INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2010; **84**:1344–1375
Published online 26 May 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nme.2944

Convergence speed in multi-objective metaheuristics: Efficiency criteria and empirical study

J. J. Durillo[1,*,†], A. J. Nebro[1], F. Luna[1], C. A. Coello Coello[2] and E. Alba[1]

[1]*Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, Spain*
[2]*Department of Computer Science, CINVESTAV-IPN, Mexico*

SUMMARY

Solving optimization problems using a reduced number of objective function evaluations is an open issue in the design of multi-objective optimization metaheuristics. The usual approach to analyze the behavior of such techniques is to choose a benchmark of known problems, to perform a predetermined number of function evaluations, and then, apply a set of performance indicators in order to assess the quality of the solutions obtained. However, this sort of methodology does not provide any insights of the efficiency of each algorithm. Here, efficiency is defined as the effort required by a multi-objective metaheuristic to obtain a set of non-dominated solutions that is satisfactory to the user, according to some pre-defined criterion. Indeed, the type of solutions of interest to the user may vary depending on the specific characteristics of the problem being solved. In this paper, the convergence speed of seven state-of-the-art multi-objective metaheuristics is analyzed, according to three pre-defined efficiency criteria. Our empirical study shows that SMPSO (based on a particle swarm optimizer) is found to be the best overall algorithm on the test problems adopted when considering the three efficiency criteria.

Received 9 September 2009; Revised 24 February 2010; Accepted 17 April 2010

KEY WORDS: multi-objective optimization; metaheuristics; efficiency; convergence

*Correspondence to: J. J. Durillo, Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, Spain.
†E-mail: durillo@lcc.uma.es

Contract/grant sponsor: CONACyT; contract/grant number: 103570
Contract/grant sponsor: UMI-LAFMIA 3175 CNRS at CINVESTAV-IPN
Contract/grant sponsor: Consejería de Innovación, Ciencia y Empresa
Contract/grant sponsor: Junta de Andalucía; contract/grant number: P07-TIC-03044
Contract/grant sponsor: Spanish Ministry of Science and Innovation and FEDER; contract/grant number: TIN2008-06491-C04-01
Contract/grant sponsor: Spanish government; contract/grant number: AP-2006-03349
Contract/grant sponsor: Spanish Ministry of Education and Science and FEDER; contract/grant number: TIN2005-08818-C04-01

1. INTRODUCTION

Many of the problems faced in engineering and other disciplines of knowledge are optimization problems. A large number of them present features such as non-linear formulations, constraints, and NP-hard complexity, making them difficult to solve.

Typically, there are two different kinds of techniques for solving optimization problems. On the one hand, exact techniques, which allow us to find the optimal solution of a given problem, but they become impractical when they have to deal with NP-hard problems because they may need a huge amount of time and/or memory to solve them. On the other hand, there are non-exact techniques, such as metaheuristics [1, 2], which sacrifice the warranty of finding the optimal solution for the sake of getting near-optimal solutions in a reasonable amount of time.

However, an increasing number of real-world problems need computationally expensive methods for computing their objective functions, or rely on complex time-consuming software simulation tools, and are, consequently, very time-consuming. Furthermore, many of these problems have hard time restrictions; hence, using metaheuristics requiring a high number of function evaluations for solving them may become impractical.

There exist different ways of dealing with this issue. One approach is to use parallelism, which allows not only a reduction of the required computational time, but also a possible improvement in the quality of the obtained solutions [3]. A second alternative could be to apply surrogate models [4, 5] (i.e. models that are built to approximate computationally expensive functions). A third choice is to design metaheuristics requiring a lower number of function evaluations to converge to the optimal solution.[‡] In this paper, we will focus on this last situation.

Herein, we are interested in problems that involve the optimization of more than one criterion at the same time, being these criteria in conflict with each other. These types of problems are known as multi-objective optimization problems (MOPs). When solving MOPs, we are interested in the best possible trade-offs (or compromises) among the different objectives (i.e. solutions in which it is not possible to improve one objective without worsening another). This is the case, because, in the absence of any further information, all the objectives of an MOP are considered as equally important. Thus, the solution to an MOP is not a single solution, but a set of them, which is called the *Pareto optimal set*. The vectors corresponding to the solutions contained in the Pareto optimal set are called *non-dominated*. The objective function values corresponding to the solutions contained in the Pareto optimal set are collectively known as the *Pareto front* [6].

Two main goals when using metaheuristics to solve an MOP are: convergence and diversity. The first goal refers to finding a set of solutions as close as possible to the true Pareto front, whereas the second goal implies that the solutions obtained should be as uniformly distributed along the Pareto front as possible [7]. However, these goals conflict with the needs that arise when solving complex real-world problems. The main motivation for the work reported in this paper comes precisely from such needs imposed by computationally demanding applications. Such needs can

[‡]Throughout this paper, we will use the term *convergence* to refer to the generation of approximations of the Pareto optimal set of a problem that is sufficiently accurate according to some pre-defined criterion. Thus, this notion of convergence is not related to limit behavior or to any warranties of generating the true Pareto optimal set of a problem.

DOI: 10.1002/nme

be summarized as follows:

- we only have a limited amount of time for solving the problem,
- we wish to obtain at least one or a few Pareto optimal solutions within the allowable time,
- we want to obtain as many Pareto optimal solutions as possible even if that implies to sacrifice their diversity, and
- we want to provide the decision maker with a reasonable approximation of the true Pareto front in a fast manner.

A wide variety of metaheuristics have been proposed for solving MOPs [6, 7]. Typically, their performance has been assessed using well-known benchmarks, and comparing their results on the basis of different indicators that measure the quality of the obtained sets of solutions (e.g. the hypervolume [8], the additive epsilon indicator [9], the generational distance [8], and the spread [7], among others) after performing the same (pre-defined) number of function evaluations. However, as indicated before, in real-world applications, the decision maker might not be interested in obtaining a Pareto front with many solutions that are uniformly distributed, and could be satisfied with obtaining a small set of Pareto optimal solutions (or a reasonably good approximation of them), as long as their generation consumes as few function evaluations as possible.

Indeed, a hot research topic nowadays is the design of efficient techniques that require a small number of function evaluations to find reasonably good approximations of the Pareto optimal set of MOPs of moderate dimensionality, and several papers that address this issue have been published in the last few years. In [10], Coello and Pulido described a micro Genetic Algorithm (micro-GA), which is a genetic algorithm (GA) with a very small population size (only four individuals) and a restart process. This algorithm is able to produce an important portion of the Pareto front with a very low computational cost. A revised version of this algorithm is described in [11]; this new version of the algorithm is able to automatically decide what parameter values to adopt at any time during the search (i.e. it is self-adaptive). Santana-Quintero *et al.* proposed in [12] a particle swarm optimization algorithm hybridized with rough sets theory, which is used for solving MOPs requiring only 4000 fitness function evaluations, which is a low number compared with today's standards in the specialized literature. In a related paper, Hernández-Díaz *et al.* [13] proposed a hybrid algorithm between a multi-objective differential evolution approach and rough sets theory, which only performs 3000 fitness function evaluations. In [14], a more efficient multi-objective PSO algorithm is presented; this algorithm is able to provide accurate Pareto fronts of MOPs computing only 2000 fitness function evaluations. Eskandari *et al.* explored in [15] the use of dynamic population sizing to design an algorithm called FastPGA, which outperforms NSGA-II, the reference algorithm in the field, when computing less than 10 000 fitness function evaluations. Knowles [16] proposed parEGO, which can produce reasonably good approximations to MOPs of low dimensionality performing only 250 fitness function evaluations. In [17], Sindhya *et al.* presented a local search method that borrows some multi-criteria decision making concepts, and included them in the form of a search operator within NSGA-II. The outcome is an approach that presents faster convergence and produces more accurate solutions than the original NSGA-II.

Our purpose in this work is not to design faster algorithms, but to study the convergence speed (i.e. the efficiency) of general purpose multi-objective metaheuristics. We carried out a previous study of this issue in [18], where six multi-objective algorithms were compared. In that work, we used as our convergence speed criterion the obtention of a Pareto front with a hypervolume indicator value higher than 98% of the hypervolume of the true Pareto front, i.e. to yield an approximation set with both good convergence and diversity. To the best of our knowledge, apart

DOI: 10.1002/nme

from this paper, very few studies exist in the specialized literature that analyze and compare the convergence speed of different multi-objective optimization techniques when converging to the true Pareto front.

Here, we intend to carry out further research in this line. While in single-objective optimization the convergence speed can be defined as the number of evaluations needed to find the optimal (or a near-optimal) solution to the target problem, in multi-objective optimization this idea is not directly applicable due to the fact that, in this context, we aim to find a set of solutions, and not only one. In this paper, we define three different efficiency criteria in order to consider that an algorithm has successfully solved an MOP attending to different needs of the decision maker that arise in the real-world scenarios. In particular, we have taken into account three issues:

- to find a certain (pre-defined) number of Pareto optimal solutions,
- to obtain a Pareto front with a certain (pre-defined) convergence level, determined by a convergence quality indicator, and
- to obtain a set of solutions considering both convergence and diversity, as in [18].

Our main contribution is to analyze the computational effort, measured as the number of objective function evaluations required by a set of multi-objective metaheuristics according to the previously defined criteria. Such studies do not practically exist in the specialized literature and are a first step to allow a characterization of multi-objective metaheuristics that allows us to identify in what types of problems they are expected to perform better. Such type of study is of great importance, if we consider the theoretical limitations imposed by the 'No Free Lunch' theorems for search [19], but are rarely addressed in today's literature.

The set adopted for our study is composed of seven state-of-the-art techniques covering a wide range of different techniques: two GAs (NSGA-II [20] and SPEA2 [21]), an Evolution Strategy (PAES [22]), a Differential Evolution algorithm (GDE3 [23]), a Particle Swarm Optimization algorithm (SMPSO [24]), a Scatter Search method (AbYSS [25]), and a cellular Genetic Algorithm (cGA) (MOCell [26]). Many benchmark problems exist in the literature, such as the Zitzler–Deb–Thiele (ZDT) test suite [27], the Deb–Thiele–Laumanns–Zitzler (DTLZ) problem family [28], the Walking–Fish-Group (WFG) test problems [29], the problems proposed by Li and Zhang [30], and those described by Zhang *et al.* [31]. Given that we are going to carry out an extensive study, we have selected the two most well-known and used benchmarks, namely ZDT and DTLZ.

It is worth remarking that our purpose is not to find the best possible configurations for making these algorithms to converge faster. On the contrary, we intend to analyze their behavior by using their typical settings. This way, researchers familiar with some of these techniques can get useful information in order to apply them to solve MOPs in cases in which it is of particular interest to be able to obtain Pareto optimal solutions as quickly as possible. On the other hand, as no studies exist that analyze in depth the efficiency of multi-objective metaheuristics according to the three convergence criteria defined here, the results of this paper can also be used as a reference for future works related to the development of techniques aimed at improving the efficiency of multi-objective metaheuristics.

The remainder of this paper is organized as follows. The next section provides some background on multi-objective optimization. In Section 3, we describe and justify the different criteria that we have taken into account for considering that a problem has been successfully solved. All the algorithms evaluated and the methodology that we have followed are presented in Section 4. The analysis of the obtained results is provided in Section 5, whereas Section 6 is aimed at deepening

DOI: 10.1002/nme

into these results. Finally, Section 7 presents a summary and our main conclusions as well as different possible lines of future research.

2. MULTI-OBJECTIVE OPTIMIZATION BACKGROUND

In this section, we include some background on multi-objective optimization. We first define basic concepts, such as Pareto optimality, Pareto dominance, Pareto optimal set, and Pareto front. In these definitions we are assuming, without loss of generality, the minimization of all the objectives.

A general MOP can be formally defined as follows:

Definition 1 (*MOP*)

$$\text{minimize} \quad \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})] \tag{1}$$

subject to

$$g_i(\vec{x}) \leqslant 0, \quad i = 1, 2, \ldots, m \tag{2}$$

$$h_i(\vec{x}) = 0, \quad i = 1, 2, \ldots, p \tag{3}$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$ is the vector of decision variables, $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, k$ are the objective functions and $g_i, h_j : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, $j = 1, \ldots, p$ are the constraint functions of the problem.

Definition 2 (*Pareto dominance*)
Given two vectors $\vec{x}, \vec{y} \in \mathbb{R}^k$, we say that $\vec{x} \leqslant \vec{y}$ if $x_i \leqslant y_i$ for $i = 1, \ldots, k$, and that $\vec{x}$ *dominates* $\vec{y}$ (denoted by $\vec{x} \prec \vec{y}$) if $\vec{x} \leqslant \vec{y}$ and $\vec{x} \neq \vec{y}$.

Definition 3 (*Non-dominance*)
We say that a vector of decision variables $\vec{x} \in \mathscr{X} \subset \mathbb{R}^n$ is *non-dominated* with respect to $\mathscr{X}$, if there does not exist another $\vec{x}' \in \mathscr{X}$ such that $\vec{f}(\vec{x}') \prec \vec{f}(\vec{x})$.

Definition 4 (*Pareto optimality*)
We say that a vector of decision variables $\vec{x}^* \in \mathscr{F} \subset \mathbb{R}^n$ ($\mathscr{F}$ is the feasible region) is *Pareto optimal* if it is non-dominated with respect to $\mathscr{F}$.

Definition 5 (*Pareto optimal set*)
The *Pareto optimal set* $\mathscr{P}^*$ is defined by

$$\mathscr{P}^* = \{\vec{x} \in \mathscr{F} \,|\, \vec{x} \text{ is Pareto-optimal}\}$$

Definition 6 (*Pareto front*)
The *Pareto front* $\mathscr{PF}^*$ is defined by

$$\mathscr{PF}^* = \{\vec{f}(\vec{x}) \in \mathbb{R}^k \,|\, \vec{x} \in \mathscr{P}^*\}$$

DOI: 10.1002/nme

3. CONVERGENCE CRITERIA

This section is devoted to presenting and justifying the three criteria that we have used for analyzing the convergence speed of multi-objective metaheuristics.

Given that the solutions of the test problems used in this work are known beforehand, it is possible to compute their true Pareto fronts. Although these fronts are continuous, and hence, consist of an infinite number of points, in the field of multi-objective optimization it is usual to represent these fronts by a finite number of them, i.e. a finite subset of the true Pareto front.[§] When generating solutions with a multi-objective metaheuristic, such solutions are compared (in terms of the Pareto dominance) to the contents of the true Pareto front. If the generated solutions are non-dominated with respect to the true Pareto front, then they are considered to be part of such a true Pareto front. As it can be seen, no actual direct matching of solutions takes place, but only a comparison in terms of the Pareto dominance. Thus, in order to assess convergence, each algorithm is executed until a maximum pre-defined number of function evaluations have been performed. Then, we determine at which number of evaluations one or more convergence criteria were met. Clearly, it is necessary to define an upper bound on the maximum number of function evaluations that will be performed. Thus, we will consider that an algorithm has not converged if it has reached such an upper bound on its number of evaluations without complying with any of the convergence criteria previously defined.

As stated in the introduction, the particularities that arise when real-world problems are solved may lead to different types of criteria for assessing the convergence speed of multi-objective metaheuristics. These criteria vary from obtaining one or a few optimal solutions to obtaining an accurate approximation, in terms of diversity and/or convergence, of the Pareto optimal set of the problem. Thus, we have considered three criteria trying to cover all these different scenarios.

