Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

A Co-evolutionary Scheme for Multi-Objective Evolutionary Algorithms based on ϵ-dominance

Adriana Menchaca-Méndez[1], Elizabeth Montero[2], (Member, IEEE), Luis Miguel Antonio[3], Saúl Zapotecas-Martínez[4], (Member, IEEE), Carlos A. Coello Coello[3], (Fellow, IEEE), María-Cristina Riff[2], (Member, IEEE)

[1]Licenciatura en Tecnologías para la Información en Ciencias, ENES, Campus Morelia, UNAM, México, (e-mail: adriana.menchacamendez@gmail.com)
[2]Departamento de Informática, Universidad Técnica Federico Santa María, Valparaíso, Chile; Universidad Andres Bello, Facultad de Ingeniería, Viña del Mar, Chile (e-mail: elizabeth.montero@unab.cl, Maria-Cristina.Riff@inf.utfsm.cl)
[3]Departamento de Computación, CINVESTAV, IPN, Ciudad de México, México, (e-mail: lmiguel@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx)
[4]Departamento de Matemáticas Aplicadas y Sistemas, Universidad Autónoma Metropolitana Unidad Cuajimalpa, Ciudad de México, México, (e-mail: szapotecas@correo.cua.uam.mx)

Corresponding author: Adriana Menchaca-Méndez (e-mail: adriana.menchacamendez@gmail.com).

First author acknowledges support from PAPIIT project IA105918. Second author acknowledges support from Fondecyt project 11150787. Fifth author gratefully acknowledges support from CONACyT project 221551.

ABSTRACT Convergence and diversity of solutions play an essential role in the design multi-objective evolutionary algorithms (MOEAs). Among the available diversity mechanisms, the ϵ-dominance has shown a proper balance between convergence and diversity. When using ϵ-dominance, diversity is ensured by partitioning the objective space into boxes of size ϵ and, typically, a single solution is allowed at each of these boxes. However, there is no easy way to determine the precise value of ϵ. In this paper, we investigate how this goal can be achieved by using a co-evolutionary scheme that looks for the proper values of ϵ along the search without any need of a previous user's knowledge. We include the proposed co-evolutionary scheme into an MOEA based on ϵ-dominance giving rise to a new MOEA. We evaluate the proposed MOEA solving standard benchmark test problems. According to results, it is a promising alternative for solving multi-objective optimization problems because three main reasons: i) it is competitive concerning state-of-the-art MOEAs, ii) it does not need extra information about the problem, and iii) it is computationally efficient.

INDEX TERMS Evolutionary Multi-objective Algorithms, ϵ-dominance, co-evolutionary schemes, parameter setting.

I. INTRODUCTION

MANY real-world applications involve the solution of problems with multiple conflicting objective functions which have to be simultaneously optimized. They are called *multi-objective optimization problems (MOPs)*. Since their objective functions conflict with each other, MOPs do not have a single optimal solution but a set of trade-off solutions for which no objective can be improved without worsening any other. This set of solutions is called *Pareto optimal set ($\mathcal{PS}$)*, and its image is known as *Pareto front ($\mathcal{PF}$)*.

In recent years, the use of *multi-objective evolutionary algorithms* (MOEAs) to solve MOPs has become very popular. Since their origins, MOEAs have had two main goals [1]: (i) to find solutions that are as close as possible to $\mathcal{PF}$ and, (ii) to produce solutions that are spread along $\mathcal{PF}$ as uniformly as possible. We can classify MOEAs according to their selection mechanism, in two classes: (i) those that incorporate the concept of Pareto dominance, and (ii) those that do not use Pareto dominance. MOEAs of type (i) have several disadvantages, e.g., they cannot scale appropriately regarding the number of objective functions, because the number of nondominated solutions overgrows as the number of objective functions is increased. This, in fact, dilutes the effect of the selection mechanism of an MOEA [2] in high-dimensional objective spaces. For this reason, the use of MOEAs of type (ii) has steadily grown in the last few years.

In this work, we are interested in MOEAs based on ϵ-dominance (MOEAs of type (ii)) which have shown to be an

excellent alternative to deal with the two challenges that we mentioned before (convergence and distribution of solutions along the $\mathcal{PF}$).

For instance, Deb et al. [3] proposed the ϵ-MOEA. This MOEA uses an archive with a fixed size in which the nondominated solutions are stored. The idea is that the search space is divided into hypercubes of size equal to ϵ and only one nondominated individual can reside in each hypercube. ϵ-MOEA is computationally efficient. However, the most critical disadvantage of MOEAs based on ϵ-dominance is related to set the value of ϵ correctly. Therefore, to define a proper ϵ value, it is necessary to know the $\mathcal{PF}$ as well as the number of nondominated solutions that we want to store.

In our investigation, we depart from the following hypothesis: The optimal value of ϵ depends on the MOP that we want to solve. Besides, we consider the possibility of having a different ϵ value per objective function. To validate our hypothesis, we use the "Evolutionary Calibrator (EVOCA) [4]" which is a specialized parameters tuning method. Once the hypothesis is validated, we propose a co-evolutionary scheme to set the ϵ values which can be adopted by any ϵ-dominance-based MOEA. Our proposed approach employs two populations: (i) a population of candidate solutions of the MOP and (ii) a population of individuals representing the possible values of ϵ. For the latter, we propose new operators (crossover and mutation) for evolving the ϵ individuals, and we also define the fitness function for them.

To validate our co-evolutionary scheme, we incorporate it into an MOEA based on ϵ-dominance called "Generational Distance and ϵ-dominance-based Multi-Objective Evolutionary Algorithm (GDE-MOEA) [5]". GDE-MOEA incorporates a mechanism to find the right values of ϵ during the search. Therefore, we can validate whether our co-evolutionary scheme can improve it, regarding the quality of the $\mathcal{PF}$ approximation obtained and the consumption time required. The resulting approach is called "Co-evolutionary Generational Distance and ϵ-dominance-based Multi-Objective Evolutionary Algorithm (CoGDE-MOEA)".

We compare CoGDE-MOEA concerning GDE-MOEA, MOEA/D [6], and a version of SMS-EMOA [7] that uses an approximation of the hypervolume indicator [8] as its fitness assignment scheme, we call this version hypeSMS-EMOA. As we will see later on, the proposed co-evolutionary scheme is a viable alternative to deal efficiently with MOPs, since CoGDE-MOEA can outperform the original GDE-MOEA and a version of GDE-MOEA that incorporates the ϵ values tuned by EVOCA. Additionally, we show the potential of our proposed approach when solving MOPs with both low (three objective functions) and high (four, five and six objective functions) dimensionality. CoGDE-MOEA was able to outperform MOEA/D and obtained competitive results concerning hypeSMS-EMOA but at a lower computational cost.

