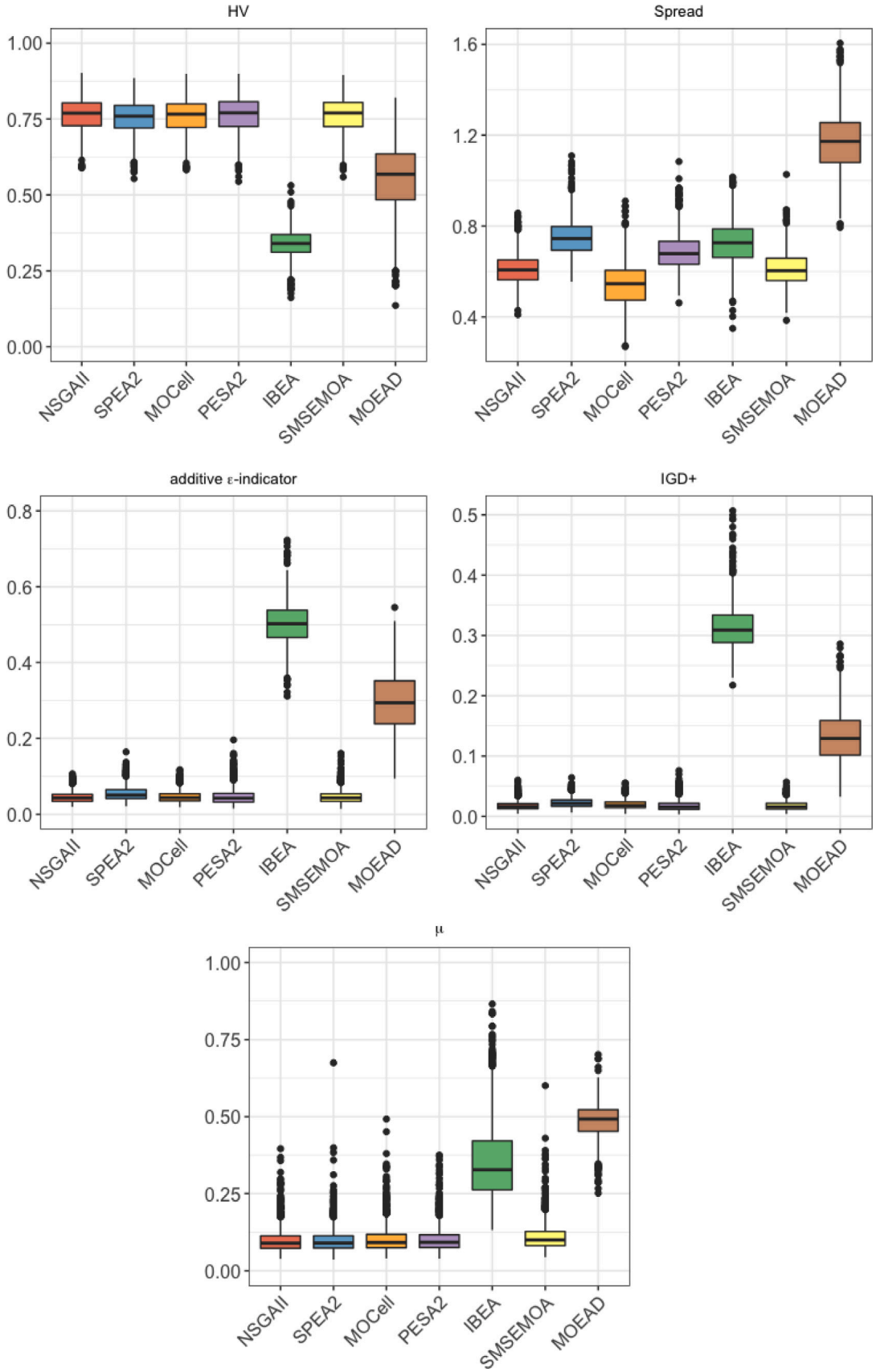


Fig. 5. Boxplots of the performance indicators for the best configuration of each metaheuristic.

Table 5
Computational results for the best configurations of the algorithms.

	HV		Δ		I_{ϵ^+}		IGD^+		μ	
	Mean	Group	Mean	Group	Mean	Group	Mean	Group	Mean	Group
NSGA-II	**0.765**	a	***0.610***	b	***0.045***	a	***0.018***	a	***0.099***	a
SPEA2	0.756	c	0.749	e	0.055	b	0.023	c	0.099	a
MOCell	0.761	b	0.540	a	0.046	a	0.020	b	0.102	a
PESA-II	**0.765**	a	*0.686*	c	0.047	a	0.018	a	0.103	a
IBEA	0.339	e	0.725	d	0.502	d	0.313	e	0.353	c
SMS-EMOA	**0.765**	a	***0.611***	b	***0.046***	a	***0.018***	a	*0.112*	b
MOEA/D	0.558	d	1.170	f	0.293	c	0.132	d	0.486	d
BEST	NSGA-II PESA-II SMS-EMOA		NSGA-II SMS-EMOA		NSGA-II SMS-EMOA		NSGA-II SMS-EMOA		NSGA-II	

this new measure. As a result, PESA-II is discarded, as it falls into the third treatment group. The procedure is repeated for the ϵ-indicator. Nonetheless, the best technique cannot be decided yet. For this reason, two performance indicators are added to the sequence: the IGD^+ and the μ-indicator. Finally, we observe that the last metric, the μ-indicator, is necessary in order to draw a final conclusion for this experimental study.