3.1. Criterion 1: *to obtain a given number of optimal solutions*

This is the simplest of the three criteria adopted in this paper. It lies in determining the evaluations at which an algorithm produces a given number of solutions that are included in the true front of the problem being solved (i.e. the Pareto optimal solutions of the problem). Thus, we have implemented this criterion by counting the number of computed solutions that are non-dominated with respect to the Pareto front. Herein, we have considered to obtain 1, 5, 10, 20, 50, and 100 Pareto optimal solutions. The upper limit of 100 solutions has been chosen because that it is a size commonly adopted in the specialized literature for the final set of solutions reported.

It is worth mentioning that in this criterion, the diversity of the solutions contained into the approximated fronts is not taken into account. Thus, fronts containing a lower number of optimal solutions could have a better diversity than others having a higher number of Pareto optimal solutions.

To clarify this situation, Figure 1 shows examples of approximation sets containing different numbers of Pareto optimal solutions of a problem having a convex Pareto front. We can observe that fronts (a) and (d), having 1 and 20 optimal solutions, respectively, present a poor diversity. Fronts (b), (c), (e), and (f) have a better diversity than fronts (a) and (d); they have 5, 10, 50, and 100 optimal solutions, respectively. Note that front (d) contains four times the number of Pareto

[§] This subset will be referred to as the *true Pareto front* in the remainder of the paper.

DOI: 10.1002/nme

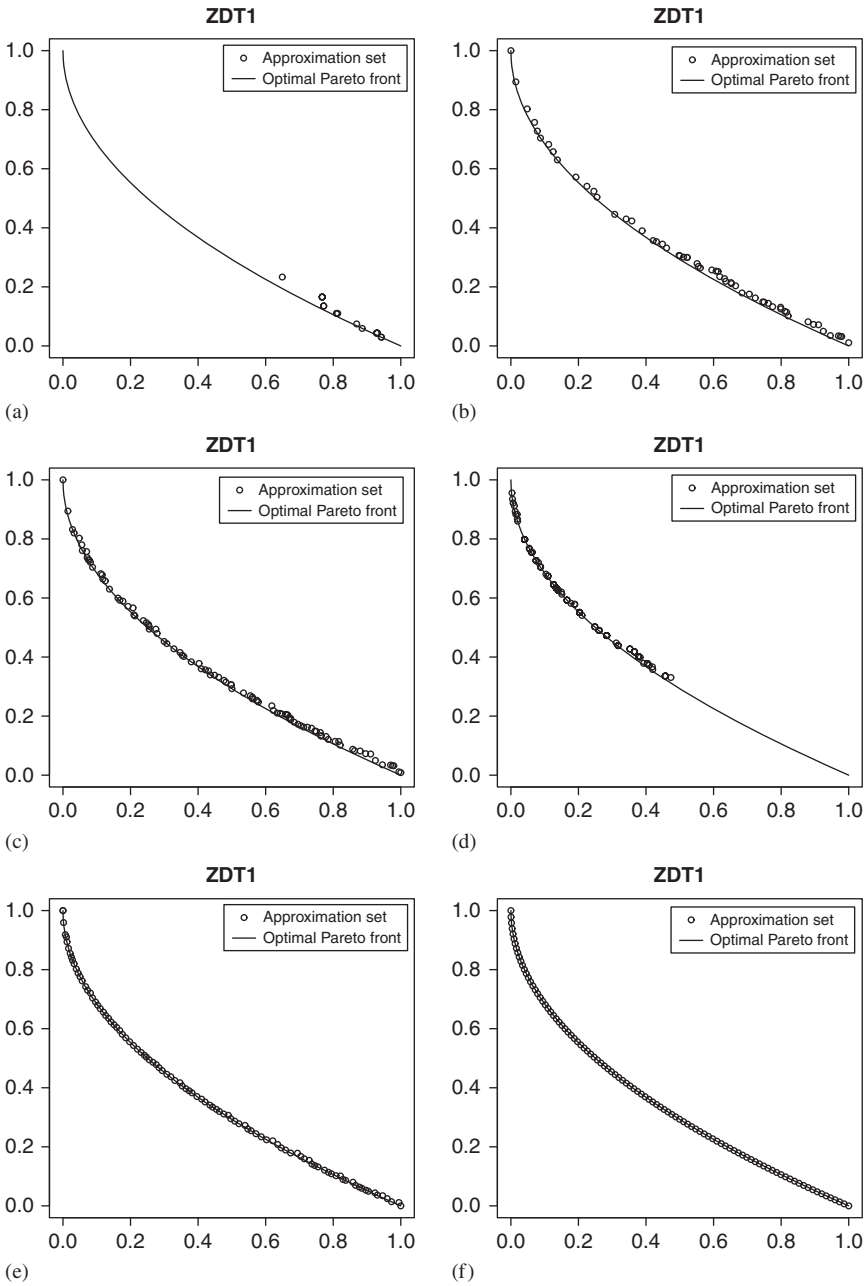

Figure 1. Examples of Pareto fronts. From left to right, from top to bottom: approximate representation of a convex Pareto front (represented by the small circles) including 1, 5, 10, 20, 50, and 100 solutions belonging to the true Pareto front (represented by a continuous line) of the problem being solved.

DOI: 10.1002/nme

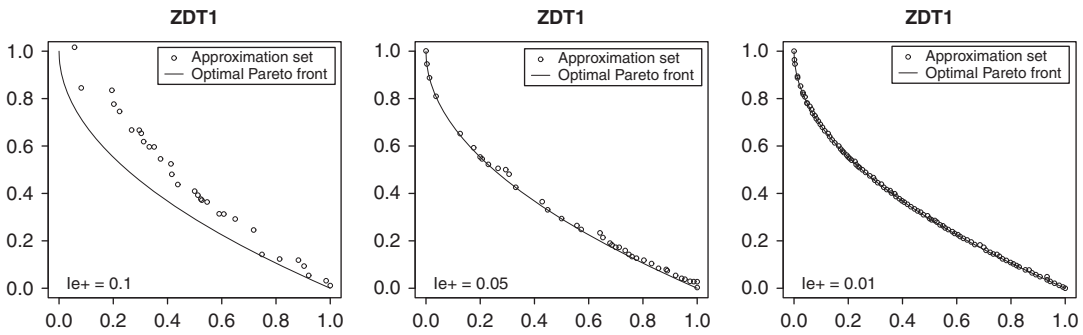

Figure 2. Examples of Pareto fronts with different values of the Additive Epsilon indicator. From left to right, we show Pareto fronts with indicator values of 0.1, 0.05, and 0.01, respectively.

optimal solutions included in front (b); however, the diversity of front (b) is much better than the diversity of front (d).

3.2. Criterion 2: using the epsilon indicator

To determine how close are the solutions obtained by multi-objective metaheuristics to the true Pareto front of a problem, performance indicators that assess convergence are usually applied. Thus, our second criteria is based on the use of one of such indicators.

We have chosen the unary Additive Epsilon ($I^1_{\varepsilon+}$) indicator, which is a Pareto-compliant indicator proposed by Zitzler and coworkers in [9]. It is defined as follows. Given an approximation set of a problem, A, the $I^1_{\varepsilon+}$ indicator is a measure of the smallest distance one would need to translate every point in A so that it dominates the true Pareto front of the problem. More formally, given $\vec{z}^1=(z^1_1,\ldots,z^1_n)$ and $\vec{z}^2=(z^2_1,\ldots,z^2_n)$, where n is the number of objectives,

$$I^1_{\varepsilon+}(A)=\inf_{\varepsilon\in\mathbb{R}}\{\forall\vec{z}^2\in\mathscr{P}\mathscr{F}^*\exists\vec{z}^1\in A:\vec{z}^1\prec_\varepsilon\vec{z}^2\} \tag{4}$$

where $\vec{z}^1\prec_\varepsilon\vec{z}^2$ if and only if $\forall 1\leqslant i\leqslant n:z^1_i<\varepsilon+z^2_i$. We apply this indicator by using normalized objective function values.

Since the Epsilon indicator is a measure of closeness between a set of solutions and the true Pareto front, we need to fix a value for it, which allows us to consider that a problem has converged to the true Pareto front. Figure 2 shows different approximations to the true Pareto front of a problem having the following values of the indicator: 0.1, 0.05, and 0.01. We can observe in this figure that the front having an $I^1_{\varepsilon+}$ value of 0.01 is close enough to the true Pareto front, and it could be considered as an accurate approximation in terms of convergence.

The fronts having values of 0.1 and 0.05 are not so satisfactory, but they could be good enough for a decision maker (particularly, the one having the value 0.05 in the indicator) if they can be obtained quickly. This experiment has been carried out with all the problems included here obtaining the same conclusions; thus, we have decided to include the three values of $I^1_{\varepsilon+}$ in our study.

In this criterion, as happened with the previous one, the diversity of solutions is not taken into account. However, there is an important difference between both criteria: this criterion implies that all the solutions contained in the approximation are closer than a given $I^1_\varepsilon+$ value to the

DOI: 10.1002/nme

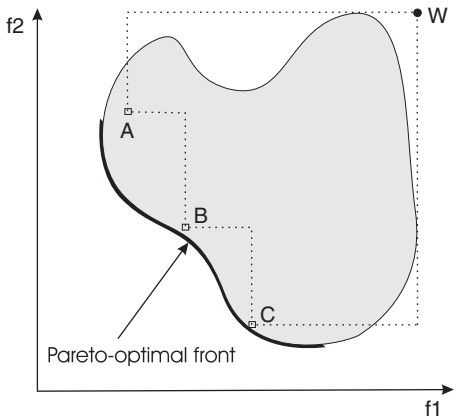

Figure 3. The hypervolume enclosed by the non-dominated solutions.

Pareto optimal front, whereas Criterion 1 does not give information about the non-optimal solutions contained in the approximation.

3.3. Criterion 3: using the hypervolume quality indicator

The main drawback of using the two previous indicators to check the convergence speed of a multi-objective algorithm is that none of them takes into account the diversity of solutions. Since the decision maker might be interested in the approximations of the Pareto optimal set that keep a balance between convergence and diversity, we have also used a criterion based on the Hypervolume (I_{HV}) [8] quality indicator, which measures somehow these two properties of a Pareto front.

The hypervolume is also a Pareto-compliant indicator, and it calculates the volume (in objective function space) covered by the members of a non-dominated set of solutions Q (the region enclosed within the discontinuous line in Figure 3, $Q=\{A, B, C\}$) for problems where all objectives are to be minimized. Mathematically, for each solution $i \in Q$, a hypercube v_i is constructed with a reference point W and the solution i as the diagonal corners of the hypercube. The reference point can be found simply by constructing a vector of worst objective function values. Thereafter, a union of all hypercubes is found and its hypervolume (I_{HV}) is calculated:

$$I_{HV} = \text{volume}\left(\bigcup_{i=1}^{|Q|} v_i\right). \tag{5}$$

Algorithms with larger I_{HV} values are desirable. Since this indicator is not free from arbitrary scaling of objectives, we evaluate it by using normalized objective function values.

To ensure that an algorithm has successfully solved a problem, we have considered that it has to find a set of solutions whose hypervolume value is higher than a pre-defined percentage of the hypervolume value of the true Pareto front. Since no theoretical basis for obtaining such percentages exists, we rely on some preliminary experimentation in order to determine them. Thus, in Figure 4, we show examples of different fronts with different percentages of I_{HV}. In this figure, we can see that a front with a hypervolume of 98.26% represents, in this example, a reasonably

DOI: 10.1002/nme

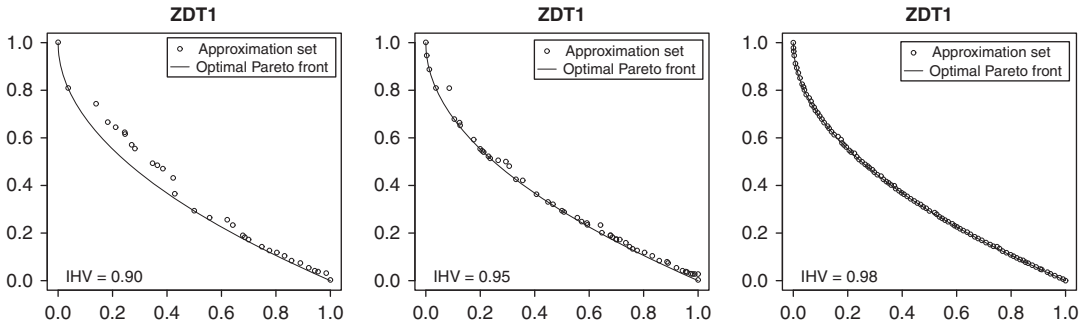

Figure 4. Examples of fronts with different hypervolume values. From left to right, we show the Pareto fronts having an I_{HV} value of 0.90, 0.95, and 0.98, respectively.

good approximation to the true Pareto front in terms of convergence and diversity of solutions. Similar preliminary experiments have been carried out for each of the test problems considered in this work, and for all of them we found that a value between 98 and 99% of the hypervolume of the true Pareto front resulted in an accurate representation of such front. Thus, we have adopted these percentages of the hypervolume of the true Pareto front as our criterion to consider that an algorithm has converged. Clearly, problems with a higher number of objectives will require further experiments in order to determine the appropriate percentages of I_{HV} to be adopted.

To illustrate the results that can be obtained by applying these efficiency criteria, we include in Figure 5 examples of approximation sets of four problems having different shapes (convex, disconnected, linear, and concave). These fronts correspond, respectively, to the well-known test problems ZDT1, ZDT3, DTLZ1, and DTLZ3 (see next section for further details). For each problem, three different solution sets are presented, and the values of the three criteria are included in the bottom left-hand side of each picture.

4. EXPERIMENTATION

In this section, we present the seven metaheuristics evaluated. Then, we describe the parameter settings used in the experiments, as well as the benchmark problems that we have used, and the methodology that we have followed in the tests.

4.1. Multi-objective optimization algorithms

The metaheuristics that we have considered in this study are briefly described in this section. We have used the implementation of these algorithms provided by jMetal [32], a Java-based framework for multi-objective optimization using metaheuristics.[¶]

The NSGA-II algorithm was proposed by Deb *et al.* [20]. It is a GA based on obtaining a new population from the original one by applying the typical genetic operators (selection, crossover, and mutation); then, the individuals in the two populations are sorted according to their rank,

[¶]jMetal is freely available for download at the following URL: `http://jmetal.sourceforge.net`.