The main contributions of our work can be summarized as follows:

1) Validation of the hypothesis: *The optimal value of ϵ depends on the MOP that we want to solve. We also consider the possibility of having a different ϵ value per objective function*. In Section III, we can see that none of the previous works make an in-depth study of this hypothesis and they assume that it is true.
2) A co-evolutionary scheme for MOEAs based on ϵ-dominance that looks for the most suitable values of ϵ during the search without any need of previous user's knowledge.

The remainder of this paper is organized as follows. Section II states the problem of our interest and explains the concept of ϵ-dominance. The motivation and previous related work on MOEAs based on ϵ-dominance are presented in Section III. Section IV presents a study about suitable ϵ values for problems with different features (e.g., linear, concave, degenerate, and disconnected Pareto Fronts). Our proposal is explained in Section V. Our experimental setup and the obtained results are provided in Section VI. Finally, our conclusions and some possible paths for future research are discussed in Section VII.

II. BACKGROUND

Without loss of generality, assuming minimization, a *multi-objective optimization problem (MOP)* consists in finding the vector of decision variables $\mathbf{x}^\star = (x_1^\star, x_2^\star, \ldots, x_n^\star)^T$ that minimizes:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T \quad (1)$$

such that $\mathbf{x}^\star \in \Omega$, where $\Omega \subset \mathbb{R}^n$ defines the feasible region of the problem. The optimal solutions of an MOP are defined by the concept of *dominance*: a vector $\mathbf{y} \in \Omega$ is *dominated* by a vector $\mathbf{x} \in \Omega$ ($\mathbf{x} \prec \mathbf{y}$) if $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ for all $i = 1, \ldots, k$, and there exists a j such that $f_j(\mathbf{x}) < f_j(\mathbf{y})$. Otherwise, $\mathbf{y}$ is nondominated by $\mathbf{x}$. A decision vector $\mathbf{x} \in \Omega$ is called *Pareto optimal* if there is no $\mathbf{y} \in \Omega$ which dominates $\mathbf{x}$. The set of all Pareto optimal solutions is called *Pareto optimal set* ($\mathcal{PS}$), and its image is called *Pareto Front* ($\mathcal{PF}$).

The concept of ϵ-dominance was proposed by Laumanns et al. [9] as an archiving technique. As mentioned before, ϵ-dominance divides the whole objective space into hypercubes, each having an ϵ_j size in the j^{th} objective. For this, an identification vector, $\mathbf{b} = (b_1, b_2, \cdots, b_k)^T$, is defined for each solution $\mathbf{x}$ as follows:

$$b_j = (\lfloor (f_j(\mathbf{x}) - f_j^{min})/\epsilon_j \rfloor) \times \epsilon_j, \quad j = 1, \cdots, k \quad (2)$$

where f_j^{min} is the minimum value of the j^{th} objective and $\epsilon_j > 0$ is the allowable tolerance of the j^{th} objective. Then, we say that all solutions dominated by $\mathbf{b}$ are ϵ-dominated by $\mathbf{x}$. Figure 1 illustrates the concept of ϵ-dominance. It is worth noticing that all points in the same hypercube have the same identification vector.

III. MOTIVATION AND PREVIOUS RELATED WORK

In the specialized literature, we can find several works that use ϵ-dominance in the same way as ϵ-MOEA, see for example those reported in [10]–[16].

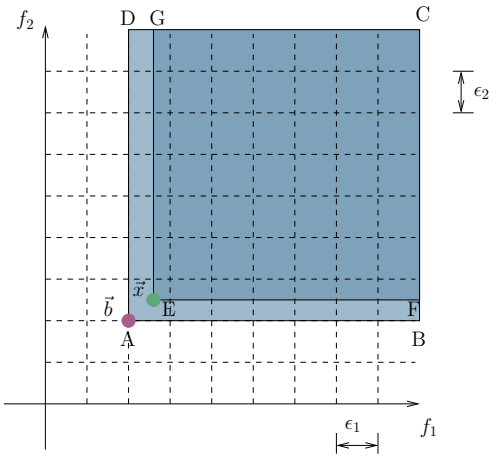

FIGURE 1: Illustration of ϵ-**dominance**. $\mathbf{b}$ is the identification vector of solution $\mathbf{x}$. We say that the area *ABCDA* is ϵ-dominated by $\mathbf{x}$. On the other hand, $\mathbf{x}$ dominates any solution in the area *EFCGE*.

Therefore, they use an ϵ value for all the objective functions of the MOP and, in order to define a suitable ϵ value, it is necessary to know the true $\mathcal{PF}$ as well as the number of nondominated solutions to be reached. Evidently, this is not possible to establish *a priori* in real-world applications where the features of the problem are unknown. Nonetheless, some investigations try to address this issue, and we discuss them below.

Sato et al. [17] proposed a method to control the degree of expansion or contraction of the dominance area of solutions using a user-defined parameter S. Although the authors did not use ϵ-dominance, they modified the fitness value for each objective function allowing to modify the size of the dominance area. This has a strong connection with MOEAs based on ϵ-dominance if the ϵ values change during the search. One disadvantage that we see in this proposal is that another parameter to determine the degree of expansion or contraction is defined and, apparently, this parameter also depends on the MOP that we want to solve. On the other hand, the authors of this work performed an interesting study in which they analyzed the effects of modifying the dominance area. They concluded that both convergence and diversity are affected, and also showed that the optimal value of the area of dominance depends strongly on the number of objectives, the size of the search space and the complexity of the problem. It is worth noticing that they only used the knapsack problem in their experiments.

Sato et al. [18] addressed the main disadvantage in [17], and they proposed to calculate S during the search. In their investigation, the S value depends on the individual and of the objective function. Therefore, each individual uses a different S value for each objective function, even different individuals use different values. This proposal considers different dominance areas for each objective function. However, it does not take into account the geometrical characteristics of the $\mathcal{PF}$. It only considers the spread of the $\mathcal{PF}$. Moreover, the authors only performed experiments with the knapsack problem.

Hernández-Díaz et al. [19] identified some disadvantages of ϵ-dominance, e.g., MOEAs based on it can lose a high number of efficient solutions if the geometrical characteristics of the $\mathcal{PF}$ are not considered. In order to address this disadvantage, the authors employed not only different values of ϵ for each objective function but also different intensities of dominance which are defined according to the position of each point along the $\mathcal{PF}$. To achieve this, they used a family of curves. The main disadvantage of this proposal is that they require to know the $\mathcal{PF}$ shape to determine the type of curve that should be used.