On average, we have noticed that the NSGA-II algorithm reports a slightly better performance in all the five measures. It belongs to the first treatment group in each of them, except for spread, where it falls into the second group. Similarly, the SMS-EMOA and MOCell algorithms offer competitive results, ranking the second group in only two metrics. As discussed above, the noticeable poor performance of IBEA and MOEA/D is confirmed by the numerical results.

In order to get additional insights from the above results, we represent the means plot of the five performance indicators with Tukey's HSD 99% confidence intervals for the selected best configurations of the metaheuristics. In Fig. 6, algorithms are ranked depending on the group they belong to. Therefore, group a is shown in the first place, and corresponds to those methods with better average values of the performance indicators. As discussed above, when HSD groups differ, there are significant differences between the observed means. Hence, the following representation allows us to have an accurate picture of the best algorithms depending on the metric considered, together with the range of these indicator values.

From these graphs and Table 5, we can draw some final conclusions of our study. It is worth noting that the performance of NSGA-II and SMS-EMOA compared to the remaining metaheuristics is noticeably superior. These algorithms provide high-quality average values for each of the measures considered. Moreover, treatment groups are reported together with these mean values, showing how NSGA-II falls into the first group in four out of the five statistical models, while SMS-EMOA attains it in three of them. Precisely, we can conclude that the NSGA-II metaheuristic reports a significantly better performance, following our discarded criterion. At the same time, graphs in Fig. 6 evidence how MOEA/D and IBEA occupy bottom positions with respect to the treatment groups.

To better support the above conclusions, further experimental research has been conducted using instances of a different size. Particularly, we have carried out the calibration and comparison phases in the medium-sized projects with 60 activities of the PSPLIB library (J60 data set). The results are detailed in Appendix B, where we can observe a behavior similar to the experiment with the J120 data set.

To conclude with the experimental results, Fig. 7 shows two representations of the PF approximation attained by the seven metaheuristics when solving two instances belonging to the evaluation set. In order to provide extensive information, we select the eighth replicate of instance 1 (J1201_8) and the fifth replicate of instance 30 (J12030_5), which report slightly different behaviors with regard to the algorithms. In particular, we notice that MOEA/D obtains better results when solving the first instance in terms of convergence, while it differs from the remaining front approximations in the second case. Likewise, the resulting number of non-dominated solutions in the final population is much less compared to that obtained by the other techniques, except for IBEA. The latter reports poor results in terms of convergence and distribution. Moreover, the given fronts have a considerably lower number of points. Regarding the spread measures, the worst performance is observed with MOEA/D. Overall, for all aspects of convergence, spread and distribution of solutions, NSGA-II, PESA-II, SPEA2, MOCell and SMS-EMOA report a better performance than the previous algorithms.

6. Final remarks

In this work, we have implemented a set of state-of-the-art metaheuristics to solve the bi-criteria RCPSP_TDRC in order to compare their performance. The benchmark has been built from instances available in the PSPLIB library, using the largest projects, those made up of 120 activities. The computational study has been divided into two differentiated stages. The first one corresponds to the calibration phase, where algorithms are adequately configured on a calibration set of instances. The second stage comprises the comparison among the best configuration of each metaheuristic on an evaluation set of instances.

The seven metaheuristics compared employ the same encoding and operators and therefore, the differences in their performance cannot be attributed to those features but to the basic template of the methodology. Such considerations are crucial to attain an appropriate comparison of the performance of the methodologies considered in this survey when solving the present variant of the RCPSP. Results show that majority of the studied methods provide sharp approximations of the reference PF. Moreover, the high-quality approximation sets are found within a CPU time limit of 2 minutes per execution, which is a considerably small amount of time.

Results are consistent along the five performance indicators included in the study. Our work has revealed that NSGA-II is one of the most promising approaches to deal with the bi-criteria RCPSP_TDRC. Specifically, regarding the comparison criterion, it has reported significantly superior results. Conversely, MOEA/D and IBEA have performed poorly compared to the remaining algorithms. Results with other instance sizes support the previous conclusions. In any case, considering the overall performance of the methodologies assessed in this paper, decision-makers have access to a rich range of first-rate algorithms that can report different solutions, thereby enhancing their ability to determine the best choice.

Different research lines are open with this work. The consideration of uncertainty in this problem makes it harder but adds a very common feature in real-world projects. Moreover, the multi-mode variant of the RCPSP could also be studied when incorporating time-dependent resource costs from a multi-objective perspective, leading to a much more realistic problem. Lastly, given that NSGA-II seems to be one of the most promising paradigms to solve this problem, some other variants of this metaheuristic, such as r-NSGA-II (Said et al., 2010), U-NSGA-III (Seada and Deb, 2015) or CHIM-NSGA (Filatovas et al., 2020) could be implemented and compared.