DOI: 10.1002/nme

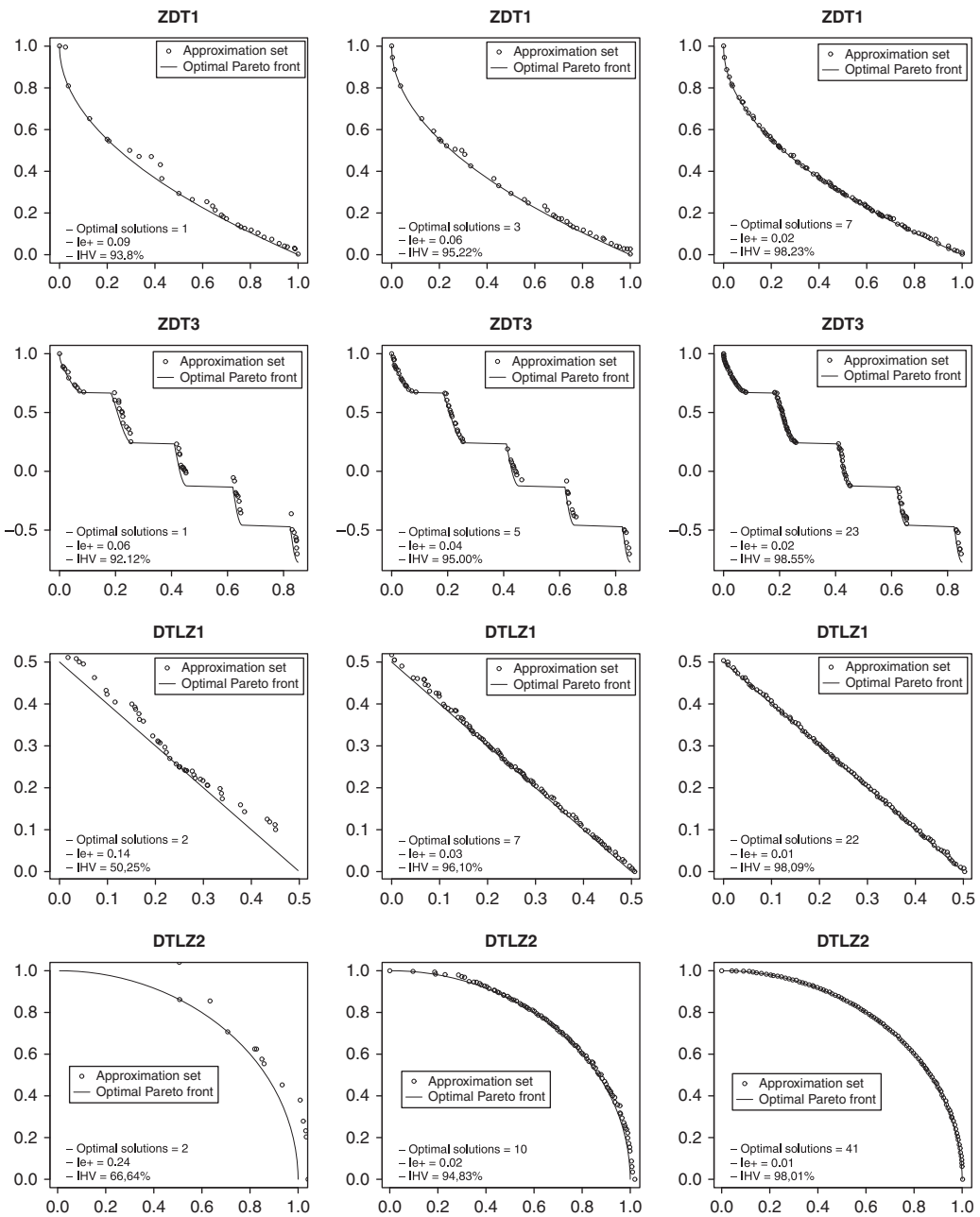

Figure 5. Examples of approximation sets and the values of the three criteria.

and the best solutions are chosen to create a new population. In case of having to select some individuals with the same rank, a density estimator based on measuring the crowding distance to the surrounding individuals belonging to the same rank is used to get the most promising solutions.

DOI: 10.1002/nme

SPEA2 was proposed by Zitler *et al.* in [33]. In this algorithm, each individual has a fitness value that is the sum of its strength raw fitness plus a density estimation value. The algorithm applies the selection, crossover, and mutation operators to fill up an archive of individuals; then, the non-dominated individuals of both the original population and the archive are copied into a new population. If the number of non-dominated individuals is greater than the population size, a truncation operator based on calculating the distances to the kth nearest neighbor is used. This way, the individuals having the minimum distance to any other individual are chosen.

PAES is a metaheuristic proposed by Knowles and Corne [22]. The algorithm is based on a simple $(1+1)$ evolution strategy. To find diverse solutions along the Pareto front, PAES uses an external archive of non-dominated solutions, which is also used to decide about the new candidate solutions. An adaptive grid is used as a density estimator in the archive. We have used a real-coded version of PAES, applying a polynomial mutation operator.

GDE3 [23] is an improved version of the Generalized Differential Evolution (GDE) algorithm [34]. Its behavior is similar to the NSGA-II procedure, but the usual recombination operator is replaced with a differential evolution operator. It starts with a population of random solutions, which becomes the current population. At each generation, an offspring population is created using the differential evolution operators; then, the current population for the next generation is updated using the solutions of both, the offspring and the current populations. Before proceeding to the next generation, the size of the population is reduced using non-dominated sorting and a pruning technique aimed at preserving diversity, in a similar way as in the NSGA-II, although the pruning process adopted in GDE3 modifies the crowding distance of the NSGA-II in order to solve some of its drawbacks when dealing with problems having more than two objectives.

Speed-constrained Multi-objective PSO (SMPSO) is a particle swarm optimization algorithm for solving MOPs [24]. Its main features include the use of the concept of crowding distance adopted by NSGA-II to filter out leader solutions, which are stored in an archive, and the use of mutation operators to accelerate the convergence of the swarm. In addition, SMPSO also incorporates a mechanism to limit the velocity of the particles and the use of polynomial mutation as a turbulence factor.

MOCell [26] is a cGA. As many multi-objective metaheuristics, it includes an external archive to store the non-dominated solutions found so far. This archive is bounded and uses the crowding distance of NSGA-II to keep diversity along the Pareto Front. We have used here an asynchronous version of MOCell, called aMOCell4 in [35], in which the cells are explored sequentially (asynchronously). The selection is based on taking an individual from the neighborhood of the current solution (called *cell* in cGAs) and another one chosen randomly from the archive. After applying the genetic crossover and mutation operators, the new offspring is compared with the current one, replacing it if better; if both solutions are non-dominated, the worst individual in the neighborhood is replaced by the current one. In these two cases, the new individual is inserted into the archive.

AbYSS is an adaptation of the *scatter search* metaheuristic to the multi-objective domain [25]. It uses an external archive similar to the one employed by MOCell. The algorithm incorporates operators of the evolutionary algorithms domain, including polynomial mutation and simulated binary crossover (SBX) in the improvement and solution combination methods, respectively.

4.2. Parameters settings

As commented in the introduction, we chose a set of parameters settings that have been used in the previous studies (see the references in Table I), without spending any effort in trying to find

DOI: 10.1002/nme

Table I. Parameterization.

NSGA-II [*20*]	
Population size	100 individuals
Selection of parents	Binary tournament+binary tournament
Recombination	Simulated binary, $p_c = 0.9$
Mutation	Polynomial, $p_m = 1.0/L$ (L = individual length)
SPEA2 [*33*]	
Population size	100 individuals
Selection of parents	Binary tournament+binary tournament
Recombination	Simulated binary, $p_c = 0.9$
Mutation	Polynomial, $p_m = 1.0/L$ (L = individual length)
PAES [*22*]	
Mutation	Polynomial, $p_m = 1.0/L$ (L = individual length)
Archive size	100
Grid size	32×32
SMPSO [*24*]	
Particles	100 particles
Mutation	Polynomial
Leaders size	100
GDE3 [*23*]	
Population size	100 individuals
Recombination	Differential Evolution, CR = 0.1, $F = 0.5$
MOCell [*35*]	
Population size	100 individuals (10×10)
Neighborhood	1-hop neighbors (8 surrounding solutions)
Selection of parents	Binary tournament+binary tournament
Recombination	Simulated binary, $p_c = 0.9$
Mutation	Polynomial, $p_m = 1.0/L$ (L = individual length)
Archive size	100 individuals
AbYSS [*25*]	
Population size	20 individuals
Reference set size	10+10
Recombination	Simulated binary, $p_c = 1.0$
Mutation (local search)	Polynomial, $p_m = 1.0/L$ (L = individual length)
Archive size	100 individuals

the best possible configurations. All the GAs (NSGA-II, SPEA2, and MOCell) and GDE3 use an internal population of size equal to 100; the size of the archive is also 100 in PAES, SMPSO, GDE3, MOCell, and AbYSS. SMPSO has been configured with 100 particles. For AbYSS, the population and the reference set have a size of 20 solutions.

In the GAs, we have used SBX and polynomial-based mutation (for details on these two operators, see [7]) as operators for crossover and mutation, respectively. The distribution indexes for both operators are $\eta_c = 20$ and $\eta_m = 20$, respectively. The crossover probability is $p_c = 0.9$ and the mutation probability is $p_m = 1/L$, where L is the number of decision variables. In PAES, we have also used the polynomial mutation operator, with the same distribution index, and an adaptive

DOI: 10.1002/nme

Table II. Summary of characteristics of the problems belonging to the ZDT and DTLZ benchmarks. All the problems are bi-objective.

Name	Domain	Number of variables	Number of objectives	Type
ZDT1	Continuous	30	2	Convex
ZDT2	Continuous	30	2	Concave
ZDT3	Continuous	30	2	Concave, disconnected
ZDT4	Continuous	10	2	Convex, multimodal
ZDT6	Continuous	10	2	Concave, non-uniformly spaced
DTLZ1	Continuous	7	2	Linear, multimodal
DTLZ2	Continuous	12	2	Concave
DTLZ3	Continuous	12	2	Concave, multimodal
DTLZ4	Continuous	12	2	Concave
DTLZ5	Continuous	12	2	Degenerate
DTLZ6	Continuous	12	2	Degenerate
DTLZ7	Continuous	22	2	Disconnected

grid of size 32×32. AbYSS uses polynomial mutation in the improvement method and SBX in the solution combination method. GDE3 uses 0.1 and 0.5 as values for parameters CR and F, respectively [23]. SMPSO also applies polynomial mutation to 15% of the particles. A detailed description of the parameters settings adopted for each approach is included in Table I.

4.3. Benchmark problems

We describe here the different sets of problems addressed in this work. These problems are well-known, and have been used in many comparative studies in this area.

The families of problems adopted are the following:

- *ZDT test suite*: This benchmark is composed of five bi-objective problems [27]: ZDT1 (convex), ZDT2 (concave), ZDT3 (concave, disconnected), ZDT4 (convex, multimodal), and ZDT6 (concave, non-uniformly spaced). These problems are scalable according to the number of decision variables.
- *DTLZ test suite*: The problems of this family are scalable both in the number of variables and in the number of objectives [28]. It is composed of the following seven problems: DTLZ1 (linear, multimodal), DTLZ2 (concave), DTLZ3 (concave, multimodal), DTLZ4 (concave), DTLZ5-6 (degenerate), and DTLZ7 (disconnected).

In this work, we have used the bi-objective formulation of the DTLZ problem family. A total of 12 MOPs are used to evaluate the seven metaheuristics. The number of variables and the application domain of each problem can be found in Table II, which summarizes the main characteristics of the problems considered in this work.

4.4. Methodology

The underlying idea of the experiments is to know at which number of evaluations the metaheuristics meet the considered convergence criteria. Our approach has been to perform a maximum of 1 000 000 function evaluations per experiment, and to check the stopping conditions at every 100 evaluations (that is, each iteration in the population-based metaheuristics). Therefore, in NSGA-II, SPEA2, and GDE3 we have considered the non-dominated solutions at each generation; in PAES,

DOI: 10.1002/nme

AbYSS, and MOCell, we have considered the external population and, in SMPSO, we have considered the leaders archive. When an algorithm is unable to obtain a front fulfilling any of the termination conditions upon reaching the maximum number of function evaluations, we consider that it has failed in solving the problem for that criterion.

We have executed 100 independent runs for each algorithm and each problem instance. Since we are dealing with stochastic algorithms, we need to perform a statistical analysis of the obtained results to compare them with a certain level of confidence. Next, we describe the statistical test that we have carried out for assuring this. First, a Kolmogorov–Smirnov test is performed in order to check whether the values of the results follow a normal (Gaussian) distribution or not. If so, the Levene test checks for the homogeneity of the variances. If samples have equal variance (positive Levene test), an ANOVA test is done; otherwise, we perform a Welch test. For non-Gaussian distributions, the non-parametric Kruskal–Wallis test is used to compare the medians of the algorithms.

We always consider in this work a confidence level of 95% (i.e. significance level of 5% or p-value under 0.05) in the statistical tests, which means that the differences are unlikely to have occurred by chance with a probability of 95%. Those tests in which the statistical confidence has been achieved are marked with '+' in the last row in the tables containing the results; conversely, '−' means that we cannot assure anything about the statistical confidence of the results ($p-$value>0.05). For the sake of homogeneity in the presentation of the results, all the tables include the median, $\tilde{x}$, and the interquartile range, IQR, as measures of location (or central tendency) and statistical dispersion, respectively.

To further analyze the results statistically, in some cases, we have also included a post-hoc testing phase that allows for a multiple comparison of samples [36]. We have used the `Wilcoxon` test for that purpose. This way, we can make pairwise comparisons between algorithms to know about the significance of the obtained data. To avoid including excessive information, in these tests we have included only the results of the three best performing algorithms.

5. ANALYSIS OF THE RESULTS

This section is aimed at presenting the obtained results. The discussion is organized according to the three convergence criteria previously defined: first, we study the computational effort that the algorithms require for computing different numbers of Pareto optimal solutions; then, we analyze the number of evaluations that they require for computing an accurate approximation to the true Pareto front considering the I_ε^1+ and the I_{HV} quality indicators.

To ease the analysis of the results in all the tables, the cells containing the lowest number of function evaluations have a gray-colored background. There are two gray levels: the darker gray indicates the best value, whereas the lighter gray points out the second best value.