Aguirre et al. [20] proposed an adaptative ϵ-ranking method to improve Pareto-based selection. The authors discussed the difficulties to know precisely how many solutions should be assigned with the highest rank for a given value of ϵ. The authors noticed that for larger ϵ values decrease the number of highest ranked solutions and vice versa. Thus, the authors proposed to set ϵ values at each generation with the aim that the number of the highest ranked solutions will be close to $\alpha \times |\boldsymbol{P}|$, where $\alpha \in [0, 1]$ is a parameter set by the user and $|\boldsymbol{P}|$ is the population size. If the number of solutions in the first rank is greater than $\alpha \times |\boldsymbol{P}|$, the adaptation step Δ is multiplied by a factor of 2 and added to ϵ. Otherwise, Δ is divided by two and subtracted from ϵ. Δ is kept in the range $[\Delta_{min}, \Delta_{max}]$ and $\epsilon > 0$. In this work, the authors performed a study about α and they used $\Delta_0 = 0.005$ (initial value), $\Delta_{min} = 0.0001$ and $\Delta_{max} = 0.05$ for all experiments. Other works use this proposal, e.g. [21]–[23]. Kowatari et al. [23] studied the behavior of an MOEA setting a fixed value of $\epsilon = \{0.01, 0.1, 0.5, 1\}$ and they concluded that adapting ϵ during the search is better than fixing it with one of the values adopted in their experiments. The method proposed by Aguirre et al. in [20] seems to be an excellent choice to state the ϵ value. However, this proposal possesses two disadvantage: (i) it employs a unique ϵ value for all the objective functions, and (ii) to find the optimal ϵ value can be slow and several iterations of the MOEA could be wasted.

Menchaca et al. addressed the second disadvantage in [5]. In their work, the authors guarantee that at each generation the MOEA can select the number of individuals that are desired in the following way. They proposed a new MOEA called "Generational Distance and ϵ-dominance-based Multi-Objective Evolutionary Algorithm (GDE-MOEA)". This approach uses a selection mechanism based on the generational distance indicator (I_{GD}) [1] to achieve convergence and uses a selection based on ϵ-dominance to explore the whole search space at early stages of the search. and to improve the distribution of solutions along $\mathcal{PF}$ at the end of the search. The authors also used a single value of ϵ for all objective functions of an MOP. To determine the ϵ value, GDE-MOEA divides

[1] I_{GD} reports how far, on average, $\mathcal{A}$ is from the true $\mathcal{PF}$ [24]–[26], where $\mathcal{A}$ is an approximation of the true $\mathcal{PF}$. I_{GD} is Pareto non-compliant.

the objective space into two equal parts for each objective function. Then, it proceeds to select the individuals (it selects a single nondominated individual for each hypercube in the search space) and puts them in a set called $\boldsymbol{S}$. After that, if the desired number of individuals has not been selected (i.e., $|\boldsymbol{S}| < N$, assuming N as the number of individuals to be selected), it divides the objective space into three equal parts for each objective function and selects the remaining individuals. This process is repeated (at each iteration one division is added to each objective function) until selecting N individuals.

Sanghamitra et al. [27] proposed an approach called Priority Based ϵ (PBE). PBE determines the value of ϵ which is different for each objective function. To achieve this, PBE computes the average correlation value of every objective function with all the others, and then, it assigns priorities to each objective: when one objective is highly positively correlated with others, its priority value is low; if the objective is negatively correlated with others, then its priority value is high. This process is performed during a predefined number of iterations. The authors said that the intuition behind their idea is that when an objective conflicts a lot with other objectives, it is considered essential and the ϵ value denotes the amount of relaxation. Thus, for the high priority objective, the ϵ value is lower compared to an objective with a lower priority value. The main disadvantage of this proposal is that Sanghamitra et al. updates the value of ϵ according to the degree of correlation between objective functions without considering the shape of $\mathcal{PF}$ nor the number of objective functions.

The above works assumed that the optimal value of ϵ depends on features of the MOP to be solved, e.g., on the geometrical characteristics of the true $\mathcal{PF}$, on the correlation between the objective functions, on the number of objective functions, and on the size of the search space. Indeed, Hernández-Díaz et al. [19] and Sato et al. [18] proposed to use a different dominance area per objective function. However, none of them makes an in-depth study to validate their corresponding hypotheses. Sato et al. [18] conducted an extensive study but only considered the knapsack problem. Kowatari et al. [23] conducted a study to show that adapting ϵ during the search is better than using a fixed ϵ value. Nevertheless, they only used four values for ϵ in their study.

On the other hand, there are several grid-based MOEAs, see for example [28]–[31]. These MOEAs also divide the search space in hypercubes, but they do not preserve the main ideas of the ϵ-dominance, e.g., in [31] the authors proposed a grid-based MOEA called "GrEA" which can select grid-dominated solutions, where the concept of grid-dominated is very similar to the concept of ϵ-dominance (both are a relaxed form of the Pareto dominance relation). On the other hand, GrEA allows more than one solution per hypercube. We have paid particular attention to GrEA because it adjusts the size of its hypercubes during the search. For that, the number of divisions into the search space is fixed, but it considers the spread of the $\mathcal{PF}$. Although GrEA modifies the grid size to maintain a fixed number of divisions at each axis, it does not consider the shape of the $\mathcal{PF}$ (e.g., if the $\mathcal{PF}$ is disconnected) to determine the hypercube size.

With the aim of addressing the disadvantages detected in the works reviewed in this section, we perform an in-depth study of the proper ϵ values in MOPs with different features. Mainly, we consider multimodal/unimodal MOPs with a different number of objectives, and different geometries of their true $\mathcal{PF}$s (linear, concave, disconnected, etc.). Besides, we propose a co-evolutionary scheme to find the proper values of ϵ that should use the MOEA based on ϵ-dominance at each stage of the search. Since we use another evolutionary algorithm to find the proper ϵ values, we aim that the ϵ-dominance-based MOEA which incorporates our co-evolutionary scheme can deal with MOPs different features.

IV. SUITABLE ϵ VALUES

As a result of the review of state of the art, we have stated the following hypothesis: "The optimal value of ϵ depends on the MOP that we want to solve. We also argue that it is possible to define a different ϵ value for each objective function". In order to validate our hypothesis, we use an algorithm for automatic parameter tuning called "Evolutionary Calibrator (EVOCA) [4]" to calibrate the ϵ value in one of the MOEAs mentioned in Section III: "GDE-MOEA".

GDE-MOEA follows the basic structure of evolutionary algorithms. First, it creates an initial population of size N called P. After that, it creates N new individuals using the operators of NSGA-II [32] (crossover and mutation). It combines the population of parents and offspring to obtain a population of size $2N$. Then, it selects the N individuals that will take part in the following generation (the new population P). Finally, it repeats this process for a (pre-defined) number of generations. It is worth noticing the selection mechanism is applied to the objective function space, and the population has to be normalized.