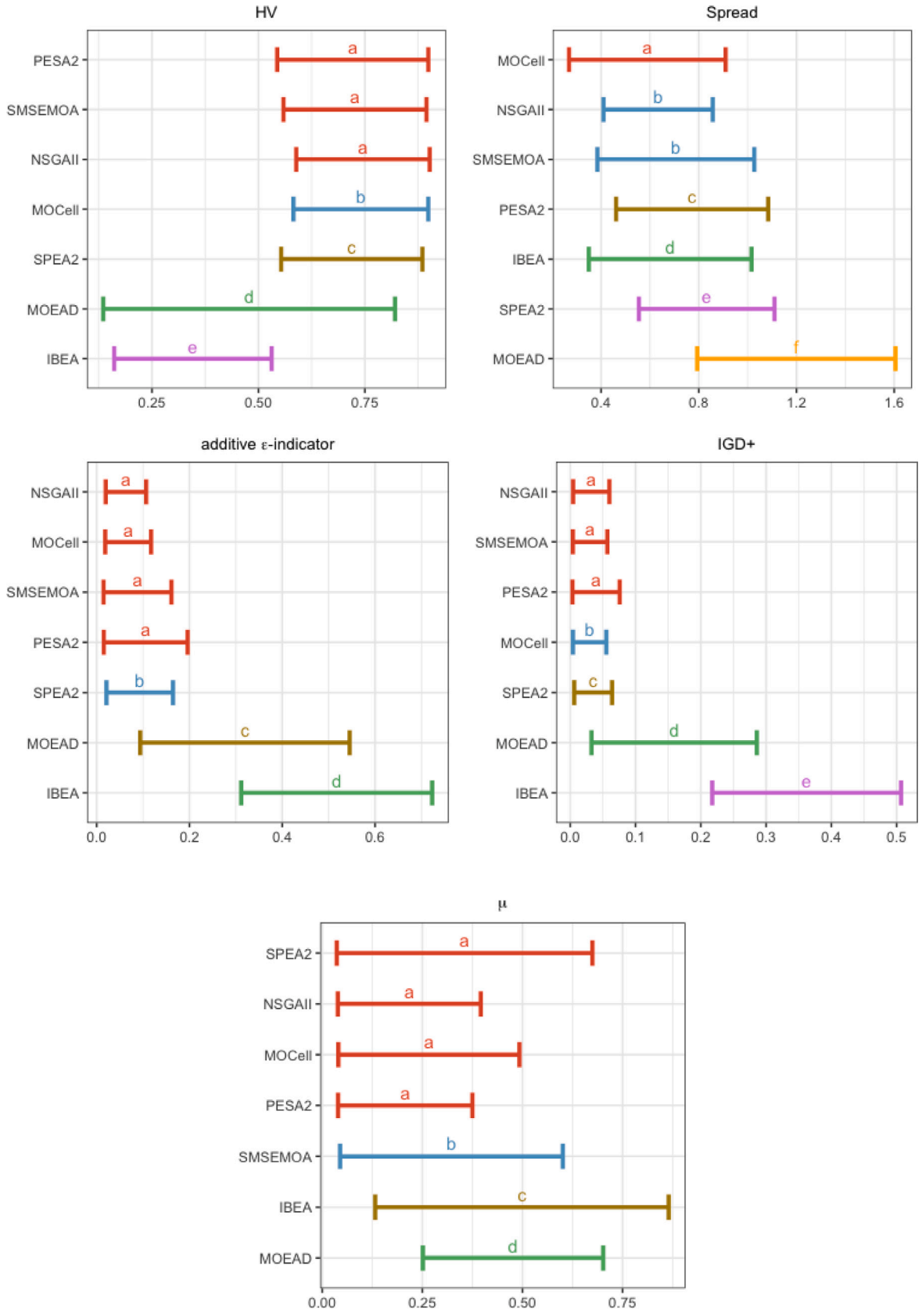


Fig. 6. Performance indicator mean plots with Tukey's HSD 99% confidence intervals.

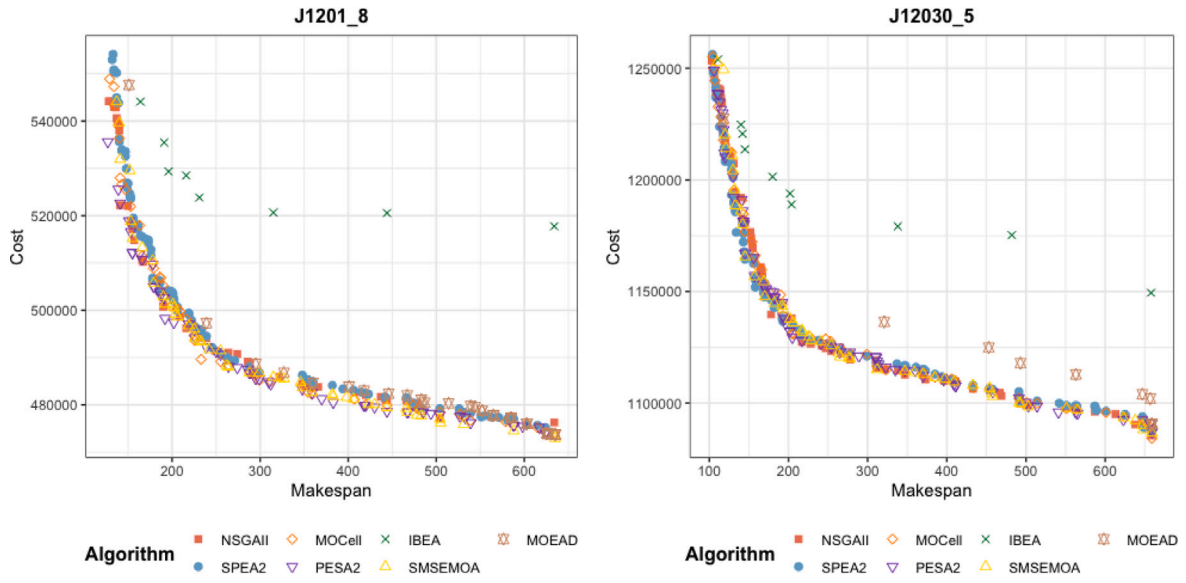


Fig. 7. Approximations of the PF.

CRediT authorship contribution statement

Sofía Rodríguez-Ballesteros: Formal analysis, Investigation, Methodology, Software, Validation, Writing– original draft, Writing – review & editing. **Javier Alcaraz:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Supervision, Writing – original draft, Writing – review & editing. **Laura Anton-Sanchez:** Conceptualization, Formal analysis, Investigation, Methodology, Software, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data sets used in this study, formed by the 600 instances of the J120 data set and the 480 instances of the J60 data set from PSPLIB where resource costs have been added, are available from the corresponding author, J.A., upon reasonable request.