5.1. Pareto optimal solutions

Tables III and IV include the number of function evaluations that the algorithms have required for computing 1, 5, 10, 20, 50, and 100 Pareto optimal solutions in the ZDT and DTLZ families, respectively. Starting with the first table, the gray-colored background reveals that PAES and SMPSO have been the fastest algorithms in computing such a number of solutions in the ZDT benchmark. Indeed, if we deepen into the obtained results, both algorithms have been the fastest or

DOI: 10.1002/nme

Table III. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for reaching Pareto optimal points in the ZTD benchmark (cells with dark and light backgrounds indicate the best and the second best values, respectively). PAES and SMPSO provide the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	n	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
ZDT1	1	$1.1e4_{1.7e3}$	$1.3e4_{1.6e3}$	$3.3e3_{1.0e3}$	$9.6e3_{7.0e2}$	$6.2e3_{1.4e3}$	$1.1e4_{1.4e3}$	$1.1e4_{1.6e3}$	+
	5	$2.0e4_{1.8e3}$	$2.2e4_{2.2e3}$	$3.7e3_{1.1e3}$	$1.0e4_{7.0e2}$	$6.6e3_{1.5e3}$	$1.6e4_{1.8e3}$	$2.0e4_{2.4e3}$	+
	10	$2.4e4_{1.8e3}$	$2.6e4_{2.0e3}$	$4.2e3_{1.2e3}$	$1.1e4_{8.0e2}$	$6.8e3_{1.5e3}$	$1.7e4_{1.6e3}$	$2.4e4_{2.0e3}$	+
	20	$2.6e4_{1.6e3}$	$3.0e4_{2.0e3}$	$5.1e3_{2.3e3}$	$1.1e4_{8.0e2}$	$7.1e3_{1.8e3}$	$1.9e4_{1.6e3}$	$2.6e4_{1.8e3}$	+
	50	$3.1e4_{2.2e3}$	$3.4e4_{2.0e3}$	$1.1e4_{5.8e3}$	$1.2e4_{8.0e2}$	$8.4e3_{2.0e3}$	$2.2e4_{2.0e3}$	$3.1e4_{2.1e3}$	+
	100	—	—	—	$1.6e4_{1.6e3}$	$1.3e5_{1.5e5}$	—	—	+
ZDT2	1	$2.3e4_{1.4e3}$	$2.6e4_{2.2e3}$	$3.4e3_{1.0e3}$	$9.8e3_{7.5e2}$	$4.4e3_{2.2e3}$	$1.6e4_{2.7e3}$	$1.8e4_{6.8e3}$	+
	5	$2.5e4_{1.3e3}$	$2.8e4_{2.0e3}$	$3.8e3_{1.3e3}$	$1.0e4_{9.0e2}$	$4.9e3_{1.6e3}$	$1.7e4_{2.7e3}$	$2.0e4_{6.4e3}$	+
	10	$2.5e4_{1.4e3}$	$2.9e4_{1.9e3}$	$4.1e3_{1.5e3}$	$1.1e4_{9.0e2}$	$5.0e3_{1.8e3}$	$1.8e4_{2.6e3}$	$2.1e4_{5.9e3}$	+
	20	$2.7e4_{1.6e3}$	$3.0e4_{2.0e3}$	$4.5e3_{2.0e3}$	$1.1e4_{9.0e2}$	$5.1e3_{1.6e3}$	$1.9e4_{3.1e3}$	$2.2e4_{6.0e3}$	+
	50	$3.1e4_{1.6e3}$	$3.5e4_{2.1e3}$	$8.0e3_{5.4e3}$	$1.2e4_{8.5e2}$	$5.7e3_{2.4e3}$	$2.2e4_{3.0e3}$	$2.7e4_{6.6e3}$	+
	100	—	—	—	$1.6e4_{1.7e3}$	$9.5e4_{1.3e5}$	—	—	+
ZDT3	1	$1.2e4_{1.3e3}$	$1.4e4_{1.4e3}$	$2.8e3_{6.0e2}$	$9.4e3_{6.0e2}$	$8.3e3_{2.4e3}$	$1.1e4_{1.8e3}$	$1.2e4_{1.8e3}$	+
	5	$1.3e4_{1.2e3}$	$1.6e4_{1.2e3}$	$3.1e3_{8.5e2}$	$1.0e4_{6.0e2}$	$9.1e3_{2.4e3}$	$1.2e4_{1.5e3}$	$1.4e4_{1.5e3}$	+
	10	$1.5e4_{1.2e3}$	$1.7e4_{1.4e3}$	$3.5e3_{9.0e2}$	$1.0e4_{6.0e2}$	$9.5e3_{2.5e3}$	$1.3e4_{1.7e3}$	$1.5e4_{1.4e3}$	+
	20	$1.6e4_{1.3e3}$	$1.9e4_{1.5e3}$	$4.0e3_{1.0e3}$	$1.1e4_{7.0e2}$	$1.0e4_{2.8e3}$	$1.4e4_{1.7e3}$	$1.7e4_{1.4e3}$	+
	50	$2.1e4_{1.8e3}$	$2.5e4_{1.7e3}$	$5.6e3_{2.8e3}$	$1.2e4_{7.5e2}$	$1.2e4_{3.8e3}$	$1.6e4_{1.8e3}$	$2.2e4_{2.0e3}$	+
	100	—	—	—	—	–	—	—	–
ZDT4	1	$1.7e4_{4.0e3}$	$1.9e4_{4.0e3}$	$2.9e4_{1.7e4}$	—	$9.0e2_{7.0e2}$	$1.5e4_{7.0e3}$	$1.2e4_{6.0e3}$	+
	5	$3.3e4_{1.1e4}$	$3.2e4_{1.4e4}$	$3.9e4_{2.3e4}$	—	$2.6e3_{1.2e3}$	$4.5e4_{3.2e4}$	$3.1e4_{1.3e4}$	+
	10	$4.2e4_{1.8e4}$	$4.5e4_{1.9e4}$	$5.4e4_{2.6e4}$	—	$3.8e3_{1.2e3}$	$6.4e4_{3.8e4}$	$3.9e4_{1.6e4}$	+
	20	$5.7e4_{2.8e4}$	$5.5e4_{2.5e4}$	$6.5e4_{3.4e4}$	—	$7.7e3_{5.4e3}$	$8.2e4_{4.8e4}$	$4.8e4_{1.9e4}$	+
	50	$8.1e4_{4.8e4}$	$7.3e4_{4.0e4}$	$9.8e4_{4.8e4}$	—	$3.1e4_{1.3e4}$	$1.2e5_{6.6e4}$	$5.9e4_{2.0e4}$	–
	100	—	—	—	—	—	—	—	–
ZDT6	1	$2.1e4_{1.2e3}$	$2.4e4_{1.6e3}$	$2.2e3_{1.0e3}$	$2.8e3_{3.0e2}$	$1.5e3_{5.5e2}$	$1.1e4_{1.2e3}$	$1.4e4_{1.3e3}$	+
	5	$2.2e4_{1.3e3}$	$2.6e4_{1.4e3}$	$2.3e3_{9.5e2}$	$3.1e3_{3.0e2}$	$1.6e3_{5.0e2}$	$1.2e4_{1.3e3}$	$1.6e4_{1.4e3}$	+
	10	$2.4e4_{1.2e3}$	$2.8e4_{1.6e3}$	$2.3e3_{9.5e2}$	$3.3e3_{3.0e2}$	$1.8e3_{4.0e2}$	$1.3e4_{1.4e3}$	$1.7e4_{1.3e3}$	+
	20	$2.7e4_{1.6e3}$	$3.2e4_{1.6e3}$	$2.3e3_{1.0e3}$	$3.4e3_{3.0e2}$	$2.0e3_{4.0e2}$	$1.5e4_{1.3e3}$	$2.1e4_{1.8e3}$	+
	50	$3.6e4_{2.0e3}$	$4.3e4_{1.8e3}$	$2.7e3_{1.4e3}$	$3.5e3_{3.0e2}$	$2.3e3_{5.0e2}$	$1.9e4_{1.8e3}$	$3.0e4_{2.4e3}$	+
	100	—	—	—	—	—	—	—	–

second fastest ones in practically all problems. It is worth pointing out the results in ZDT4, a multi-frontal problem, which has been specially difficult for GDE3: the differential evolution algorithm has been unable to solve it in any of the experiments. If we look at which of the algorithms are able to compute 100 Pareto optimal solutions, only GDE3 and SMPSO have obtained such

DOI: 10.1002/nme

Table IV. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for reaching Pareto optimal points in the DTLZ benchmark (cells with dark and light backgrounds indicate the best and the second best values, respectively). PAES and SMPSO provide the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	n	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
DTLZ1	1	$2.4e4_{8.5e3}$	$2.4e4_{9.2e3}$	$2.5e4_{1.6e4}$	$9.3e3_{1.1e3}$	$7.0e2_{3.5e2}$	$2.8e4_{1.7e4}$	$1.9e4_{6.6e3}$	+
	5	$2.6e4_{8.8e3}$	$2.6e4_{1.1e4}$	$2.7e4_{1.8e4}$	$9.8e3_{9.0e2}$	$2.2e3_{1.4e3}$	$3.0e4_{1.9e4}$	$2.1e4_{7.4e3}$	+
	10	$2.6e4_{9.6e3}$	$2.8e4_{1.2e4}$	$2.8e4_{1.8e4}$	$9.9e3_{9.0e2}$	$3.2e3_{1.6e3}$	$3.1e4_{1.8e4}$	$2.2e4_{8.0e3}$	+
	20	$2.8e4_{1.1e4}$	$2.9e4_{1.2e4}$	$3.1e4_{2.0e4}$	$1.0e4_{1.0e3}$	$4.1e3_{1.5e3}$	$3.2e4_{2.1e4}$	$2.4e4_{9.0e3}$	+
	50	$3.4e4_{1.6e4}$	$3.5e4_{1.5e4}$	$4.4e4_{3.6e4}$	$1.1e4_{1.0e3}$	$6.9e3_{2.5e3}$	$4.3e4_{2.9e4}$	$2.9e4_{1.0e4}$	+
	100	—	—	—	—	—	—	—	−
DTLZ2	1	$3.5e3_{5.0e2}$	$4.0e3_{7.0e2}$	$7.0e2_{2.5e2}$	$4.5e3_{5.0e2}$	$6.0e2_{3.0e2}$	$2.2e3_{6.0e2}$	$2.4e3_{8.5e2}$	+
	5	$4.2e3_{5.0e2}$	$4.7e3_{6.5e2}$	$7.0e2_{3.0e2}$	$5.0e3_{4.0e2}$	$1.4e3_{6.0e2}$	$2.6e3_{6.0e2}$	$3.2e3_{9.0e2}$	+
	10	$4.7e3_{6.0e2}$	$5.2e3_{8.0e2}$	$8.0e2_{3.0e2}$	$5.3e3_{3.0e2}$	$1.9e3_{6.0e2}$	$2.8e3_{6.5e2}$	$3.7e3_{9.0e2}$	+
	20	$5.3e3_{5.0e2}$	$6.0e3_{9.0e2}$	$1.0e3_{7.0e2}$	$5.7e3_{4.0e2}$	$2.8e3_{1.1e3}$	$3.2e3_{9.0e2}$	$4.5e3_{9.0e2}$	+
	50	$7.7e3_{1.2e3}$	$8.5e3_{1.0e3}$	$3.2e3_{1.8e3}$	$6.8e3_{4.0e2}$	$6.4e3_{2.4e3}$	$4.9e3_{9.0e2}$	$7.0e3_{1.1e3}$	+
	100	—	—	—	—	—	—	—	−
DTLZ3	1	$6.0e4_{2.0e4}$	$6.8e4_{2.0e4}$	$8.6e4_{5.5e4}$	—	$1.0e3_{6.0e2}$	$7.7e4_{3.2e4}$	$4.8e4_{1.6e4}$	+
	5	$6.2e4_{2.2e4}$	$7.3e4_{2.2e4}$	$9.2e4_{6.0e4}$	—	$3.6e3_{3.3e3}$	$8.4e4_{3.8e4}$	$5.1e4_{1.6e4}$	+
	10	$6.7e4_{2.5e4}$	$7.7e4_{2.2e4}$	$1.1e5_{7.4e4}$	—	$5.0e3_{3.6e3}$	$8.6e4_{4.0e4}$	$5.5e4_{1.7e4}$	+
	20	$7.6e4_{2.9e4}$	$8.4e4_{2.8e4}$	$1.2e5_{1.0e5}$	—	$6.0e3_{3.8e3}$	$9.8e4_{4.9e4}$	$5.8e4_{1.6e4}$	+
	50	$1.0e5_{4.1e4}$	$1.1e5_{5.2e4}$	—	—	$8.4e3_{4.6e3}$	$1.4e5_{1.0e5}$	$7.8e4_{2.9e4}$	+
	100	—	—	—	—	—	—	—	−
DTLZ4	1	—	—	—	—	—	—	—	−
	5	—	—	—	—	—	—	—	−
	10	—	—	—	—	—	—	—	−
	20	—	—	—	—	—	—	—	−
	50	—	—	—	—	—	—	—	−
	100	—	—	—	—	—	—	—	−

DOI: 10.1002/nme

Table IV. *Continued.*

DTLZ5	1	$3.4e3_{5.5e2}$	$4.0e3_{6.0e2}$	$7.0e2_{3.0e2}$	$4.5e3_{3.0e2}$	$6.0e2_{3.0e2}$	$2.2e3_{6.0e2}$	$2.4e3_{8.5e2}$	+
	5	$4.1e3_{5.5e2}$	$4.7e3_{6.0e2}$	$7.0e2_{3.0e2}$	$5.0e3_{3.0e2}$	$1.4e3_{6.5e2}$	$2.6e3_{6.0e2}$	$3.2e3_{8.5e2}$	+
	10	$4.6e3_{5.0e2}$	$5.2e3_{8.0e2}$	$8.5e2_{4.0e2}$	$5.3e3_{3.5e2}$	$2.0e3_{8.0e2}$	$2.8e3_{6.5e2}$	$3.8e3_{8.5e2}$	+
	20	$5.2e3_{6.5e2}$	$5.9e3_{9.5e2}$	$1.1e3_{6.0e2}$	$5.7e3_{4.0e2}$	$3.1e3_{1.2e3}$	$3.2e3_{8.0e2}$	$4.6e3_{8.0e2}$	+
	50	$7.6e3_{1.2e3}$	$8.5e3_{1.1e3}$	$3.4e3_{1.8e3}$	$6.8e3_{4.0e2}$	$6.5e3_{2.6e3}$	$4.7e3_{1.2e3}$	$7.3e3_{1.0e3}$	+
	100	—	—	—	—	—	—	—	−
DTLZ6	1	—	—	$3.2e3_{1.2e3}$	$2.5e3_{2.0e2}$	$4.7e3_{2.2e3}$	—	—	+
	5	—	—	$3.2e3_{1.2e3}$	$2.7e3_{2.0e2}$	$5.0e3_{2.2e3}$	—	—	+
	10	—	—	$3.2e3_{1.2e3}$	$2.8e3_{2.0e2}$	$5.1e3_{2.2e3}$	—	—	+
	20	—	—	$3.3e3_{1.4e3}$	$2.9e3_{1.0e2}$	$5.3e3_{2.3e3}$	—	—	+
	50	—	—	$3.6e3_{1.6e3}$	$3.0e3_{2.0e2}$	$5.6e3_{2.4e3}$	—	—	+
	100	—	—	—	—	—	—	—	−
DTLZ7	1	$1.3e4_{1.2e3}$	$1.5e4_{1.3e3}$	$2.3e3_{6.0e2}$	$7.1e3_{5.0e2}$	$3.5e3_{1.6e3}$	$9.4e3_{1.5e3}$	$1.1e4_{1.0e3}$	+
	5	$1.4e4_{8.0e2}$	$1.6e4_{1.0e3}$	$2.6e3_{1.0e3}$	$7.4e3_{5.5e2}$	$3.7e3_{1.4e3}$	$1.0e4_{1.2e3}$	$1.2e4_{1.0e3}$	+
	10	$1.5e4_{1.0e3}$	$1.7e4_{1.1e3}$	$2.8e3_{1.0e3}$	$7.7e3_{6.0e2}$	$3.9e3_{1.6e3}$	$1.1e4_{1.4e3}$	$1.3e4_{1.1e3}$	+
	20	$1.6e4_{1.0e3}$	$1.9e4_{1.2e3}$	$3.1e3_{1.0e3}$	$7.9e3_{5.5e2}$	$4.1e3_{1.6e3}$	$1.1e4_{1.4e3}$	$1.4e4_{1.1e3}$	+
	50	$2.0e4_{1.6e3}$	$2.3e4_{1.4e3}$	$4.2e3_{1.8e3}$	$8.6e3_{7.5e2}$	$4.6e3_{1.8e3}$	$1.4e4_{1.4e3}$	$1.8e4_{1.8e3}$	+
	100	—	—	—	—	—	—	—	−

DOI: 10.1002/nme

Table V. Problems in which statistical confidence cannot be assured in Criterion 1 (number of Pareto optimal solutions found).

	n	GDE3	SMPSO
PAES	1	—	DTLZ2, DTLZ5
	5	—	—
	10	—	—
	20	—	ZDT2
	50	—	DTLZ7
	100	—	ZDT3, ZDT4
GDE3	1		—
	5		—
	10		—
	20		ZDT3
	50		ZDT3, DTLZ2, DTLZ5
	100		ZDT6, DTLZ6

a number in less than 1 000 000 evaluations in problems ZDT1 and ZDT2. Statistical confidence can be ensured in practically all cases, as indicated in the last column.