In Algorithm 1, we show the selection mechanism. In this process, the nondominated individuals are obtained and stored in $\boldsymbol{S}$. If the number of nondominated individuals is lower than N, the generational distance indicator (I_{GD}) is used to select the $(N - |\boldsymbol{S}|)$ remaining individuals, using $\boldsymbol{S}$ as the reference set, see Lines 3–23. The computational cost of this selection procedure is $O(|\boldsymbol{P}|^2)$. We use the concept of ϵ-dominance, if the number of nondominated solutions is greater than N, see Lines 26–47. The computational cost of this procedure is $O(m * |\boldsymbol{P}|^2)$, where m is the number of divisions in each objective function[2].

EVOCA is itself an evolutionary algorithm that works with a population of parameter configurations. A parameter configuration is a set of values for each parameter that the tuned algorithm must set. Algorithm 2 shows EVOCA's structure.

The population size of EVOCA is set considering the number of parameters tuned as well as their domain sizes in

[2]In the multi-objective optimization problems adopted in our experiments, m did not exceed the value of 100.

Algorithm 1: GDE-MOEA Selection

Input : $\boldsymbol{P}$ (population) and N (number of individuals to choose).
Output: $\boldsymbol{S}$ (selected individuals).

Put the nondominated individuals of $\boldsymbol{P}$ in $\boldsymbol{S}$;
if $N > |\boldsymbol{S}|$ **then**
  /* I_{GD}-selection */
  Put the dominated individuals of $\boldsymbol{P}$ in $\boldsymbol{B}$;
  Calculate the Euclidean distance d_i from each individual $\mathbf{x}^i \in \boldsymbol{B}$ to its nearest neighbor in $\boldsymbol{S}$ ($\boldsymbol{S}$ is the reference set) and we also save its closest nondominated neighbor;
  Sort $\boldsymbol{B}$ with respect to d_i (ascending order);
  $\boldsymbol{S'} \leftarrow \emptyset$, $allowedNeighbors \leftarrow 0$, $i \leftarrow 0$;
  while $|\boldsymbol{S'}| < N - |\boldsymbol{S}|$ **do**
    $countNeighbors \leftarrow 0$;
    foreach $\mathbf{s} \in \boldsymbol{S'}$ **do**
      if $\mathbf{s}.neighbor = \boldsymbol{B}.\mathbf{x}^i.neighbor$ **then**
        $countNeighbors \leftarrow countNeighbors + 1$;
    if $countNeighbors \leq allowedNeighbors$ **then**
      $\boldsymbol{S'} \leftarrow \boldsymbol{S'} \cup \boldsymbol{B}.\mathbf{x}^i$, $\boldsymbol{B} \leftarrow \boldsymbol{B} - \boldsymbol{B}.\mathbf{x}^i$;
    else
      $i \leftarrow i + 1$;
    if $i = |\boldsymbol{B}|$ **then**
      $i \leftarrow 0$,
      $allowedNeighbors \leftarrow allowedNeighbors + 1$;
  return $\boldsymbol{S} \cup \boldsymbol{S'}$;
else
  /* ϵ-selection */
  if $N < |\boldsymbol{S}|$ **then**
    $n \leftarrow 1$, $\boldsymbol{S'} \leftarrow \emptyset$;
    while $|\boldsymbol{S'}| < N$ **do**
      $n \leftarrow n + 1$;
      Set the vector $\boldsymbol{\epsilon}$: $\epsilon_j \leftarrow 1/n$ (where j indicates the objective function);
      Update the identification vector, $\mathbf{b}$, for each individual in $\boldsymbol{S}$ and for each individual in $\boldsymbol{S'}$;
      foreach $\mathbf{x}^i \in \boldsymbol{S}$ *and* $|\boldsymbol{S'}| < N$ **do**
        if $\mathbf{x}^i \notin \boldsymbol{S'}$ **then**
          $flag \leftarrow 0$;
          foreach $\mathbf{s}^i \in \boldsymbol{S'}$ **do**
            if $\mathbf{s}^i.\mathbf{b} = \mathbf{x}^i.\mathbf{b}$ **then**
              if $\mathbf{x}^i$ *is nearest to* $\mathbf{b}$ *than* $\mathbf{s}^i$ **then**
                $\mathbf{x}^i$ replaces $\mathbf{s}^i$, $flag \leftarrow 1$;
          if $flag = 0$ **then**
            $\boldsymbol{S'} \leftarrow \boldsymbol{S'} \cup \mathbf{x}^i$;
    return $\boldsymbol{S'}$;
return $\boldsymbol{S}$;

Algorithm 2: EVOCA tuning method

Input : $\boldsymbol{A_t}$ (tuned algorithm), P (parameters), R (repetitions) and pc (precision).
Output: $\boldsymbol{c*}$ (best parameter configuration).

Generate $Population$ of configuations (P, pc, $Evoca_{psize}$);
while *not termination condition met* **do**
  $Child \leftarrow$ Wheel-crossover($Population$);
  Evaluate ($\boldsymbol{A_t}$, $Child$, R);
  Replace the worst configuration in $Population$ by $Child$;
  $Mutatedchild \leftarrow$ Hillclimbing($Child$);
  Evaluate ($\boldsymbol{A_t}$, $Mutatedchild$, R);
  if $Mutatedchild$ *is better than* $Child$ **then**
    Replace the second worst configuration in $Population$ by $Mutatedchild$;
return $\boldsymbol{c*}$ (best parameter configuration in $Population$);

order to include a set of relevant values for each parameter in an independent way on its initial population. For real-valued parameters, an initial precision level (pc) is established in order to determine a maximum number of relevant values for those parameters without exceeding a maximum population size ($Evoca_{psize}$). EVOCA uses two transformation operators.

First, it adopts a wheel-crossover operator that constructs one offspring parameter configuration from the whole population. For each parameter, EVOCA selects its value from the parent configuration according to the roulette-wheel based on the quality of parameter configurations in the current population. The crossed configuration replaces the worst configuration on the current population. Second, it adopts a hill climbing-based mutation operator. Mutation takes a copy of the offspring generated by crossover and tries to improve it by modifying one of its parameter values. Initially, it selects a parameter to mutate and then tries $domain_{size}$ times to generate a new random value on its domain to improve the parameter configuration quality. The precision level of real-valued parameters is increased when using this operator. If a better configuration is obtained, it replaces the second worst configuration in the current population.

EVOCA defines three (meta)-parameters which were set as follows: population size ($Evoca_{psize}$)= 20, number of repetitions (R)=10, and termination condition= 10,000 GDE-MOEA executions. EVOCA has been successfully applied to several tuning scenarios in the literature [33], [34] and its implementation is available at its author's website (http://ecco.informaticae.org/).