Acknowledgments

This work was supported by the Ministerio de Ciencia e Innovación/ Agencia Estatal de Investigación/10.13039/501100011033, Spain under grants PID2019-105952GB-I00 and PID2021-122344NB-I00 by "ERDF A way of making Europe". It was also supported by the government of the Valencian Community, Spain under grant PROMETEO/2021/063.

Appendix A. Mathematical model for the bi-objective RCPSP_TDRC

We need to define some elements in order to present the mathematical formulation of the bi-objective RCPSP_TDRC. Firstly, given an activity $j \in V$, we define its earliest and latest starting time, denoted by E_j and L_j, respectively. We state the former as the minimum time required to execute all the direct and transitive predecessors of j, while the latter is the maximum time j can be held up without causing a delay in the project, i.e., the duration of the project surpasses the planning horizon, T. On the other hand, the binary response variables are defined as

$$y_{jt} = \begin{cases} 1 & \text{if } j \text{ starts at time } t, \\ 0 & \text{otherwise,} \end{cases}$$

$\forall j \in V$ and $t \in \{E_j, \dots, L_j\}$. Hence, the problem can be formulated as follows:

minimize

$$\mathbf{f}(\mathbf{y}) = \left(\sum_{t=E_{n+1}}^{T} t\, y_{(n+1),t}, \sum_{j \in V \setminus \{n+1\}} \sum_{t=\max\{0,E_j\}}^{\min\{T-1,L_j\}} \left(y_{jt} \sum_{\tau=t}^{t+d_j-1} \sum_{k \in K} r_{jk} c_{k\tau} \right) \right) \tag{4}$$

$$\text{s.t.:} \quad \sum_{t=E_j}^{L_j} y_{jt} = 1 \qquad \forall j \in V, \tag{5}$$

$$\sum_{t=E_j}^{L_j} t y_{jt} - \sum_{t=E_i}^{L_i} t y_{it} \geq d_i \qquad \forall i, j \in V : i \in P_j, \tag{6}$$

$$\sum_{j \in V} r_{jk} \cdot \sum_{\tau=\max\{t-d_j+1,E_j\}}^{\min\{t,L_j\}} y_{j\tau} \leq B_k \quad \forall k \in K, t \in \{0, \dots, T-1\}, \tag{7}$$

$$y_{jt} \in \{0,1\} \qquad \forall j \in V, t \in \{E_j, \dots, L_j\}. \tag{8}$$

The two objectives of the problem, the makespan and the total cost for resource usage, respectively, are defined in (4). Constraints (5) guarantee the execution of every activity of the project, beginning at a particular time period between its earliest and latest starting times. Constraints (6) refer to the precedence relationship, ensuring no activity starts before its predecessors have been completed. The resource constraints are included in (7), making sure the availability of each resource is not exceeded during any time period. Finally, (8) restrict the decision variables to the subset $\{0,1\}$.

Appendix B. Computational results for J60 data set

Experimental results for the J60 data set are shown next. This set is conformed by a total of 480 instances, with 60 activities per project. The instances in this set were also generated by a procedure which combines the same factors considered for the J120 data set:

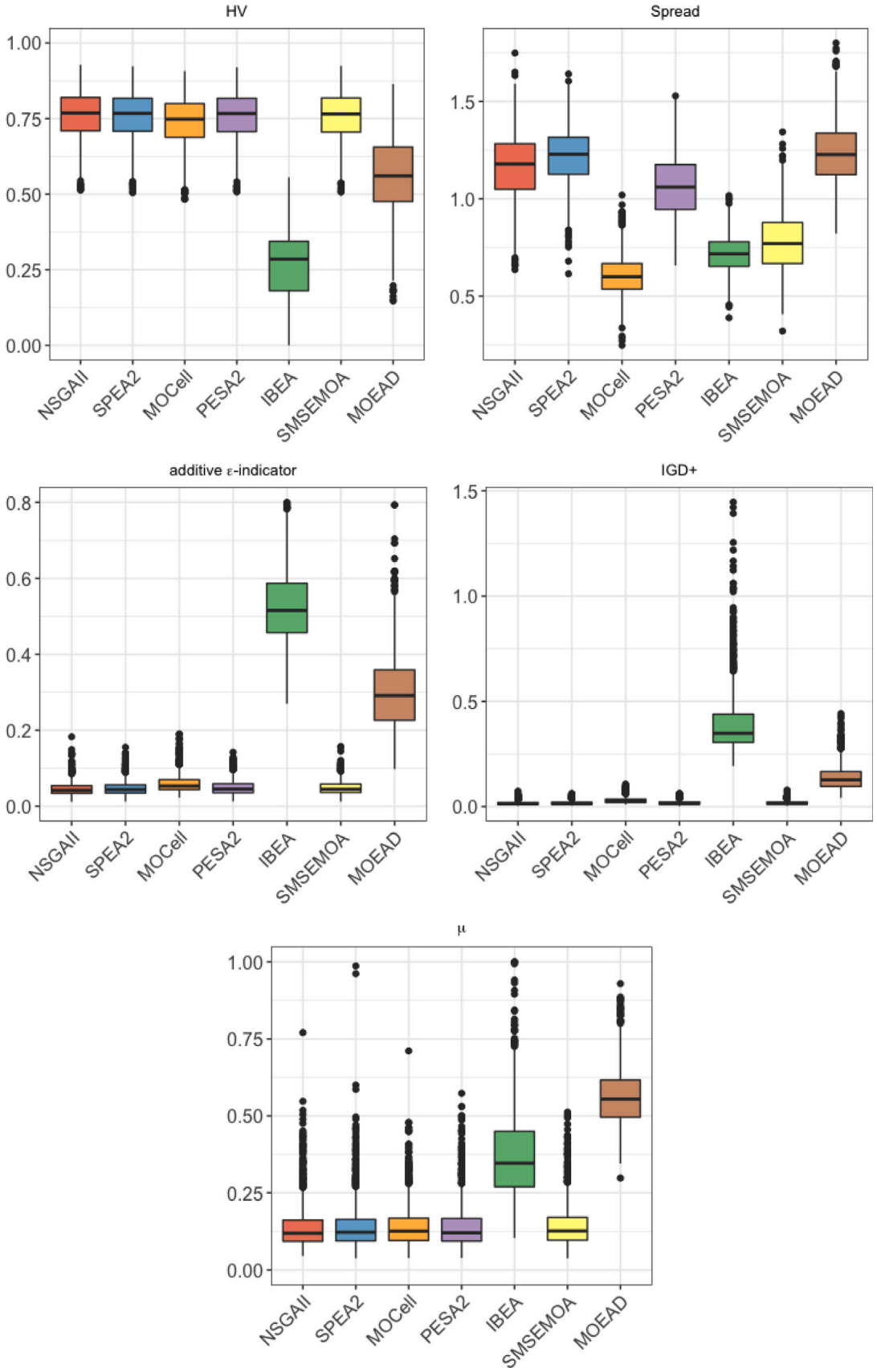


Fig. B.1. Boxplots of the performance indicators for the best configuration of each metaheuristic.

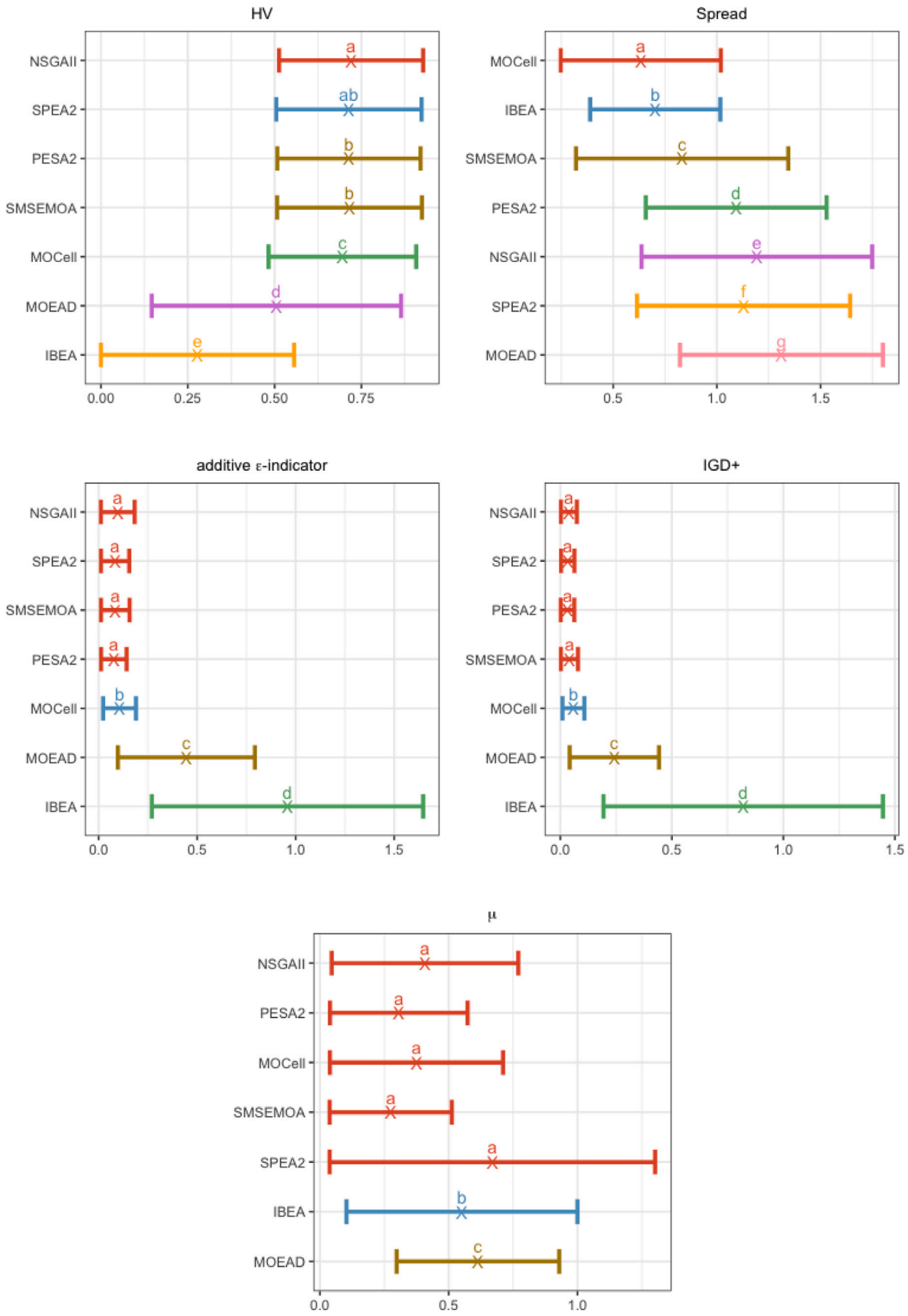


Fig. B.2. Performance indicator mean plots with Tukey's HSD 99% confidence intervals.

Table B.1
Selected configurations from the calibration phase (n = number of activities).