As to the number of evaluations needed for computing the Pareto optimal solutions for the DTLZ family (Table IV), the algorithms have behaved in a similar way than in the ZDT benchmark. PAES and SMPSO have been the fastest algorithms in the comparison. GDE3 has also shown a good performance: it has been the fastest algorithm in meeting the convergence criteria in the DTLZ6 problem, and the second fastest in the problem DTLZ1. Within this set of problems, GDE3 has been unable to obtain any Pareto optimal solution in DTLZ3. It is also remarkable that none of the algorithms has obtained any Pareto optimal solutions for the DTLZ4 problem, and all the algorithms but PAES, GDE3, and SMPSO have failed when solving problem DTLZ6. As in ZDT, statistical confidence has been found among the algorithms in practically all cases from the DTLZ benchmark.

Thus, according to the obtained results, the three fastest algorithms in reaching Pareto optimal solutions in the chosen benchmark problems have been PAES, SMPSO, and GDE3. Table V includes the problems in which statistical differences between these algorithms cannot be ensured. This table indicates that there are statistical differences between PAES and GDE3 in all the problems; focusing on PAES and SMPSO, statistical differences cannot be assured in two cases, DTLZ2 and DTLZ5, when they obtain only one Pareto optimal solution. If the criterion is to obtain a higher number of solutions, these two algorithms present confidence in practically all problems. As to the comparison between GDE3 and SMPSO, the most remarkable fact is that, when both algorithms compute 50 Pareto optimal solutions, statistical confidence cannot be assured in 3 out of the 12 problems evaluated: ZDT3, DTLZ2, and DTLZ5.

A conclusion of the carried out experiments is that finding 100 Pareto optimal solutions seems to be very hard to the algorithms. Only in 2 out of the 12 problems, two techniques have succeeded at this. Furthermore, some interesting facts can be drawn from these tables:

1. In problems ZDT1, ZDT6, and DTLZ7, PAES has generated a Pareto front containing 20 (10 in the case of ZDT2) or more Pareto optimal solutions in fewer evaluations than those required by the second fastest algorithm to find a single Pareto optimal solution.
2. In the multi-frontal problems ZDT4, DTLZ1, and DTLZ3, SMPSO has required a lower number of function evaluations to produce a Pareto front with 50 Pareto optimal solutions

DOI: 10.1002/nme

Table VI. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for different values of the I_ε^1+ indicator in the ZDT benchmark (cells with dark and light backgrounds indicate the best and the second best values, respectively). SMPSO provides the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	I_ε^1+	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
ZDT1	0.10	$6.1e3_{6.0e2}$	$7.6e3_{1.1e3}$	$8.2e3_{6.5e3}$	$6.2e3_{3.0e2}$	$5.8e3_{1.2e3}$	$8.0e3_{1.3e3}$	$5.5e3_{1.2e3}$	+
	0.05	$7.8e3_{6.0e2}$	$9.5e3_{9.5e2}$	$9.9e3_{8.7e3}$	$7.5e3_{3.5e2}$	$6.4e3_{1.4e3}$	$9.5e3_{1.6e3}$	$6.9e3_{1.1e3}$	+
	0.01	$2.9e4_{8.2e3}$	$1.9e4_{1.4e3}$	—	$1.1e4_{7.0e2}$	$7.9e3_{1.7e3}$	$1.9e4_{5.0e3}$	$1.5e4_{1.6e3}$	+
ZDT2	0.10	$8.9e3_{9.0e2}$	$1.2e4_{8.8e3}$	$8.9e3_{5.6e3}$	$7.4e3_{5.5e2}$	$4.8e3_{1.7e3}$	$9.2e3_{3.2e3}$	$6.5e3_{2.5e3}$	+
	0.05	$1.1e4_{9.5e2}$	$1.5e4_{7.2e3}$	$1.0e4_{8.8e3}$	$8.3e3_{3.5e2}$	$5.1e3_{1.6e3}$	$1.0e4_{3.0e3}$	$7.7e3_{1.7e3}$	+
	0.01	$2.8e4_{5.4e3}$	$2.2e4_{2.4e3}$	—	$1.2e4_{8.5e2}$	$6.2e3_{1.8e3}$	$1.8e4_{4.6e3}$	$1.2e4_{4.0e3}$	+
ZDT3	0.10	$6.3e3_{7.5e2}$	$7.7e3_{9.5e2}$	$3.7e4_{9.3e4}$	$6.7e3_{3.0e2}$	$8.6e3_{2.5e3}$	$1.3e4_{2.6e4}$	$7.2e3_{3.0e3}$	+
	0.05	$8.0e3_{8.0e2}$	$9.6e3_{1.2e3}$	$4.0e4_{9.3e4}$	$8.0e3_{6.0e2}$	$9.5e3_{2.4e3}$	$1.3e4_{2.6e4}$	$8.7e3_{2.6e3}$	+
	0.01	$1.4e4_{1.4e3}$	$1.8e4_{2.2e3}$	—	$1.2e4_{1.6e3}$	$1.2e4_{2.7e3}$	$1.5e4_{2.1e4}$	$1.4e4_{2.2e3}$	+
ZDT4	0.10	$1.6e4_{3.8e3}$	$2.2e4_{5.7e3}$	$1.6e4_{1.1e4}$	$1.4e4_{1.0e3}$	$3.2e3_{1.2e3}$	$1.5e4_{4.2e3}$	$1.3e4_{3.2e3}$	+
	0.05	$1.8e4_{4.0e3}$	$2.4e4_{7.5e3}$	$2.3e4_{1.1e4}$	$1.4e4_{1.0e3}$	$4.0e3_{1.4e3}$	$1.6e4_{4.3e3}$	$1.4e4_{3.2e3}$	+
	0.01	$5.0e4_{1.6e4}$	$3.1e4_{7.4e3}$	—	—	$8.7e3_{2.2e3}$	$3.0e4_{1.9e4}$	$2.1e4_{7.1e3}$	+
ZDT6	0.10	$1.5e4_{1.1e3}$	$1.8e4_{1.0e3}$	$4.4e3_{3.9e3}$	$3.7e3_{4.0e2}$	$2.7e3_{1.4e3}$	$8.1e3_{1.1e3}$	$7.9e3_{9.0e2}$	+
	0.05	$1.8e4_{9.0e2}$	$2.1e4_{1.3e3}$	$6.3e3_{5.4e3}$	$4.2e3_{6.0e2}$	$3.3e3_{1.4e3}$	$9.8e3_{1.2e3}$	$1.1e4_{1.0e3}$	+
	0.01	$3.8e4_{3.5e3}$	$3.5e4_{2.0e3}$	—	$6.9e3_{2.0e3}$	$4.7e3_{1.7e3}$	$1.7e4_{1.4e3}$	$2.2e4_{1.3e3}$	+

than the number of function evaluations required by the rest of the algorithms to find only one Pareto optimal solution.

3. SMPSO, the particle swarm optimization algorithm, is always in the group of the three fastest techniques.

5.2. *Epsilon indicator*

The number of function evaluations at which the algorithms compute an approximation with the desired values of I_ε^1+ in the ZDT family are included in Table VI. Attending to the values shown in this table, the fastest algorithm has been SMPSO, the particle swarm optimizer, because it required the lowest number of function evaluations for meeting the termination criteria in four out of the five problems composing this benchmark. GDE3 has been the second fastest algorithm in the ZDT3 problem, and the second fastest in ZDT6; it has also been the second algorithm requiring the lowest number of evaluations for computing an approximation to the Pareto front with an I_ε^1+ value of 0.01 in problems ZDT1 and ZDT2. It is also worth highlighting MOCell and NSGA-II: the former algorithm has been the fastest algorithm if we consider $I_\varepsilon^1+ = 0.1$ in problem ZDT1, and the second fastest algorithm in several cases in problems ZDT1, ZDT2, and ZDT4; the latter has been the algorithm requiring the lowest number of evaluations for computing an approximation front with an I_ε^1+ value of 0.1 and 0.05 in problem ZDT3.

DOI: 10.1002/nme

Table VII. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for different values of the I_ε^1+ indicator in the DTLZ benchmark (cells with dark and light backgrounds indicate the best and second best values, respectively). SMPSO provides the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	I_ε^1+	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
DTLZ1	0.10	$1.5e4_{3.9e3}$	$1.8e4_{4.5e3}$	—	$8.4e3_{8.0e2}$	$3.6e3_{1.8e3}$	$1.7e4_{8.6e3}$	$1.5e4_{4.6e3}$	+
	0.05	$1.6e4_{3.6e3}$	$1.9e4_{4.0e3}$	—	$8.9e3_{7.5e2}$	$4.3e3_{2.0e3}$	$1.7e4_{8.4e3}$	$1.6e4_{4.8e3}$	+
	0.01	$4.3e4_{2.5e4}$	$2.7e4_{7.1e3}$	—	$1.1e4_{1.2e3}$	$8.1e3_{2.0e3}$	$2.8e4_{1.3e4}$	$2.2e4_{5.4e3}$	+
DTLZ2	0.10	$1.5e3_{1.0e2}$	$1.8e3_{3.0e2}$	$2.8e3_{2.3e3}$	$2.1e3_{1.0e2}$	$6.0e2_{3.0e2}$	$1.4e3_{6.0e2}$	$9.0e2_{3.0e2}$	+
	0.05	$2.1e3_{2.0e2}$	$2.5e3_{3.0e2}$	$3.7e3_{2.7e3}$	$2.9e3_{3.0e2}$	$1.1e3_{3.0e2}$	$2.1e3_{6.0e2}$	$1.4e3_{4.0e2}$	+
	0.01	$6.4e3_{1.6e3}$	$5.2e3_{7.0e2}$	—	$5.6e3_{4.0e2}$	$3.7e3_{1.1e3}$	$4.2e3_{1.0e3}$	$3.5e3_{6.0e2}$	+
DTLZ3	0.10	$3.1e4_{8.2e3}$	$3.7e4_{8.5e3}$	—	—	$5.1e3_{3.4e3}$	$4.0e4_{1.3e4}$	$3.0e4_{9.0e3}$	+
	0.05	$3.4e4_{8.6e3}$	$3.9e4_{7.6e3}$	—	—	$5.9e3_{3.2e3}$	$4.1e4_{1.2e4}$	$3.1e4_{9.0e3}$	+
	0.01	$7.0e4_{2.7e4}$	$6.1e4_{1.4e4}$	—	—	$8.6e3_{3.5e3}$	$6.8e4_{3.0e4}$	$4.3e4_{1.0e4}$	+
DTLZ4	0.10	—	—	—	$4.7e3_{1.8e3}$	$1.4e3_{5.0e2}$	$1.5e3_{6.0e2}$	—	+
	0.05	—	—	—	$6.4e3_{2.1e3}$	$1.9e3_{5.0e2}$	$2.0e3_{5.0e2}$	—	+
	0.01	—	—	—	$8.8e3_{1.8e3}$	$4.2e3_{1.1e3}$	$4.8e3_{1.1e3}$	—	+
DTLZ5	0.10	$1.5e3_{1.0e2}$	$1.8e3_{2.5e2}$	$3.2e3_{3.4e3}$	$2.1e3_{2.0e2}$	$7.0e2_{3.0e2}$	$1.5e3_{6.5e2}$	$9.0e2_{2.0e2}$	+
	0.05	$2.1e3_{2.0e2}$	$2.6e3_{4.5e2}$	$3.6e3_{3.8e3}$	$2.9e3_{2.0e2}$	$1.1e3_{3.0e2}$	$2.1e3_{6.5e2}$	$1.4e3_{3.0e2}$	+
	0.01	$6.2e3_{1.7e3}$	$5.3e3_{6.0e2}$	–	$5.6e3_{3.0e2}$	$3.6e3_{8.0e2}$	$4.2e3_{1.0e3}$	$3.6e3_{6.0e2}$	+
DTLZ6	0.10	$2.4e4_{8.5e2}$	$2.8e4_{1.1e3}$	$1.2e4_{1.2e4}$	$2.9e3_{2.0e2}$	$5.1e3_{1.9e3}$	—	—	+
	0.05	—	—	$1.5e4_{1.5e4}$	$3.1e3_{2.0e2}$	$5.3e3_{2.0e3}$	—	—	+
	0.01	—	—	—	$4.3e3_{4.0e2}$	$6.3e3_{2.2e3}$	—	—	+
DTLZ7	0.10	$6.0e3_{5.0e2}$	—	—	$5.8e3_{3.5e2}$	$3.9e3_{1.3e3}$	—	—	+
	0.05	$7.2e3_{7.0e2}$	—	—	$6.7e3_{4.5e2}$	$4.3e3_{1.2e3}$	—	—	+
	0.01	$1.3e4_{1.3e3}$	—	—	$9.8e3_{1.0e3}$	$5.4e3_{1.4e3}$	—	—	+

Table VII contains the evaluations required in problems belonging to the DTLZ family. The fastest technique at meeting these criteria has been SMPSO: it has been the algorithm needing the lowest number of evaluations in six out of the seven problems adopted in this comparison; additionally, it had the second lowest number of evaluations in the other problems. After SMPSO, GDE3, MOCell, and AbYSS have obtained outstanding results in different problems:

- GDE3 has shown to be fast for solving problems DTLZ1 (second best values), DTLZ6 (best values), and DTLZ7 (second fastest algorithm).
- MOCell has been the second algorithm requiring the lowest number of evaluations in problems DTLZ2, DTLZ3, and DTLZ5.
- AbYSS has been the second fastest in meeting the convergence criteria in the problem DTLZ4.

DOI: 10.1002/nme

Table VIII. Problems in which statistical confidence cannot be assured in Criterion 2 (I_ε^1+ indicator values).

	n	SMPSO	MOCell
GDE3	0.10	—	DTLZ4
	0.05	—	DTLZ4
	0.01	ZDT3	ZDT2, DTLZ4, DTLZ7
SMPSO	0.1		—
	0.05		ZDT3
	0.01		DTLZ2, DTLZ5

In summary, the three fastest algorithms in meeting this convergence criterion have been SMPSO, GDE3, and MOCell. Table VIII summarizes the pairwise comparison between these algorithms, showing those problems in which statistical confidence cannot be ensured. According to this table, the differences between GDE3 and SMPSO are significant in most problems; as to MOCell versus the other two algorithms, statistical confidence can also be assured in most cases for I_ε^1+ values of 0.1 and 0.05. However, with the smallest I_ε^1+ value (0.01), the number of problems in which confidence cannot be confirmed is higher: ZDT2, DTLZ4, and DTLZ7 in the comparison against GDE3, and DTLZ2 and DTLZ5 against SMPSO.

5.3. Hypervolume indicator

Table IX shows the number of evaluations required by all the algorithms for reaching an accurate approximation to the true Pareto front making use of the I_{HV} indicator. In this table, we see that SMPSO requires the lowest number of function evaluations, i.e. it is the fastest in meeting this convergence criterion. GDE3 has obtained the second best values in most of the problems but ZDT4; in this problem, MOCell has been the second fastest algorithm.