In our study, we adopted ϵ as a vector of k components, where k is the number of objective functions. The idea is the following: Let $\boldsymbol{A}$ be the approximate $\mathcal{PF}$ found by GDE-MOEA. Then, we use EVOCA to maximize the hypervolume indicator, i.e., $\max I_H(\boldsymbol{A})$. For this task, we execute Algorithm 2 with $\boldsymbol{A_t}$ = GDE-MOEA, $P = \{\epsilon\}$, $R = 10$, and $pc = 1$ decimal positions. We chose I_H because it is the unique unary indicator which is strictly "Pareto compliant"[3] [35]. Furthermore, I_H rewards convergence towards the $\mathcal{PF}$ as well as the maximum spread of the solutions obtained. Additionally, we adopted four MOPs taken from DTLZ [36] test suite: DTLZ1, DTLZ2, DTLZ5, and DTLZ7, all of them with 3, 4, 5, and 6 objective functions. We chose these problems because they have different characteristics: DTLZ1 is multimodal with a linear $\mathcal{PF}$, DTLZ2 is unimodal with a concave $\mathcal{PF}$, DTLZ5 is unimodal with a degenerate $\mathcal{PF}$, and DTLZ7 has a disconnected $\mathcal{PF}$.

In Table 1, we show the best $\boldsymbol{\epsilon}$ values found by EVOCA for these test problems. It is worth noticing the following:

(i) In all cases, the ϵ value is different for each objective function.

[3]An indicator $I : \Omega \rightarrow \mathbb{R}$ is **Pareto compliant** if for all $\boldsymbol{A}, \boldsymbol{B} \subseteq \Omega : \boldsymbol{A} \preceq \boldsymbol{B} \Rightarrow I(\boldsymbol{A}) \geq I(\boldsymbol{B})$ assuming that greater indicator values correspond to higher quality, where $\boldsymbol{A}$ and $\boldsymbol{B}$ are approximations of the Pareto optimal set, Ω is the feasible region and $\boldsymbol{A} \preceq \boldsymbol{B}$ means that every point $\mathbf{b} \in \boldsymbol{B}$ is weakly dominated by at least one point $\mathbf{a} \in \boldsymbol{A}$.

(ii) There is no relationship between the ϵ values found for the same problem with the different number of objective functions.
(iii) There is no connection between the ϵ values found for problems with different geometrical characteristics in their $\mathcal{PF}$ and the same number of objective functions.

With these results, we can corroborate our hypothesis and the assumptions made by the previous related works. However, a new question arises: Is it better to set the ϵ values during the search instead of using a specialized algorithm to set parameters like EVOCA? From the study done so far, we consider that it could be better to set the ϵ values during the search because of two important reasons: 1) Using a tuning method like EVOCA implies a high computational effort to execute several times the tuned MOEA, which can be very expensive. In these experiments, the tuning processes took from 86 to 262 minutes depending on the number of objectives considered; and 2) Adjusting the ϵ values during the search, the MOEA can exploit the accumulated information at each moment of the search.

Therefore, we propose a co-evolutionary scheme that allows us to adjust the value of ϵ for each objective function as the search progresses. We introduce this scheme in the next Section.

V. OUR PROPOSED APPROACH: COGDE-MOEA

As mentioned before, one of the main problems of using ϵ-dominance is to determine the appropriate values for ϵ in a specific MOP. In this section, we propose a co-evolutionary scheme to deal with this issue.

A. CO-EVOLUTIONARY ALGORITHMS

A co-evolutionary search involves the use of multiple species as the representation of a solution to an optimization problem. Each species population represents a piece of a larger problem, and it is the task of those populations to evolve increasingly fit pieces for the larger problem. Recent work in co-evolutionary algorithms (CAs) research considers co-evolution as a form of multi-objective optimization [1], [37], [38]. In our investigation, the focus is on how coevolution can be integrated in order to provide a better way of computing ϵ values for an MOEA which uses ϵ-dominance as its density estimator. Here, we adopt two populations. The first has individuals who are candidate solutions to solve the MOP (we call it the main population) and the second population contains the possible values of ϵ. In the following, we provide a more detailed description of this approach.

B. ϵ-POPULATION

We propose the use of an evolution strategy $(\mu + 1)$-ES to evolve a population of ϵ-individuals, i.e., we generate a new individual at each iteration and it competes against the μ individuals in the current population (the best μ individuals will survive). We construct an ϵ-population of size equal to a percentage of the main population. At each iteration, we carry out the parent selection as follows: the first parent is the best ϵ-individual in the ϵ-population, and we chose the second parent randomly. We aim to find the ϵ values that allow to select exactly N individuals from the main population. For this reason, the best ϵ-individual is the one who achieves to select a number of individuals of the main population close to N, and we use this individual to guide the search for the best ϵ values in a generation. Algorithm 3 shows how to evolve the ϵ-population.

In the following subsections, we define the representation of an ϵ-individual, its fitness, the initialization process, and the genetic operators (crossover and mutation).

1) Representation

One ϵ-individual represents, for each objective function, the number of divisions on the current objective, e.g., the individual $\boldsymbol{\epsilon}_1 = (12, 16)$ indicates that we will have twelve divisions for the first objective function and sixteen divisions for the second objective function. It is worth mentioning that the objective space is normalized, see Figure 2a. Note besides that ϵ value commonly indicates the size of the hypercube, i.e., for the previous example $\boldsymbol{\epsilon}_1 = (0.08333, 0.0625)$. However, for convenience, we employ the integer representation for the ϵ-individual. This representation does not affect the meaning of ϵ-dominance.

2) Initialization

In our proposed approach, we initialize each component of an ϵ-individual with a random value in the range $[2, 100]$. We chose this range according according to the results shown in Table 1, i.e., the best values for ϵ found by EVOCA, which aimed to maximize the hypervolume of the $\mathcal{PF}$ approximation achieved by GDE-MOEA.

3) Fitness evaluation

Let us assume that we are solving an MOP and we desire to obtain a $\mathcal{PF}$ approximation with N nondominated solutions. If we use an MOEA based on ϵ-dominance with a population size N, we need to know which are the values for each component of vector $\boldsymbol{\epsilon}$ that allow us to select exactly N solutions at each generation. If we are not able to obtain these values, our MOEA will not be able to produce good results.