	Parameter values					
	N	p_c	p_m	S	κ	p_n
NSGA-II	100	0.9	$1.0/n$	–	–	–
SPEA2	100	0.7	$1.0/n$	–	–	–
MOCell	50	0.9	$1.0/n$	–	–	–
PESA-II	200	0.9	$2.0/n$	5	–	–
IBEA	200	0.7	$2.0/n$	–	0.05	–
SMS-EMOA	50	0.9	$2.0/n$	–	–	–
MOEA/D	200	0.7	$2.0/n$	–	–	0.7

Table B.2
Average number of non-dominated solutions of each metaheuristic.

NSGA-II	SPEA2	MOCell	PESA-II	IBEA	SMS-EMOA	MOEA/D
38.308	39.347	34.480	37.553	7.945	35.555	9.464

Network complexity (NC), resource factor (RF), and resource strength (RS) (Kolisch and Sprecher, 1997). In particular, we can find three different levels for NC (1.5, 1.8, and 2.1), four levels for RF (0.25, 0.5, 0.75, and 1) and four levels for RS (0.2, 0.5, 0.7, and 1), which gives a total of 48 different combinations. From the 480 instances available, a total of 96 instances are considered for the calibration set, while 384 instances remain for the evaluation set, chosen in the same way as for the J120 data set.

Table B.1 includes the best configurations selected during the calibration phase. The values of the parameters considered in this phase are the same as those described in Section 5.3. Once again, the discarded criterion (Section 5.2) is used to decide the best set of parameters for each metaheuristic. Regarding the results, it is worth highlighting the similarities to the best parameter values obtained for the J120 data set. Following the same procedure as before, we can calculate the execution times for all the algorithms resulting in a CPU time of almost 53 days to complete the full calibration phase.

As for the comparison phase, we first include in Table B.2 the average number of non-dominated solutions obtained by each of the seven algorithms. Like before, IBEA and MOEA/D found the fewest optimal solutions.

Observed values of the performance indicators attained by each of the seven metaheuristics are included in Fig. B.1. By examining the boxplots, we can observe a behavior similar to the experiment with the J120 data set. Overall, we may discern a markedly worse performance of IBEA and MOEA/D compared to the other five algorithms. However, slight differences can be seen when it comes to the spread measure, where IBEA does not appear to perform as poorly.

Finally, we have included in Fig. B.2 the graphs with the mean values of the performance indicators, as well as the Tukey's HSD confidence intervals. Algorithms are ranked following the treatment groups provided by the statistical test. Following the discarded criterion, we begin by examining the results for the hypervolume. We get that NSGA-II and SPEA2 are the algorithms that report the best results, i.e., there is no significant difference between their observed means. Hence, we consult the following metric: spread. The results indicate that SPEA2 can be eliminated, so we conclude that NSGA-II exhibits significantly better performance.

As we can see, the remaining three metrics are not necessary in order to draw a conclusion for the comparison phase. Nonetheless, they provide valuable information regarding the behavior of the different metaheuristics. For example, it may be observed that, in general, MOEA/D and IBEA tend to fall into the worst treatment groups, while NSGA-II consistently occupies the top positions in most of the statistical models. To conclude, the total CPU time necessary to complete the comparison phase is approximately 11 days.

References

Alba, E., Dorronsoro, B., 2008. On the effects of structuring the population. In: Cellular Genetic Algorithms. Springer US, Boston, MA, ISBN: 978-0-387-77610-1, pp. 37–46. http://dx.doi.org/10.1007/978-0-387-77610-1_3.

Alcaraz, J., 2022. Redesigning a multiobjective metaheuristic for the support vector machine with feature selection. Preprint on webpage at https://www.scopus.com/inward/record.uri?eid=2-s2.0-85142011330&partnerID=40&md5=410a800d3af7187a5f09ab3f226d0dc5.

Alcaraz, J., Anton-Sanchez, L., Saldanha-da-Gama, F., 2022. Bi-objective resource-constrained project scheduling problem with time-dependent resource costs. J. Manuf. Syst. 63, 506–523. http://dx.doi.org/10.1016/j.jmsy.2022.05.002.

Alcaraz, J., Maroto, C., 2001. A robust genetic algorithm for resource allocation in project scheduling. Ann.. OR 102, 83–109. http://dx.doi.org/10.1023/A:1010949931021.

Alcaraz, J., Maroto, C., 2006. A hybrid genetic algorithm based on intelligent encoding for project scheduling. In: Józefowska, J., Weglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Vol. 92. Springer, Boston, MA, pp. 249–274. http://dx.doi.org/10.1007/978-0-387-33768-5_10.

Arnold, K., Gosling, J., Holmes, D., 2005. The Java Programming Language, fourth ed. Addison Wesley Professional, ISBN: 978-0321349804.

Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., Salomon, L., 2021. Performance indicators in multiobjective optimization. European J. Oper. Res. (ISSN: 0377-2217) 292 (2), 397–422. http://dx.doi.org/10.1016/j.ejor.2020.11.016.

Ballestín, F., Blanco, R., 2015. Theoretical and practical fundamentals. In: Schwindt, C., Zimmermann, J. (Eds.), Handbook on Project Management and Scheduling. Vol. 1. Springer, Cham, ISBN: 978-3-319-05443-8, pp. 411–427. http://dx.doi.org/10.1007/978-3-319-05443-8_19.

Blazewicz, J., Lenstra, J., Kan, A., 1983. Scheduling subject to resource constraints: Classification and complexity. Discrete Appl. Math. (ISSN: 0166-218X) 5 (1), 11–24. http://dx.doi.org/10.1016/0166-218X(83)90012-4.

Boctor, F.F., 1996. Resource-constrained project scheduling by simulated annealing. Int. J. Prod. Res. 34 (8), 2335–2351. http://dx.doi.org/10.1080/00207549608905028.

Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J., 2001. PESA-II: Region-based selection in evolutionary multiobjective optimization. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation. GECCO '01, Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, ISBN: 1558607749, pp. 283–290.

Coughlan, E.T., Lübbecke, M.E., Schulz, J., 2015. A branch-price-and-cut algorithm for multi-mode resource leveling. Eur. J. Opera. Res. (ISSN: 0377-2217) 245 (1), 70–80. http://dx.doi.org/10.1016/j.ejor.2015.02.043.

Custódio, A.L., Madeira, J.F.A., Vaz, A.I.F., Vicente, L.N., 2011. Direct multisearch for multiobjective optimization. SIAM J. Optim. 21 (3), 1109–1140. http://dx.doi.org/10.1137/10079731X.

Danloup, N., Allaoui, H., Goncalves, G., 2018. A comparison of two meta-heuristics for the pickup and delivery problem with transshipment. Comput. Oper. Res. 100, 155–171. http://dx.doi.org/10.1016/j.cor.2018.07.013.

Deb, K., 2011. Multi-objective optimisation using evolutionary algorithms: An introduction. In: Wang, L., Ng, A.H.C., Deb, K. (Eds.), Multi-Objective Evolutionary Optimisation for Product Design and Manufacturing. Springer, London, pp. 3–34. http://dx.doi.org/10.1007/978-0-85729-652-8_1.

Deb, K., Agrawal, S., Pratap, A., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (2), 182–197. http://dx.doi.org/10.1109/4235.996017.

Durillo, J.J., Nebro, A.J., 2011. Jmetal: A Java framework for multi-objective optimization. Adv. Eng. Softw. (ISSN: 0965-9978) 42 (10), 760–771. http://dx.doi.org/10.1016/j.advengsoft.2011.05.014.

Durillo, J.J., Nebro, A.J., Coello, C.A.C., Garcia-Nieto, J., Luna, F., Alba, E., 2010. A study of multiobjective metaheuristics when solving parameter scalable problems. IEEE Trans. Evol. Comput. 14 (4), 618–635. http://dx.doi.org/10.1109/TEVC.2009.2034647.

Emmerich, M., Beume, N., Naujoks, B., 2005. An EMO algorithm using the hypervolume measure as selection criterion. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (Eds.), Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg, ISBN: 978-3-540-31880-4, pp. 62–76.

Emmerich, M., Deutz, A., 2018. A tutorial on multiobjective optimization: Fundamentals and evolutionary methods. Nat. Comput. 17, 585–609. http://dx.doi.org/10.1007/s11047-018-9685-y.

Filatovas, E., Kurasova, O., Redondo, J., Fernández, J., 2020. A reference point-based evolutionary algorithm for approximating regions of interest in multiobjective problems. TOP 28 (2), 402–423. http://dx.doi.org/10.1007/s11750-019-00535-z.

Florez-Perez, L., Castro-Lacouture, D., Medaglia, A., 2013. Sustainable workforce scheduling in construction program management. J. Oper. Res. Soc. 64 (8), 1169–1181. http://dx.doi.org/10.1057/jors.2012.164.

Fonseca, C.M., Fleming, P.J., 1993. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: Forrest, S. (Ed.), Proccedings of the CGA-93: Fith International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, CA, pp. 416–423.

Fonseca, C.M., Fleming, P.J., 1996. On the performance assessment and comparison of stochastic multiobjective optimizers. In: Voigt, H.-M., Ebeling, W., Rechenberg, I., Schwefel, H.-P. (Eds.), Parallel Problem Solving from Nature - PPSN IV. Springer, Berlin, Heidelberg, ISBN: 978-3-540-70668-7, pp. 584–593.

Govindan, K., Jafarian, A., Nourbakhsh, V., 2019. Designing a sustainable supply chain network integrated with vehicle routing: A comparison of hybrid swarm intelligence metaheuristics. Comput. Oper. Res. 110, 220–235. http://dx.doi.org/10.1016/j.cor.2018.11.013.

Habibi, F., Barzinpour, F., Sadjadi, S., 2018. Resource-constrained project scheduling problem: Review of past and recent developments. J. Proj. Manag. 3, 55–88. http://dx.doi.org/10.5267/j.jpm.2018.1.005.

Hartmann, S., 1998. A competitive genetic algorithm for resource-constrained project scheduling. Nav. Res. Logist. 45 (7), 733–750. http://dx.doi.org/10.1002/(SICI)1520-6750(199810)45:7<733::AID-NAV5>3.0.CO;2-C.

Hartmann, S., Briskorn, D., 2010. A survey of variants and extensions of the resource-constrained project scheduling problem. Eur. J. Oper. Res. (ISSN: 0377-2217) 207 (1), 1–14. http://dx.doi.org/10.1016/j.ejor.2009.11.005.

Hartmann, S., Briskorn, D., 2022. An updated survey of variants and extensions of the resource-constrained project scheduling problem. European J. Oper. Res. (ISSN: 0377-2217) 297 (1), 1–14. http://dx.doi.org/10.1016/j.ejor.2021.05.004.

Herroelen, W., Leus, R., 2005. Project scheduling under uncertainty: Survey and research potentials. European J. Oper. Res. (ISSN: 0377-2217) 165 (2), 289–306. http://dx.doi.org/10.1016/j.ejor.2004.04.002.

Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y., 2015. Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (Eds.), Evolutionary Multi-Criterion Optimization. Springer International Publishing, Cham, pp. 110–125. http://dx.doi.org/10.1007/978-3-319-15892-1_8.