We include in Table X the number of evaluations at which the algorithms compute a Pareto front with the desired values of I_{HV} in the DTLZ family. As happened in the ZDT family, SMPSO has obtained good figures (i.e. lowest or second lowest number of evaluations) in practically all the problems. GDE3 achieved the best results in four out of the seven DTLZ problems when the stopping condition is 99% the HV of the true Pareto front, whereas AbYSS needed the lowest number of evaluations in three problems when 98% is considered.

Summarizing the results, the three algorithms requiring the lowest number of evaluations to compute an approximate front with those values of I_{HV} have been SMPSO, GDE3, and AbYSS. Table XI shows the results of the pairwise comparison between these algorithms. As we can see in the table, the differences are statistically significant in practically all the cases.

6. DISCUSSION OF THE RESULTS

In this section, we provide a ranking of the algorithms according to the three criteria previously described with the idea of analyzing the results globally. The main aim is to identify the strengths and the weaknesses of the algorithms when converging toward the true Pareto front of a problem. We also evaluate the effect of considering all the solutions generated during the execution of an

DOI: 10.1002/nme

Table IX. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for different values of the I_{HV} indicator in the ZDT benchmark (cells with dark and light backgrounds indicate the best and the second best values, respectively). SMPSO provides the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	%	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	
ZDT1	98	$1.4e4_{8.5e2}$	$1.6e4_{1.1e3}$	$1.2e4_{9.6e3}$	$9.5e3_{4.0e2}$	$7.0e3_{1.6e3}$	$1.3e4_{1.5e3}$	$1.3e4_{1.3e3}$	+
	99	$2.4e4_{1.4e3}$	$2.3e4_{1.2e3}$	—	$1.1e4_{5.5e2}$	$7.6e3_{1.6e3}$	$1.8e4_{1.8e3}$	$2.0e4_{1.3e3}$	−
ZDT2	98	$2.4e4_{1.6e3}$	$2.5e4_{1.6e3}$	$1.6e5_{2.4e5}$	$1.1e4_{6.5e2}$	$5.7e3_{1.6e3}$	$1.7e4_{2.6e3}$	$1.7e4_{6.6e3}$	+
	99	—	—	—	—	—	—	—	−
ZDT3	98	$1.3e4_{1.2e3}$	$1.5e4_{1.3e3}$	$2.4e4_{2.9e4}$	$1.0e4_{5.0e2}$	$1.0e4_{2.6e3}$	$1.2e4_{2.4e3}$	$1.3e4_{1.4e3}$	+
	99	$1.6e4_{1.2e3}$	$2.0e4_{1.4e3}$	$5.1e4_{9.7e4}$	$1.1e4_{6.0e2}$	$1.1e4_{2.8e3}$	$1.5e4_{4.3e3}$	$1.7e4_{1.8e3}$	+
ZDT4	98	$2.2e4_{5.6e3}$	$2.6e4_{5.7e3}$	$4.3e4_{2.0e4}$	—	$4.9e3_{1.3e3}$	$2.1e4_{1.0e4}$	$1.7e4_{6.2e3}$	+
	99	$4.1e4_{1.7e4}$	$3.8e4_{1.6e4}$	—	—	$8.8e3_{2.0e3}$	$5.1e4_{2.7e4}$	$3.1e4_{9.8e3}$	−
ZDT6	98	$2.9e4_{1.2e3}$	$3.3e4_{1.2e3}$	$9.7e3_{7.4e3}$	$4.6e3_{6.0e2}$	$3.4e3_{1.5e3}$	$1.5e4_{1.4e3}$	$2.2e4_{1.4e3}$	+
	99	$3.6e4_{1.7e3}$	$4.0e4_{1.5e3}$	—	$5.2e3_{6.5e2}$	$3.8e3_{1.5e3}$	$1.8e4_{1.4e3}$	$2.7e4_{1.6e3}$	−

Table X. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for reaching different values of the I_{HV} quality indicator in the DTLZ benchmark (cells with dark and light backgrounds indicate the best and the second best values, respectively). SMPSO and GDE3 provide the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	%	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	
DTLZ1	98	$2.5e4_{8.0e3}$	$2.5e4_{8.0e3}$	—	$1.0e4_{9.5e2}$	$5.5e3_{2.0e3}$	$2.7e4_{1.4e4}$	$2.1e4_{6.2e3}$	+
	99	$4.9e4_{2.6e4}$	$3.6e4_{1.6e4}$	—	$1.1e4_{1.0e3}$	$8.3e3_{2.1e3}$	$4.4e4_{3.0e4}$	$3.0e4_{1.1e4}$	+
DTLZ2	98	$7.9e3_{1.2e3}$	$7.4e3_{8.0e2}$	$2.9e4_{2.4e4}$	$6.4e3_{3.5e2}$	$5.5e3_{1.4e3}$	$4.6e3_{8.5e2}$	$5.9e3_{1.0e3}$	+
	99	—	—	—	$9.6e3_{6.0e2}$	$3.6e4_{1.1e4}$	$1.3e4_{3.6e3}$	$1.7e4_{1.6e3}$	+
DTLZ3	98	$1.1e5_{5.4e4}$	$1.0e5_{4.3e4}$	—	—	$8.8e3_{4.2e3}$	$1.1e5_{6.8e4}$	$6.6e4_{2.3e4}$	+
	99	—	$3.4e5_{1.7e5}$	—	—	$3.5e4_{1.5e4}$	$3.5e5_{2.5e5}$	$1.4e5_{4.6e4}$	+
DTLZ4	98	—	—	—	$8.0e3_{1.2e3}$	$5.9e3_{1.4e3}$	$4.8e3_{6.0e2}$	—	−
	99	—	—	—	$1.0e4_{8.5e2}$	$4.1e4_{1.1e4}$	$1.3e4_{3.0e3}$	—	−
DTLZ5	98	$8.2e3_{1.2e3}$	$7.4e3_{7.5e2}$	$2.7e4_{2.8e4}$	$6.4e3_{4.0e2}$	$5.6e3_{1.5e3}$	$4.5e3_{8.0e2}$	$5.9e3_{1.0e3}$	+
	99	—	—	—	$9.6e3_{5.5e2}$	$3.5e4_{1.1e4}$	$1.3e4_{2.7e3}$	$1.7e4_{2.2e3}$	−
DTLZ6	98	—	—	—	$3.8e3_{3.0e2}$	$5.9e3_{2.3e3}$	—	—	+
	99	—	—	—	$5.0e3_{5.5e2}$	$7.5e3_{4.1e3}$	—	—	−
DTLZ7	98	$1.4e4_{8.0e2}$	—	—	$8.4e3_{5.0e2}$	$4.8e3_{1.4e3}$	—	—	−
	99	$2.0e4_{1.6e3}$	—	—	$9.4e3_{5.0e2}$	$5.4e3_{1.4e3}$	—	—	−

DOI: 10.1002/nme

Table XI. Problems in which statistical confidence cannot be assured in Criterion 3 (I_{HV} indicator percentages).

	n (%)	SMPSO	AbYSS
GDE3	98	ZDT3	—
	99	ZDT3	—
SMPSO	98		—
	99		—

Table XII. Ranking of the algorithms: obtaining a pre-defined number (1, 5, 10, 20, 50, and 100) of the Pareto optimal solutions.

Rank	1 sol.	5 sols.	10 sols.	20 sols.	50 sols.	100 sols.	Global
1.	SMPSO	PAES	PAES	PAES	SMPSO	GDE3	PAES
2.	PAES	SMPSO	SMPSO	SMPSO	PAES	SMPSO	SMPSO
3.	GDE3	GDE3	GDE3	GDE3	GDE3	—	GDE3
4.	MOCell	MOCell	MOCell	MOCell	MOCell	—	MOCell
5.	AbYSS	AbYSS	AbYSS	AbYSS	AbYSS	—	AbYSS
6.	NSGA-II	NSGA-II	NSGA-II	NSGA-II	NSGA-II	—	NSGA-II
7.	SPEA2	SPEA2	SPEA2	SPEA2	SPEA2	—	SPEA2

algorithm for measuring the convergence speed. Finally, we compare the results of this study with respect to our own previous work [18].

6.1. Ranking of the algorithms

The rankings of the algorithms using the three convergence speed criteria are presented in Tables XII, XIII, and XIV, respectively. These rankings consider first those algorithms that required the lowest or second lowest number of function evaluations in most cases. In case of algorithms having the same number of best or second best results, we pay attention to the number of evaluations performed by them so that we can establish an ordering. We have included in the last column of each table a global ranking to make clearer the identification of the most efficient techniques. Table XII shows the ranking taking into account the number of evaluations that they need for computing a front containing a given number of Pareto optimal solutions. In this ranking, PAES, SMPSO, and GDE3 are the most outstanding metaheuristics: PAES achieves 3 first and 2 second best positions in the ranking; SMPSO gets 2 best and 4 second best positions; and GDE has a best and 5 third positions. MOCell and AbYSS are invariably in the fourth and fifth positions in all cases, respectively. The two GAs, NSGA-II and SPEA2, are the worst techniques according to this ranking. The most remarkable conclusion according to this convergence criterion is that PAES, the simplest of the compared algorithms, is globally the fastest metaheuristic!

Table XIII includes the ranking of the algorithms considering the speed of the techniques when computing a Pareto front with the target values of the I_ε^1+ indicator. In this case, the three fastest algorithms are SMPSO, GDE3, and MOCell. However, it is worth mentioning that the most efficient technique in the first criterion, PAES, is the worst in this new ranking. This suggests that PAES is very fast obtaining some Pareto optimal solutions, but it has problems for computing an entire set of solutions closer to the true Pareto front (let us remind that the I_ε^1+ indicator is the smallest

DOI: 10.1002/nme

Table XIII. Ranking of the algorithms: computing Pareto fronts with different values (0.1, 0.05, and 0.01) of the Epsilon indicator.

Rank	0.1	0.05	0.01	Global
1.	SMPSO	SMPSO	SMPSO	SMPSO
2.	MOCell	GDE3	GDE3	GDE3
3.	GDE3	MOCell	MOCell	MOCell
4.	NSGA-II	NSGA-II	AbYSS	NSGA-II
5.	AbYSS	AbYSS	NSGA-II	AbYSS
6.	SPEA2	SPEA2	SPEA2	SPEA2
7.	PAES	PAES	PAES	PAES

Table XIV. Ranking of the algorithms: computing Pareto fronts with different percentages (0.98 and 0.99) of the Hypervolume indicator.

Rank	0.98	0.99	Global
1.	SMPSO	SMPSO	SMPSO
2.	AbYSS	GDE3	GDE3
3.	GDE3	AbYSS	AbYSS
4.	MOCell	MOCell	MOCell
5.	NSGA-II	NSGA-II	NSGA-II
6.	SPEA2	SPEA2	SPEA2
7.	PAES	PAES	PAES

distance to translate all the points in the found approximation set to dominate the Pareto front of the problem). We also observe that NSGA-II has obtained a better position than AbYSS in this ranking, being SPEA2 the second to last in the global ranking.

Table XIV contains the ranking according to the strongest of the three defined criteria, taking into account both convergence and diversity, which is led again by SMPSO. GDE3 and AbYSS appear as the second and third fastest algorithms, respectively, being MOCell in the middle of the table. This ranking confirms the poor results of the GAs NSGA-II and SPEA2 (fifth and sixth), and PAES.

Summarizing this section, according to the considered benchmark, efficiency criteria, and parameters settings used in our study, we can conclude that SMPSO is the most efficient out of the seven compared metaheuristics. It is the second best performing algorithm in the first criteria, and the best one in the other two. PAES is the algorithm that takes into account if we are interested in obtaining some Pareto optimal solutions quickly. It has the advantage of being the simplest of the compared metaheuristics, requiring to set only one parameter (the size of the adaptive grid). However, PAES should be the last choice if we require to get an approximation set with a certain level of quality. GDE3 follows SMPSO in the three criteria; hence, it is also a technique to take into consideration. MOCell and AbYSS are modern algorithms whose performance in the tests led them to be in the middle of the rankings. Finally, NSGA-II and SPEA2, which are the two most well-known and popular multi-objective algorithms in current use, should be out of our preferences if efficiency is our major concern.

DOI: 10.1002/nme

6.2. Evaluating the effects of the Pareto drift issue

Previous works [37–40] in the field of multi-objective optimization have reported that some multi-objective evolutionary algorithms may lose good solutions (including Pareto optimal ones) previously generated as a consequence of using a finite size population or a fixed size external archive. Specifically, those algorithms make use of a diversity preserving mechanism (such as the crowding distances of NSGA-II or the hypercubes of PAES) that may discard optimal solutions for the sake of non-optimal ones (still non-dominated with respect to the population or archive), but better in terms of the density estimator adopted. This is known as the *Pareto drift* problem [40].

Pareto drift has a direct impact on the first convergence criterion used in this work since none of the studied algorithms are free from this issue. Nevertheless, as commented in [40], this problem can be simply solved by using an external unbounded archive where all the non-dominated solutions generated during the search are stored. Thus, in this section, we analyze the behavior of NSGA-II, SPEA2, PAES, GDE3, SMPSO, AbYSS, and MOCell with respect to the first convergence criterion when using an unbounded external archive. It is worth remarking that the use of such an archive does not modify the behavior of the algorithms.

Tables XV and XVI include the number of required evaluations by the seven algorithms included in this work to meet the convergence conditions of Criterion 1 using the ZDT and DTLZ test suites, respectively.

Deepening on the analysis of the tables, we observe that, when no solutions are lost, the algorithms converge faster than their original versions and, furthermore, all of them are able to compute up to 100 optimal solutions (the strongest condition in this criterion) in practically all the problems evaluated. The consequences of avoiding the Pareto drift issues become more appreciable if we compare their effects over the same problem. For example, taking into account the NSGA-II algorithm and the ZDT1 problem, we observe that the version without Pareto drift (Table XV) is able to compute 100 optimal solutions in a lower number of evaluations than the number required by the original version (Table XV) to generate up to 20 Pareto optimal solutions.

Comparing those tables with those that include the number of evaluations performed by the original algorithms (Tables III and IV for the ZDT and DTLZ test suites, respectively), it is possible to see at a glance that the behavior of the algorithms is pretty much the same: PAES and SMPSO have been the algorithms obtaining the overall best results in the two test suites.

6.3. Comparison with previous work

In this section, we compare the results of this paper with those obtained in our previous work [18]. In that paper, the main conclusions were that, considering the speed to converge toward the true Pareto front of a problem, using the hypervolume as our indicator for measuring convergence, MOCell, OMOPSO, and AbYSS were the most competitive algorithms followed by NSGA-II and PAES, whereas SPEA2 appeared as the slowest algorithm of the comparison. The fact that PAES appeared as one of the slowest algorithms in that comparison is explained by the criterion used in that work: it measured both convergence and diversity of solutions. Here, we have also considered another two criteria to measure the convergence speed: to compute a Pareto front containing a certain pre-defined number of optimal solutions, and to compute a Pareto front with a fixed value of the I_ε^1+ indicator. The former criterion has allowed us to observe that PAES is one of the fastest algorithms in obtaining Pareto optimal solutions, but at the expense of not taking into account diversity.