For the above reason, we evaluate each ϵ-individual as follows. Let us assume that we want to select N solutions from a set $\boldsymbol{P}$ such that $N < |\boldsymbol{P}|$. Our aim is to find an ϵ-individual such that the partitions in the objective space imply to select exactly N solutions. It is worth remembering that we can only select one individual per hypercube. Suppose that the ϵ-individual $\boldsymbol{\epsilon}_1$ allows selecting M individuals. Therefore, we define the fitness of $\boldsymbol{\epsilon}_1$ as the distance between M and N. For example, let us assume that it is required to select 10 individuals in Figure 2. By using $\boldsymbol{\epsilon}_1 = (12, 16)$, we shall select 15 individuals. Therefore, the distance of $\boldsymbol{\epsilon}_1$ is equal to 5 (i.e., $|15 - 10|$). On the other hand, considering $\boldsymbol{\epsilon}_2 = (3, 4)$ then we shall select 5 individuals. Therefore, the distance of $\boldsymbol{\epsilon}_2$ is also 5 (i.e., $|5 - 10|$). However, we consider that $\boldsymbol{\epsilon}_1$ is better than $\boldsymbol{\epsilon}_2$. In the first case, we can randomly

TABLE 1: Best values found by EVOCA for $\boldsymbol{\epsilon}$ using DTLZ1, DTLZ2, DTLZ5 and DTLZ7 with 3, 4, 5, and 6 objective functions.

	3	4	5	6
DTLZ1	$[0.03, 0.058, 0.055]$	$[0.142, 0.09, 0.033, 0.021]$	$[0.058, 0.5, 0.5, 0.029, 0.023]$	$[0.09, 0.25, 0.333, 0.142, 0.25, 0.1]$
DTLZ2	$[0.111, 0.142, 0.125]$	$[0.25, 0.111, 0.5, 0.333]$	$[0.25, 0.166, 0.5, 0.052, 0.055]$	$[0.111, 0.2, 0.111, 0.5, 0.055, 0.022]$
DTLZ5	$[0.013, 0.25, 0.017]$	$[0.014, 0.5, 0.5, 0.333]$	$[0.033, 0.010, 0.071, 0.5, 0.037]$	$[0.1, 0.013, 0.018, 0.5, 0.5, 0.016]$
DTLZ7	$[0.055, 0.111, 0.125]$	$[0.2, 0.09, 0.5, 0.166]$	$[0.166, 0.047, 0.013, 0.017, 0.5]$	$[0.029, 0.01, 0.166, 0.021, 0.071, 0.018]$

select ten individuals from the set of current fifteen well-distributed selected individuals. While in the second case, we have five individuals and we need to select the remaining five individuals from the nondominated individuals that have not been selected yet. Then, the probability that the distribution obtained in the first case is better than the distribution obtained in the second case is high. Therefore, we define the fitness of one ϵ-individual as follows:

$$fitness(\boldsymbol{\epsilon}_j) = -1 \times (|x_j - N| + y_j) \tag{3}$$

where x_j is the number of individuals that we will select if we use $\boldsymbol{\epsilon}_j$ and, y_j is equal to 0.0 if $x_j \geq N$ and it is equal to 0.5 if $x_j < N$. See Lines 2–8 in Algorithm 3.

4) Crossover operator

In our proposed approach, we employ intermediate recombination. Let us assume that we have two parents $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$. Then, the offspring $\boldsymbol{\epsilon}$ is obtained by:

$$\boldsymbol{\epsilon} = \frac{1}{2} \times (\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}_2) \tag{4}$$

5) Mutation operator

We perform the mutation operator of each ϵ-individual according to the following scheme:

1) If both parents select more individuals than required, we reduce the number of divisions in the objective space because we need bigger hypercubes.
2) If both parents select fewer individuals than required, we increase the number of divisions in objective space because we need smaller hypercubes.

Let us assume that $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are the parents of individual $\boldsymbol{\epsilon}$ and such individual $\boldsymbol{\epsilon}$ must be mutated. For each j-th component of $\boldsymbol{\epsilon}$ (i.e., ϵ_j), we simulate a coin toss to decide whether the j-th component is perturbed, see Algorithm 3, Lines 14–15 and 24–25. In the case 1, ϵ_j will be a random value between $0.5 \times \left(\min\{\epsilon_j^1, \epsilon_j^2\}\right)$ and $\min\{\epsilon_j^1, \epsilon_j^2\}$, see Line 16 in Algorithm 3. In the case 2, ϵ_j will be a random value between $\max\{\epsilon_j^1, \epsilon_j^2\}$ and $1.5 \times \left(\max\{\epsilon_j^1, \epsilon_j^2\}\right)$, see Line 26 in Algorithm 3. Note besides that the minimum number of divisions is 2.

C. MAIN POPULATION

We use GDE-MOEA to evolve the main population, but using individuals from the ϵ-Population to set the ϵ value for each objective function. Our proposal only affects the selection process based on ϵ-dominance as follows. Before choosing the ϵ-individual that we will use in the selection operator

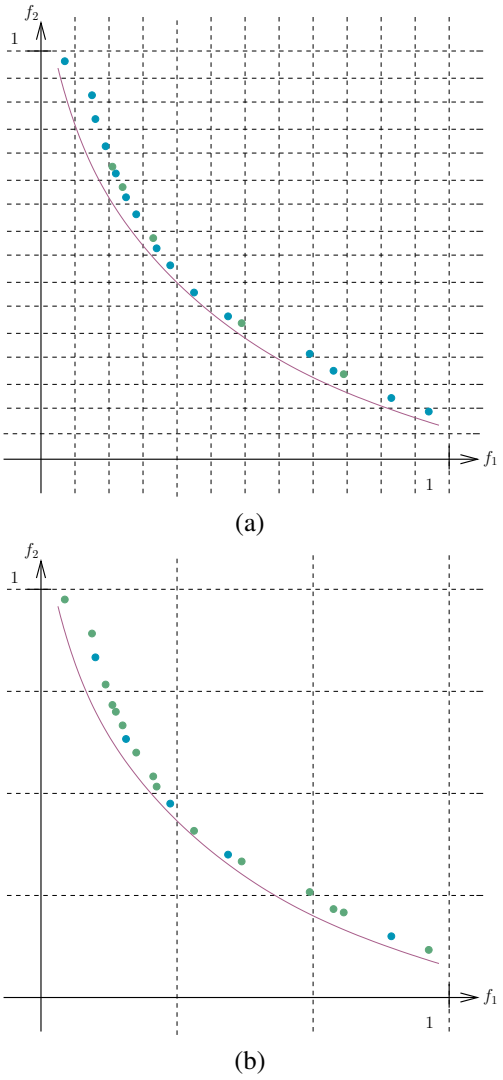

FIGURE 2: **ϵ-individuals**. Suppose that we want to select ten individuals. (a) If we use $\boldsymbol{\epsilon}_1 = [12, 16]$, we select fifteen individuals from the main population (blue points). Therefore, the fitness of $\boldsymbol{\epsilon}_1$ is equal to -5. (b) If we use $\boldsymbol{\epsilon}_2 = [3, 4]$, we select five individuals from the main population (blue points). Therefore, the fitness of $\boldsymbol{\epsilon}_2$ is equal to -5.5.