Jie, L., Liu, W., Sun, Z., Teng, S., 2017. Hybrid fuzzy clustering methods based on improved self-adaptive cellular genetic algorithm and optimal-selection-based fuzzy c-means. Neurocomputing (ISSN: 0925-2312) 249, 140–156. http://dx.doi.org/10.1016/j.neucom.2017.03.068.

Knowles, J., Corne, D., 1999. The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation. Vol. 1. CEC99, IEEE Press, Washington, DC, USA, pp. 98–105. http://dx.doi.org/10.1109/CEC.1999.781913.

Knowles, J.D., Corne, D.W., 2000. Approximating the nondominated front using the Pareto archived evolution strategy. Evolut. Comput. 8 (2), 149–172. http://dx.doi.org/10.1162/106365600568167.

Kolisch, R., Sprecher, A., 1997. PSPLIB - a project scheduling problem library: OR software - ORSEP operations research software exchange program. European J. Oper. Res. (ISSN: 0377-2217) 96 (1), 205–216. http://dx.doi.org/10.1016/S0377-2217(96)00170-1.

Mavrotas, G., 2009. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. Appl. Math. Comput. 213, 455–465. http://dx.doi.org/10.1016/j.amc.2009.03.037.

Miettinen, K., 1998. Nonlinear Multiobjective Optimization, first ed. Springer, New York, http://dx.doi.org/10.1007/978-1-4615-5563-6.

Nebro, A.J., Durillo, J.J., Luna, F., Dorronsoro, B., Alba, E., 2009. MOCell: A cellular genetic algorithm for multiobjective optimization. Int. J. Intell. Syst. 24 (7), 726–746. http://dx.doi.org/10.1002/int.20358.

Nebro, A.J., Durillo, J.J., Vergne, M., 2015. Redesigning the jmetal multi-objective optimization framework. In: Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, USA, pp. 1093–1100. http://dx.doi.org/10.1145/2739482.2768462.

Neumann, K., 1990. Stochastic Project Networks: Temporal Analysis, Scheduling and Cost Minimization. Vol. 344. Springer Berlin, Heidelberg, http://dx.doi.org/10.1007/978-3-642-61515-3.

Pritsker, A.A.B., Watters, L.J., Wolfe, P.M., 1969. Multiproject scheduling with limited resources: A zero-one programming approach. Manage. Sci. 16 (1), 93–108. http://dx.doi.org/10.1287/mnsc.16.1.93.

Qi, J.-J., Liu, Y.-J., Jiang, P., Guo, B., 2014. Schedule generation scheme for solving multi-mode resource availability cost problem by modified particle swarm optimization. J. Sched. 18, 285–298. http://dx.doi.org/10.1007/s10951-014-0374-0.

Said, L.B., Bechikh, S., Ghedira, K., 2010. The r-dominance: A new dominance relation for interactive evolutionary multicriteria decision making. IEEE Trans. Evol. Comput. 14 (5), 801–818. http://dx.doi.org/10.1109/TEVC.2010.2041060.

Salto, C., Alba, E., 2019. Cellular genetic algorithms: Understanding the behavior of using neighborhoods. Appl. Artif. Intell. 33 (10), 863–880. http://dx.doi.org/10.1080/08839514.2019.1646005.

Schlünz, E.B., Bokov, P.M., van Vuuren, J.H., 2016. A comparative study on multiobjective metaheuristics for solving constrained in-core fuel management optimisation problems. Comput. Oper. Res. 75, 174–190. http://dx.doi.org/10.1016/j.cor.2016.06.001.

Seada, H., Deb, K., 2015. U-NSGA-III: A unified evolutionary optimization procedure for single, multiple, and many objectives: Proof-of-principle results. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (Eds.), Evolutionary Multi-Criterion Optimization. Springer International Publishing, Cham, pp. 34–49. http://dx.doi.org/10.1007/978-3-319-15892-1_3.

Wang, J., Hu, X., Demeulemeester, E., Zhao, Y., 2021. A bi-objective robust resource allocation model for the RCPSP considering resource transfer costs. Int. J. Prod. Res. 59 (2), 367–387. http://dx.doi.org/10.1080/00207543.2019.1695168.

Whitley, L.D., 1993. Cellular genetic algorithms. In: International Conference on Genetic Algorithms. http://dx.doi.org/10.1007/978-981-10-7497-4_5.

Yepes-Borrero, J.C., Perea, F., Ruiz, R., Villa, F., 2021. Bi-objective parallel machine scheduling with additional resources during setups. European J. Oper. Res. (ISSN: 0377-2217) 292 (2), 443–455. http://dx.doi.org/10.1016/j.ejor.2020.10.052.

Zhang, Q., Li, H., 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. 11 (6), 712–731. http://dx.doi.org/10.1109/TEVC.2007.892759.

Zitzler, E., 1999. Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. Shaker Verlag, GmbH, Germany, ISBN: 3826568311.

Zitzler, E., Deb, K., Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. Evolut. Comput. (ISSN: 1063-6560) 8 (2), 173–195. http://dx.doi.org/10.1162/106365600568202.

Zitzler, E., Künzli, S., 2004. Indicator-based selection in multiobjective search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (Eds.), Parallel Problem Solving from Nature - PPSN VIII. Springer, Berlin, Heidelberg, ISBN: 978-3-540-30217-9, pp. 832–842.

Zitzler, E., Laumanns, M., Thiele, L., 2001. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Vol. 103. Technical report, ETH Zurich, Computer Engineering and Networks Laboratory, http://dx.doi.org/10.3929/ethz-a-004284029.

Zitzler, E., Thiele, L., 1999. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. 3 (4), 257–271. http://dx.doi.org/10.1109/4235.797969.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., da Fonseca, V., 2003. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Trans. Evol. Comput. 7 (2), 117–132. http://dx.doi.org/10.1109/TEVC.2003.810758.