DOI: 10.1002/nme

Table XV. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for reaching Pareto optimal points in the ZTD benchmark when using an external unbounded archive for avoiding Pareto drift issues (cells with dark and light backgrounds indicate the best and the second best values, respectively). PAES and SMPSO also provide the best overall results in this case.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	n	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
ZDT1	1	$1.1e4_{1.2e3}$	$1.3e4_{1.6e3}$	$3.4e3_{1.3e3}$	$9.7e3_{9.5e2}$	$6.1e3_{1.8e3}$	$1.1e4_{1.8e3}$	$1.0e4_{1.5e3}$	+
	5	$1.5e4_{1.4e3}$	$1.7e4_{1.8e3}$	$4.2e3_{1.5e3}$	$1.0e4_{8.0e2}$	$6.3e3_{1.8e3}$	$1.3e4_{2.0e3}$	$1.3e4_{1.8e3}$	+
	10	$1.7e4_{1.5e3}$	$1.9e4_{1.4e3}$	$4.7e3_{1.8e3}$	$1.1e4_{7.0e2}$	$6.5e3_{1.8e3}$	$1.4e4_{2.1e3}$	$1.6e4_{1.6e3}$	+
	20	$1.9e4_{1.6e3}$	$2.2e4_{1.3e3}$	$5.7e3_{2.3e3}$	$1.1e4_{7.0e2}$	$6.7e3_{1.8e3}$	$1.5e4_{1.9e3}$	$1.8e4_{1.6e3}$	+
	50	$2.2e4_{1.3e3}$	$2.5e4_{1.2e3}$	$9.5e3_{3.0e3}$	$1.2e4_{8.0e2}$	$7.1e3_{1.8e3}$	$1.7e4_{1.8e3}$	$2.2e4_{2.0e3}$	+
	100	$2.5e4_{1.4e3}$	$2.8e4_{1.3e3}$	$1.4e4_{9.9e5}$	$1.2e4_{8.0e2}$	$7.6e3_{2.0e3}$	$1.8e4_{1.7e3}$	$2.4e4_{1.9e3}$	+
ZDT2	1	$2.3e4_{1.8e3}$	$2.6e4_{2.1e3}$	$3.6e3_{1.1e3}$	$9.8e3_{9.0e2}$	$4.4e3_{2.0e3}$	$1.6e4_{2.6e3}$	$1.6e4_{6.8e3}$	+
	5	$2.4e4_{1.4e3}$	$2.7e4_{2.0e3}$	$4.2e3_{1.5e3}$	$1.0e4_{9.0e2}$	$4.7e3_{1.5e3}$	$1.6e4_{2.6e3}$	$1.7e4_{6.6e3}$	+
	10	$2.4e4_{1.4e3}$	$2.8e4_{2.0e3}$	$4.6e3_{1.8e3}$	$1.1e4_{9.5e2}$	$4.8e3_{1.6e3}$	$1.6e4_{2.5e3}$	$1.8e4_{6.6e3}$	+
	20	$2.5e4_{1.5e3}$	$2.8e4_{2.0e3}$	$5.9e3_{2.2e3}$	$1.1e4_{8.5e2}$	$5.1e3_{1.6e3}$	$1.7e4_{2.6e3}$	$1.8e4_{6.7e3}$	+
	50	$2.6e4_{1.4e3}$	$2.9e4_{1.9e3}$	$9.3e3_{2.6e3}$	$1.2e4_{8.5e2}$	$5.4e3_{1.8e3}$	$1.7e4_{2.6e3}$	$1.9e4_{6.9e3}$	+
	100	$2.7e4_{1.6e3}$	$3.0e4_{2.0e3}$	$1.4e4_{9.9e5}$	$1.2e4_{8.0e2}$	$5.7e3_{1.9e3}$	$1.8e4_{2.6e3}$	$2.0e4_{6.8e3}$	+
ZDT3	1	$1.2e4_{1.0e3}$	$1.4e4_{1.2e3}$	$2.9e3_{7.0e2}$	$9.4e3_{6.0e2}$	$8.4e3_{2.8e3}$	$1.1e4_{1.4e3}$	$1.2e4_{1.2e3}$	+
	5	$1.3e4_{1.0e3}$	$1.5e4_{1.4e3}$	$3.4e3_{1.0e3}$	$1.0e4_{5.5e2}$	$9.2e3_{2.8e3}$	$1.2e4_{1.5e3}$	$1.3e4_{1.1e3}$	+
	10	$1.4e4_{9.0e2}$	$1.6e4_{1.4e3}$	$4.0e3_{1.1e3}$	$1.0e4_{6.5e2}$	$9.5e3_{2.8e3}$	$1.2e4_{1.5e3}$	$1.4e4_{1.3e3}$	+
	20	$1.4e4_{9.5e2}$	$1.7e4_{1.2e3}$	$5.0e3_{1.5e3}$	$1.1e4_{6.0e2}$	$1.0e4_{3.0e3}$	$1.3e4_{1.4e3}$	$1.4e4_{1.2e3}$	+
	50	$1.6e4_{9.5e2}$	$1.8e4_{1.3e3}$	$8.0e3_{2.6e3}$	$1.2e4_{6.0e2}$	$1.1e4_{3.0e3}$	$1.4e4_{1.4e3}$	$1.6e4_{1.3e3}$	+
	100	$1.7e4_{9.0e2}$	$2.0e4_{1.3e3}$	$1.6e4_{9.9e5}$	$1.3e4_{5.0e2}$	$1.2e4_{3.0e3}$	$1.5e4_{1.6e3}$	$1.7e4_{1.2e3}$	+
ZDT4	1	$1.7e4_{3.0e3}$	$1.9e4_{3.7e3}$	$2.9e4_{1.8e4}$	—	$8.0e2_{6.0e2}$	$1.6e4_{5.7e3}$	$1.1e4_{3.9e3}$	+
	5	$2.1e4_{4.7e3}$	$2.3e4_{4.4e3}$	$4.3e4_{2.4e4}$	—	$2.6e3_{1.0e3}$	$1.7e4_{5.3e3}$	$1.5e4_{4.0e3}$	+
	10	$2.4e4_{6.2e3}$	$2.7e4_{5.8e3}$	$5.1e4_{2.3e4}$	—	$3.7e3_{1.4e3}$	$2.0e4_{6.1e3}$	$1.8e4_{5.1e3}$	+
	20	$2.9e4_{7.8e3}$	$3.2e4_{7.0e3}$	$5.6e4_{2.6e4}$	—	$5.2e3_{1.8e3}$	$2.2e4_{7.8e3}$	$2.2e4_{6.9e3}$	+
	50	$3.6e4_{1.0e4}$	$4.0e4_{1.0e4}$	$7.1e4_{9.4e5}$	—	$8.0e3_{2.0e3}$	$2.7e4_{8.2e3}$	$3.0e4_{9.8e3}$	+
	100	$4.3e4_{1.2e4}$	$4.6e4_{1.5e4}$	$7.9e4_{9.4e5}$	—	$1.1e4_{2.1e3}$	$3.2e4_{1.4e4}$	$3.6e4_{1.0e4}$	+
ZDT6	1	$2.0e4_{1.2e3}$	$2.4e4_{1.3e3}$	$2.2e3_{8.0e2}$	$2.8e3_{4.5e2}$	$1.6e3_{6.0e2}$	$1.2e4_{1.0e3}$	$1.4e4_{1.2e3}$	+
	5	$2.2e4_{1.2e3}$	$2.5e4_{1.4e3}$	$2.3e3_{7.0e2}$	$3.2e3_{4.0e2}$	$1.8e3_{7.0e2}$	$1.3e4_{1.2e3}$	$1.5e4_{1.2e3}$	+
	10	$2.2e4_{1.1e3}$	$2.6e4_{1.4e3}$	$2.4e3_{9.0e2}$	$3.4e3_{3.0e2}$	$2.0e3_{7.0e2}$	$1.3e4_{1.4e3}$	$1.6e4_{1.4e3}$	+
	20	$2.3e4_{1.1e3}$	$2.7e4_{1.4e3}$	$2.6e3_{1.0e3}$	$3.5e3_{3.0e2}$	$2.1e3_{7.0e2}$	$1.4e4_{1.2e3}$	$1.7e4_{1.5e3}$	+
	50	$2.5e4_{1.4e3}$	$2.9e4_{1.4e3}$	$3.3e3_{1.7e3}$	$3.8e3_{3.0e2}$	$2.4e3_{7.0e2}$	$1.5e4_{1.4e3}$	$1.9e4_{1.4e3}$	+
	100	$2.8e4_{1.4e3}$	$3.3e4_{1.3e3}$	$5.8e3_{1.0e6}$	$4.1e3_{3.5e2}$	$2.7e3_{7.0e2}$	$1.7e4_{1.4e3}$	$2.2e4_{1.7e3}$	+

DOI: 10.1002/nme

Table XVI. Median ($\tilde{x}$) and interquartile range (IQR) of the number of evaluations for reaching Pareto optimal points in the DTLZ benchmark when using an external unbounded archive for avoiding Pareto drift issues (cells with dark and light backgrounds indicate the best and the second best values, respectively). PAES, GDE3, and SMPSO provide the best overall results.

		NSGA-II	SPEA2	PAES	GDE3	SMPSO	AbYSS	MOCell	
Problem	n	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	$\tilde{x}_{\text{IQR}}$	
DTLZ1	1	$2.4e4_{1.1e4}$	$2.4e4_{7.2e3}$	$2.6e4_{1.5e4}$	$9.3e3_{1.0e3}$	$6.5e2_{4.0e2}$	$2.8e4_{1.5e4}$	$2.0e4_{6.2e3}$	+
	5	$2.6e4_{1.1e4}$	$2.5e4_{7.8e3}$	$2.7e4_{1.6e4}$	$9.7e3_{9.5e2}$	$2.3e3_{1.6e3}$	$2.9e4_{1.6e4}$	$2.1e4_{6.9e3}$	+
	10	$2.6e4_{1.1e4}$	$2.5e4_{7.7e3}$	$2.9e4_{1.6e4}$	$9.9e3_{9.0e2}$	$3.4e3_{2.0e3}$	$3.0e4_{1.6e4}$	$2.1e4_{6.8e3}$	+
	20	$2.7e4_{1.1e4}$	$2.6e4_{8.2e3}$	$3.1e4_{2.1e4}$	$1.0e4_{9.5e2}$	$4.4e3_{2.2e3}$	$3.0e4_{1.5e4}$	$2.2e4_{7.0e3}$	+
	50	$2.9e4_{1.2e4}$	$2.7e4_{8.6e3}$	$4.2e4_{9.7e5}$	$1.0e4_{9.0e2}$	$5.7e3_{2.2e3}$	$3.1e4_{1.5e4}$	$2.4e4_{7.4e3}$	+
	100	$2.9e4_{1.2e4}$	$2.8e4_{8.6e3}$	$5.2e4_{9.6e5}$	$1.1e4_{9.0e2}$	$6.9e3_{2.6e3}$	$3.2e4_{1.4e4}$	$2.5e4_{7.8e3}$	+
DTLZ2	1	$3.6e3_{5.0e2}$	$3.8e3_{6.0e2}$	$7.0e2_{2.0e2}$	$4.5e3_{4.0e2}$	$6.0e2_{3.0e2}$	$2.2e3_{5.0e2}$	$2.2e3_{7.0e2}$	+
	5	$4.0e3_{5.0e2}$	$4.4e3_{7.0e2}$	$9.0e2_{4.0e2}$	$5.0e3_{3.0e2}$	$1.5e3_{5.0e2}$	$2.5e3_{5.0e2}$	$2.8e3_{8.0e2}$	+
	10	$4.3e3_{4.5e2}$	$4.7e3_{7.0e2}$	$1.2e3_{6.0e2}$	$5.3e3_{3.0e2}$	$2.0e3_{6.0e2}$	$2.7e3_{5.0e2}$	$3.3e3_{8.0e2}$	+
	20	$4.7e3_{5.0e2}$	$5.1e3_{7.5e2}$	$1.7e3_{7.0e2}$	$5.5e3_{3.0e2}$	$2.6e3_{8.0e2}$	$2.9e3_{5.0e2}$	$3.7e3_{8.0e2}$	+
	50	$5.3e3_{6.0e2}$	$5.8e3_{7.0e2}$	$3.1e3_{1.0e3}$	$6.1e3_{4.0e2}$	$3.5e3_{1.0e3}$	$3.4e3_{5.5e2}$	$4.6e3_{7.0e2}$	+
	100	$6.0e3_{6.0e2}$	$6.5e3_{8.0e2}$	$5.3e3_{1.0e6}$	$6.8e3_{3.0e2}$	$4.6e3_{1.2e3}$	$4.0e3_{4.5e2}$	$5.4e3_{7.0e2}$	+
DTLZ3	1	$6.2e4_{2.6e4}$	$6.4e4_{1.7e4}$	$9.0e4_{6.1e4}$	—	$1.1e3_{6.0e2}$	$7.8e4_{3.7e4}$	$4.6e4_{1.6e4}$	+
	5	$6.4e4_{2.5e4}$	$6.6e4_{1.8e4}$	$9.7e4_{6.7e4}$	—	$3.7e3_{8.2e3}$	$8.1e4_{3.7e4}$	$4.8e4_{1.7e4}$	+
	10	$6.6e4_{2.4e4}$	$6.7e4_{1.9e4}$	$1.0e5_{7.7e4}$	—	$5.1e3_{8.2e3}$	$8.2e4_{3.9e4}$	$5.0e4_{1.7e4}$	+
	20	$6.7e4_{2.6e4}$	$6.9e4_{2.0e4}$	$1.2e5_{8.8e4}$	—	$6.1e3_{8.4e3}$	$8.3e4_{4.1e4}$	$5.1e4_{1.8e4}$	+
	50	$7.1e4_{2.6e4}$	$7.3e4_{2.2e4}$	$1.7e5_{9.0e5}$	—	$7.5e3_{9.2e3}$	$8.4e4_{4.1e4}$	$5.3e4_{1.9e4}$	+
	100	$7.4e4_{2.6e4}$	$7.6e4_{2.3e4}$	$2.2e5_{8.7e5}$	—	$8.7e3_{8.8e3}$	$8.7e4_{4.1e4}$	$5.5e4_{1.9e4}$	+
DLTZ4	1	—	—	—	—	—	—	—	−
	5	—	—	—	—	—	—	—	−
	10	—	—	—	—	—	—	—	−
	20	—	—	—	—	—	—	—	−
	50	—	—	—	—	—	—	—	−
	100	—	—	—	—	—	—	—	−
DLTZ5	1	$3.5e3_{5.0e2}$	$3.9e3_{7.0e2}$	$7.0e2_{3.0e2}$	$4.5e3_{5.0e2}$	$6.0e2_{3.0e2}$	$2.2e3_{6.5e2}$	$2.3e3_{9.0e2}$	+
	5	$4.1e3_{5.0e2}$	$4.5e3_{7.0e2}$	$1.0e3_{4.0e2}$	$5.0e3_{3.0e2}$	$1.5e3_{6.0e2}$	$2.5e3_{7.0e2}$	$3.0e3_{8.5e2}$	+
	10	$4.4e3_{5.0e2}$	$4.8e3_{7.0e2}$	$1.4e3_{4.0e2}$	$5.2e3_{3.0e2}$	$2.0e3_{7.0e2}$	$2.6e3_{6.5e2}$	$3.3e3_{9.0e2}$	+
	20	$4.7e3_{4.0e2}$	$5.2e3_{6.0e2}$	$1.9e3_{5.0e2}$	$5.5e3_{3.0e2}$	$2.6e3_{8.0e2}$	$2.9e3_{6.5e2}$	$3.8e3_{8.5e2}$	+
	50	$5.3e3_{5.0e2}$	$5.8e3_{7.0e2}$	$3.0e3_{8.0e2}$	$6.1e3_{3.0e2}$	$3.5e3_{9.0e2}$	$3.4e3_{6.0e2}$	$4.5e3_{8.0e2}$	+
	100	$6.0e3_{5.0e2}$	$6.6e3_{7.0e2}$	$4.9e3_{8.0e2}$	$6.8e3_{3.5e2}$	$4.6e3_{1.3e3}$	$4.0e3_{8.0e2}$	$5.5e3_{8.5e2}$	+
DLTZ6	1	—	—	$3.7e3_{1.8e3}$	$2.5e3_{2.0e2}$	$4.3e3_{2.0e3}$	—	—	+
	5	—	—	$3.8e3_{1.8e3}$	$2.7e3_{2.0e2}$	$4.6e3_{2.0e3}$	—	—	+
	10	—	—	$4.1e3_{2.0e3}$	$2.8e3_{1.0e2}$	$4.7e3_{2.2e3}$	—	—	+
	20	—	—	$4.5e3_{2.3e3}$	$3.0e3_{2.0e2}$	$4.9e3_{2.2e3}$	—	—	+