of GDE-MOEA, we perform one iteration of the $(\mu + 1)$-ES for the ϵ-population, see Algorithm 3. After that, we choose the best-known ϵ-individual and use it to select the individuals that will take part in the following generation. Let us assume that it is desirable to select N individuals. If the best ϵ-individual allows us to select M individuals with $M > N$, we randomly select N individuals from those M individuals. If the best ϵ-individual allows us to select M individuals with $M < N$, we randomly select the remaining $N - M$ individuals from the current set of nondominated individuals that have not been selected. The new MOEA is called "CoGDE-MOEA: Co-evolutionary Generational Distance and ϵ-dominance-based Multi-Objective Evolutionary Algorithm", see Algorithm 4.

Algorithm 3: Evolve ϵ-Population

Input : $\boldsymbol{E}$ (current ϵ-Population), $\boldsymbol{P}$ (current main population and N (number of individuals that we want to select).
Output: $\boldsymbol{E}$ (the new ϵ-Population).

/*Calculate the fitness of each ϵ-individual */
foreach $\boldsymbol{\epsilon} \in \boldsymbol{E}$ **do**
  /*Remember that this fitness depends on the current main population $\boldsymbol{P}$ */
  $x_j \leftarrow$ number of individuals that we will select from $\boldsymbol{P}$, if we use $\boldsymbol{\epsilon}$;
  if $x_j \geq N$ **then**
    $y_j \leftarrow 0.0$;
  else
    $y_j \leftarrow 0.5$;
  $fitness(\boldsymbol{\epsilon}^j) \leftarrow -1 * (|x_j - N| + y_j)$;
Select the best ϵ-individual ($\boldsymbol{\epsilon}^1$);
Select a random ϵ-individual ($\boldsymbol{\epsilon}^2$);
/*Apply the crossover operator */
$\boldsymbol{\epsilon} = 0.5 * (\boldsymbol{\epsilon}^1 + \boldsymbol{\epsilon}^2)$;
/*Apply the mutation operator */
if $\lfloor fitness(\boldsymbol{\epsilon}^1) \rfloor = fitness(\boldsymbol{\epsilon}^1)$ *and* $\lfloor fitness(\boldsymbol{\epsilon}^2) \rfloor = fitness(\boldsymbol{\epsilon}^2)$ **then**
  foreach $\epsilon_j \in \boldsymbol{\epsilon}$ **do**
    if $rand(0,1) < 0.5$ **then**
      $\epsilon_j \leftarrow rand\left(0.5 * \min(\epsilon_j^1, \epsilon_j^2), \min(\epsilon_j^1, \epsilon_j^2)\right)$;
      if $\epsilon_j < 2$ **then**
        $\epsilon_j \leftarrow 2$;
else
  if $\lfloor fitness(\boldsymbol{\epsilon}^1) \rfloor != fitness(\boldsymbol{\epsilon}^1)$ *and* $\lfloor fitness(\boldsymbol{\epsilon}^2) \rfloor != fitness(\boldsymbol{\epsilon}^2)$ **then**
    foreach $\epsilon_j \in \boldsymbol{\epsilon}$ **do**
      if $rand(0,1) < 0.5$ **then**
        $\epsilon_j \leftarrow rand\left(\max(\epsilon_j^1, \epsilon_j^2), 1.5 * \max(\epsilon_j^1, \epsilon_j^2)\right)$;
Select the worst ϵ-individual ($\boldsymbol{\epsilon}^w$);
if $\boldsymbol{\epsilon}$ *is better than* $\boldsymbol{\epsilon}^w$ **then**
  $\boldsymbol{\epsilon}$ replaces $\boldsymbol{\epsilon}^w$;

Algorithm 4: CoGDE-MOEA

Input : N (size of the main population) and p (size of ϵ-population: percentage regarding the main population).
Output: $\boldsymbol{P}$ (final main population).

Create the initial main population of size N called $\boldsymbol{P}$;
Create the initial ϵ-population of size $p \cdot N$ called $\boldsymbol{E}$;
while $numGeneration < totalGenerations$ **do**
  Create N new individuals using the operators of NSGA-II (crossover and mutation) and put them in $\boldsymbol{O}$;
  /*Select the best N individuals */
  Obtain the nondominated individuals in $\boldsymbol{P} \cup \boldsymbol{O}$ and put them in $\boldsymbol{S}$;
  Obtain the dominated individuals in $\boldsymbol{P} \cup \boldsymbol{O}$ and put them in $\boldsymbol{B}$;
  if $|\boldsymbol{S}| < N$ **then**
    Calculate the Euclidean distance d_i from each individual $\mathbf{x}^i \in \boldsymbol{B}$ to its nearest neighbor in $\boldsymbol{S}$ ($\boldsymbol{S}$ is the reference set) and we also save its closest nondominated neighbor;
    Sort $\boldsymbol{B}$ with respect to d_i (ascending order);
    $\boldsymbol{S'} \leftarrow \emptyset, allowedNeighbors \leftarrow 0, i \leftarrow 0$;
    while $|\boldsymbol{S'}| < N - |\boldsymbol{S}|$ **do**
      $countNeighbors \leftarrow 0$;
      foreach $\mathbf{s} \in \boldsymbol{S'}$ **do**
        if $\mathbf{s}.neighbor = \boldsymbol{B}.\mathbf{x}^i.neighbor$ **then**
          $countNeighbors \leftarrow countNeighbors + 1$;
      if $countNeighbors \leq allowedNeighbors$ **then**
        $\boldsymbol{S'} \leftarrow \boldsymbol{S'} \cup \boldsymbol{B}.\mathbf{x}^i, \boldsymbol{B} \leftarrow \boldsymbol{B} - \boldsymbol{B}.\mathbf{x}^i$;
      else
        $i \leftarrow i + 1$;
      if $i = |\boldsymbol{B}|$ **then**
        $i \leftarrow 0, allowedNeighbors \leftarrow allowedNeighbors + 1$;
    $\boldsymbol{P} \leftarrow \boldsymbol{S} \cup \boldsymbol{S'}$;
  if $|\boldsymbol{S}| > N$ **then**
    Evolve the ϵ-population, see Algorithm 3;
    Use the best ϵ-individual to calculate the identification vector $\mathbf{b}$ of each individual in $\boldsymbol{S}$;
    $\boldsymbol{P} \leftarrow \emptyset, \boldsymbol{S'} \leftarrow \emptyset$;
    foreach $\mathbf{x}^i \in \boldsymbol{S}$ **do**
      $flag \leftarrow 0$;
      foreach $\mathbf{y}^i \in \boldsymbol{P}$ **do**
        if $\mathbf{y}^i.\mathbf{b} = \mathbf{x}^i.\mathbf{b}$ **then**
          if $\mathbf{x}^i$ *is nearest to* $\mathbf{b}$ *than* $\mathbf{y}^i$ **then**
            $\mathbf{x}^i$ replaces $\mathbf{y}^i$;
            $\boldsymbol{S'} \leftarrow \boldsymbol{S'} \cup \mathbf{y}^i$;
            $flag \leftarrow 1$;
      if $flag = 0$ **then**
        $\boldsymbol{P} \leftarrow \boldsymbol{P} \cup \mathbf{x}^i$;
    if $|\boldsymbol{P}| > N$ **then**
      Delete $|\boldsymbol{P}| - N$ random individuals from $\boldsymbol{P}$;
    if $|\boldsymbol{P}| < N$ **then**
      Select $N - |\boldsymbol{P}|$ random individuals from $\boldsymbol{S'}$ and append them to $\boldsymbol{P}$;
  $numGeneration \leftarrow numGeneration + 1$;