DOI: 10.1002/nme

Table XVI. *Continued.*

	50	—	—	$5.3e3_{2.4e3}$	$3.2e3_{1.0e2}$	$5.1e3_{2.2e3}$	—	—	+
	100	—	—	$7.0e3_{2.2e3}$	$3.6e3_{2.0e2}$	$5.4e3_{2.1e3}$	—	—	+
DTLZ7	1	$1.3e4_{1.0e3}$	$1.5e4_{1.2e3}$	$2.4e3_{7.0e2}$	$7.0e3_{6.0e2}$	$3.4e3_{1.4e3}$	$9.3e3_{1.2e3}$	$1.0e4_{9.5e2}$	+
	5	$1.3e4_{9.5e2}$	$1.6e4_{1.0e3}$	$3.0e3_{9.0e2}$	$7.6e3_{5.0e2}$	$3.7e3_{1.2e3}$	$9.7e3_{1.2e3}$	$1.1e4_{1.0e3}$	+
	10	$1.4e4_{9.0e2}$	$1.6e4_{1.0e3}$	$3.4e3_{8.5e2}$	$7.8e3_{5.0e2}$	$3.9e3_{1.2e3}$	$9.9e3_{1.2e3}$	$1.2e4_{1.1e3}$	+
	20	$1.4e4_{9.0e2}$	$1.7e4_{1.0e3}$	$4.0e3_{1.2e3}$	$8.1e3_{5.0e2}$	$4.2e3_{1.2e3}$	$1.0e4_{1.1e3}$	$1.2e4_{1.0e3}$	+
	50	$1.5e4_{9.5e2}$	$1.8e4_{1.1e3}$	$5.9e3_{1.2e3}$	$8.6e3_{5.0e2}$	$4.6e3_{1.3e3}$	$1.1e4_{1.0e3}$	$1.3e4_{1.2e3}$	+
	100	$1.6e4_{9.5e2}$	$1.9e4_{1.0e3}$	$8.3e3_{1.4e3}$	$9.3e3_{5.0e2}$	$5.0e3_{1.4e3}$	$1.2e4_{1.2e3}$	$1.4e4_{1.2e3}$	+

Two new algorithms have been included in this comparison: on the one hand, we have replaced OMOPSO by an improved version, SMPSO; on the other hand, we have also included GDE3. If we consider SMPSO, its results improve significantly those obtained by OMOPSO in [18]. Specifically, SMPSO has been the fastest algorithm in the present study, outperforming the behavior of AbYSS and, particularly, MOCell, which was the fastest algorithm in our previous study [18].

7. CONCLUSIONS AND FUTURE WORK

We have analyzed the convergence speed of seven state-of-the-art multi-objective metaheuristics in order to study their efficiency in terms of the required number of evaluations, when converging toward the Pareto optimal front. The benchmark has been composed of 12 bi-objective problems from the ZDT and DTLZ test suites. In order to perform a broader analysis, we have proposed three different convergence criteria based on

- determining the number of Pareto optimal solutions generated by each algorithm,
- the convergence of that approximation to the true Pareto front, and,
- both, the convergence and the diversity of that Pareto front.

Our study has revealed that the analyzed particle swarm optimization and differential evolution algorithms are the most promising approaches to deal with the problems used in this work. Furthermore, it is also worth mentioning that PAES is the fastest algorithm in obtaining a pre-fixed number of Pareto optimal solutions, but it fails when the aim is to compute an entire front with good convergence and diversity.

The results have shown that MOCell and AbYSS, the cGA and scatter search metaheuristics, respectively, are in general in the middle of the rankings according to the three criteria. The genetic algorithms NSGA-II and SPEA2 are always in the last positions, which indicates that, in the context of the considered convergence criteria, benchmark problems, and parameters settings, they are far from being the best choices in terms of efficiency. These four algorithms share in common the use of the SBX crossover operator, which suggests an open research line consisting in studying whether or not the use of other crossover operators can make them converge faster.

Furthermore, the effects of the Pareto drift issue (loss of solutions) in the convergence speed of the algorithms have been analyzed with respect to the first criterion. Although the algorithms have

DOI: 10.1002/nme

converged faster when using such criterion, the behavior of the algorithms, when compared with respect to each other, did not change.

Our next step is an extension of this work including not only benchmark problems but also real-world problems from industry in order to assess whether or not the features of these problems confirm the results obtained in this work. Furthermore, it is also a matter of future research to extend the convergence criteria so that they can cover problems with a higher number of objectives.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their valuable comments, which greatly helped them to improve the contents of this paper.

J. J. Durillo, A. J. Nebro, F. Luna, and E. Alba are with the Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, Spain (e-mail: {durillo, antonio, flv, eat}@lcc.uma.es). C. A. Coello Coello is with the Department of Computer Science, CINVESTAV-IPN, Mexico (e-mail: {ccoello@cs.cinvestav.mx). He gratefully acknowledges the support of CONACyT project no. 103570 and UMI-LAFMIA 3175 CNRS at CINVESTAV-IPN.

This work has been partially funded by the 'Consejería de Innovación, Ciencia y Empresa', Junta de Andalucía under contract P07-TIC-03044 DIRICOM project (`http://diricom.lcc.uma.es`), and the Spanish Ministry of Science and Innovation and FEDER under contract TIN2008-06491-C04-01 (the M* project). Juan J. Durillo is supported by grant AP-2006-03349 from the Spanish government. F. Luna acknowledges support from the Spanish Ministry of Education and Science and FEDER under contract TIN2005-08818-C04-01 (the OPLINK project).

REFERENCES

1. Glover FW, Kochenberger GA. *Handbook of Metaheuristics*. Kluwer: Dordrecht, 2003.
2. Blum C, Roli A. Metaheuristics in combinatorial optimization: overview and conceptual comparison. *ACM Computing Surveys* 2003; **35**(3):268–308.
3. Alba E. *Parallel Metaheuristics*: *A New Class of Algorithms*. Wiley: New York, 2005.
4. Goel T, Vaidyanathan R, Haftka RT, Shyy W, Queipo NV, Tucker K. Response surface approximation of Pareto optimal front in multi-objective optimization. *Computer Methods in Applied Mechanics and Engineering* 2007; **196**:879–893.
5. Lian Y, Liou M. Multiobjective optimization using coupled response surface model and evolutionary algorithm. *AIAA Journal* 2005; **43**(6):1316–1325.
6. Coello C, Van Veldhuizen D, Lamont G. *Evolutionary Algorithms for Solving Multi-objective Problems*. Genetic Algorithms and Evolutionary Computation. Kluwer: Dordrecht, 2002.
7. Deb K. *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley: New York, 2001.
8. Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation* 1999; **3**(4):257–271.
9. Knowles J, Thiele L, Zitzler E. A tutorial on the performance assessment of stochastic multiobjective optimizers. *Technical Report 214*, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.
10. Coello CAC, Pulido GT. A micro-genetic algorithm for multiobjective optimization. *Evolutionary Multi-criterion Optimization. First International Conference*, *EMO 2001*. Lecture Notes in Computer Science. Springer: Berlin, 2001.
11. Pulido GT, Coello CAC. The micro genetic algorithm 2: towards on-line adaptation in evolutionary multiobjective optimization. *Evolutionary Multi-Criterion Optimization. First International Conference*, *EMO 2003*. Lecture Notes in Computer Science. Springer: Berlin, 2003.
12. Santana-Quintero LV, Ramírez-Santiago N, Coello Coello CA, Molina Luque J, Hernández-Díaz AG. A new proposal for multiobjective optimization using particle swarm optimization and rough sets theory. *Parallel Problem Solving from Nature* (*PPSN IX*). Lecture Notes in Computer Science, vol. 4193. Springer: Berlin, 2006; 483–492.
13. Hernández-Díaz AG, Santana-Quintero LV, Coello Coello C, Caballero R, Molina J. A new proposal for multi-objective optimization using differential evolution and rough sets theory. In *Genetic and Evolutionary Computation Conference* (*GECCO'2006*), Keijzer M *et al.* (eds). ACM: New York, NY, U.S.A., 2006; 675–682.

DOI: 10.1002/nme

14. Toscano-Pulido G, Coello Coello CA, Santana-Quintero LV. EMOPSO: a multi-objective particle swarm optimizer with emphasis on efficiency. *Evolutionary Multi-criterion Optimization* (*EMO 2007*). Lecture Notes in Computer Science, vol. 4403. Springer: Berlin, 2007; 272–285.
15. Eskandari H, Geiger CD, Lamont GB. FastPGA: a dynamic population sizing approach for solving expensive multiobjective optimization problems. In *Evolutionary Multi-criterion Optimization. Fourth International Conference, EMO 2007*, Obayashi S, Deb K, Poloni C, Hiroyasu T, Murata T (eds). Lecture Notes in Computer Science, vol. 4403. Springer: Berlin, 2007; 141–155.
16. Knowles J. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation* 2006; **10**(1):50–66.
17. Sindhya K, Deb K, Miettinen K. A local search based evolutionary multi-objective optimization approach for fast and accurate convergence. In *Parallel Problem Solving for Nature*, Rudolph G, Jansen T, Lucas S, Poloni C, Beume N (eds). Springer: Berlin, Germany, 2008; 815–824.
18. Nebro AJ, Durillo JJ, Coello CAC, Luna F, Alba E. A study of convergence speed in multi-objective metaheuristics. In *Parallel Problem Solving for Nature*, Rudolph G, Jansen T, Lucas S, Poloni C, Beume N (eds). Springer: Berlin, Germany, 2008; 763–772.
19. Wolpert DH, Macready WG. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* 1997; **1**(1):67–82.
20. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 2002; **6**(2):182–197.
21. Zitzler E, Laumanns M, Thiele L. SPEA2: improving the strength Pareto evolutionary algorithm. *EUROGEN 2001*, Athens, Greece, 2002; 95–100.
22. Knowles J, Corne D. The Pareto archived evolution strategy: a new baseline algorithm for multiobjective optimization. *Proceedings of the 1999 Congress on Evolutionary Computation*. IEEE Press: Piscataway, NJ, 1999; 9–105.
23. Kukkonen S, Lampinen J. GDE3: the third evolution step of generalized differential evolution. *IEEE Congress on Evolutionary Computation* (*CEC'2005*), Edinbourgh, U.K., 2005; 443–450.
24. Nebro A, Durillo J, García-Nieto J, Coello CC, Luna F, Alba E. SMPSO: a new PSO-based metaheuristic for multi-objective optimization. *Proceedings of the IEEE Symposium Series on Computational Intelligence*, Nashville, TN, U.S.A., 2009; 66–73.
25. Nebro AJ, Luna F, Alba E, Dorronsoro B, Durillo JJ, Beham A. AbYSS: adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation* 2008; **12**(4):439–457.
26. Nebro A, Durillo J, Luna F, Dorronsoro B, Alba E. A cellular genetic algorithm for multiobjective optimization. In *Proceedings of the Workshop on Nature Inspired Cooperative Strategies for Optimization* (*NICSO 2006*), Pelta DA, Krasnogor N (eds), Granada, Spain, 2006; 25–36.
27. Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: empirical results. *IEEE Transactions on Evolutionary Computation* 2000; **8**(2):173–195.
28. Deb K, Thiele L, Laumanns M, Zitzler E. Scalable test problems for evolutionary multiobjective optimization. In *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, Abraham A, Jain L, Goldberg R (eds). Springer: U.S.A., 2005; 105–145.
29. Huband S, Hingston P, Barone L, While RL. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation* 2006; **10**(5):477–506.
30. Li H, Zhang Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation* 2008; **12**(2):284–302.
31. Zhang Q, Zhou A, Zhao S, Suganthan P, Liu W, Tiwari S. Multiobjective optimization test instances for the CEC 2009 special session and competition. *Technical Report CES-487*, Department of Computing and Electronic Systems, University of Essex, Colchester, U.K., 2008.
32. Durillo J, Nebro A, Luna F, Dorronsoro B, Alba E. jMetal: a Java framework for developing multi-objective optimization metaheuristics. *Technical Report ITI-2006-10*, Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, E.T.S.I. Informática, Campus de Teatinos, 2006.
33. Zitzler E, Laumanns M, Thiele L. SPEA2: improving the strength Pareto evolutionary algorithm. *Technical Report 103*, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 2001.
34. Lampinen J. DE's selection rule for multiobjective optimization. *Technical Report*, Department of Information Technology, Lappeenranta University of Technology, 2001.
35. Nebro A, Durillo J, Luna F, Dorronsoro B, Alba E. Design issues in a multiobjective cellular genetic algorithm. In *Evolutionary Multi-criterion Optimization. Fourth International Conference*, *EMO 2007*, Obayashi S, Deb K,

DOI: 10.1002/nme

Poloni C, Hiroyasu T, Murata T (eds). Lecture Notes in Computer Science, vol. 4403. Springer: Berlin, 2007; 126–140.
36. Demšar J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 2006; **7**:1–30.
37. Everson M, Fieldsend J, Singh S. Full elite sets for multi-objective optimization. *Proceedings of the Fifth International Conference on Adaptive Computing Design and Manufacture* (*ACDM 2002*), Exeter, Devon, U.K., 2002; 343–354.
38. Fieldsend J, Everson R, Singh S. Using unconstrained elite archives for multiobjective optimization. *IEEE Transactions on Evolutionary Computation* 2003; **7**(3):305–323.
39. Knowles J, Corne D. Bounded Pareto archiving: theory and practice. *Metaheuristics for Multiobjective Optimisation*. Lecture Notes in Economics and Mathematical Systems, vol. 535. Springer: Berlin, 39–64.
40. Goel T, Vaidyanathan R, Haftka R, Shyy W, Queipo N, Tucker K. Response surface approximation of Pareto optimal front in multi-objective optimization. *Computer Methods in Applied Mechanics and Engineering* 2007; **196**:879–893.

DOI: 10.1002/nme