VI. EXPERIMENTAL RESULTS

In our experimental study, we adopted seven problems from the DTLZ [36] test suite and seven problems from the Walking-Fish Group (WFG) test suite [39]. The adopted MOPs were employed up to six objective functions. For the DTLZ test problems, we adopted $k = 5$ for DTLZ1, DTLZ3, and DTLZ6, and $k = 10$ for the remaining DTLZ test problems. For the WFG test problems, we use $k\_factor = 2$ and $l\_factor = 10$. For each test problem, we performed 30 independent runs. All MOEAs adopted the genetic operators of NSGA-II using the parameters suggested by its authors, i.e., $p_c = 0.9$ (crossover probability), $p_m = 1/n$ (mutation probability), where n is the number of decision variables. For crossover and mutation operators, we adopted $\eta_c = 15$ and $\eta_m = 20$, respectively. We performed a maximum of 50,000 fitness function evaluations (we used a population size of 100 individuals and we iterated for 500 generations).

To assess the performance of the adopted MOEAs, we employed the following indicators:

a: ***Hypervolume indicator*** (I_H)

It was proposed by Zitzler et al. [40], and it is defined as the size of the space covered by the Pareto optimal solutions. This indicator is used to assess convergence towards the $\mathcal{PF}$, as well as maximum spread of the solutions obtained. I_H is mathematically stated as:

$$I_{Hv}(A) = \mathcal{L}\left(\bigcup_{\mathbf{z}\in A}\{\mathbf{x}|\mathbf{z} \prec \mathbf{x} \prec \mathbf{y}_{ref}\}\right) \tag{5}$$

where $\mathcal{L}$ denotes the Lebesgue measure and $\mathbf{y}_{ref} \in \mathbb{R}^M$ denotes a reference vector being dominated by all solutions in A. To calculate I_H, we normalized the approximations of the $\mathcal{PF}$, generated by the MOEAs and we used $y_{ref} = (y_1, \cdots, y_k)$ such that $y_i = 1.1$ as our reference point. We perform the normalization considering all approximations generated by the different MOEAs adopted in each comparison (i.e., we put, in one set, all the nondominated solutions found and from this set we calculated the maximum and minimum for each objective function).

b: ***Two Set Coverage*** (I_{SC})

We used this indicator to assess the convergence of the MOEAs adopted in this paper. It was proposed by Zitzler et al. [41], and it is a binary Pareto compliant indicator. I_{SC} is defined as follows: Let $\boldsymbol{A}, \boldsymbol{B}$ be two approximations of $\mathcal{PF}$, we have that:

$$I_{SC}(\boldsymbol{A}, \boldsymbol{B}) = \frac{|\mathbf{b} \in \boldsymbol{B} \text{ such that } \exists \mathbf{a} \in \boldsymbol{A} \text{ with } \mathbf{a} \prec \mathbf{b}|}{|\boldsymbol{B}|}$$

If all points in $\boldsymbol{A}$ dominate or are equal to all points in $\boldsymbol{B}$, then by definition $I_{SC} = 1$. $I_{SC} = 0$ implies that no element in $\boldsymbol{B}$ is dominated by any element of $\boldsymbol{A}$. In general, both $I_{SC}(\boldsymbol{A}, \boldsymbol{B})$ and $I_{SC}(\boldsymbol{B}, \boldsymbol{A})$ have to be considered.

c: ***Spacing*** (I_S)

We used this indicator to measure the spread of solutions in the approximate $\mathcal{PF}$. It was proposed by Schott [42] and it is defined as follows:

$$I_S(\boldsymbol{A}) = \sqrt{\frac{1}{|\boldsymbol{A}| - 1}\sum_{i=1}^{|\boldsymbol{A}|}\left(\overline{d} - d_i\right)^2}$$

where: $d_i = \min_{j,j\neq i}\sum_k |f_k^i - f_k^j|$ and $\overline{d} = \frac{1}{|\boldsymbol{A}|}\sum_{i=1}^{|\boldsymbol{A}|} d_i$, k is the number of objective functions, $i, j = 1, \cdots, |\boldsymbol{A}|$. When $I_S = 0$, all the solutions in $\boldsymbol{A}$ are uniformly spread.

A. STUDY: SIZE OF THE ϵ-POPULATION

Before comparing our CoGDE-MOEA against the other MOEAs, we show a study of the behavior of CoGDE-MOEA when using different sizes for the ϵ-population. Figure 3 shows two histograms. The first histogram corresponds to the results obtained in the DTLZ test problems with three, four, five, and six objective functions. The second histogram corresponds to the WFG test problems with the same number of objective functions as DTLZ test problems. The value plotted corresponds to the number of times that each version of CoGDE-MOEA obtains specific place (first, second, ..., ninth).

From these plots, we can observe that all versions of the algorithm are competitive with each other. However, in the DTLZ test problems, it is clear that the versions in the extremes (10%, 20%, 80%, and 90%) are better and quite competitive among them. In the WFG test problems, it is clear that the version that uses 10% is the best. This characteristic caught our attention, and then we decided to compare in more detail the versions that use an ϵ-population size equal to 10% and 90% of the main population.

Table 2 summarizes the results obtained by performing 30 independent runs for each of the 56 adopted test problems. For the comparison, we used the hypervolume indicator and we applied a statistical analysis using Wilcoxon's rank sum to know how many times one algorithm outperformed another one (the null hypothesis "medians are equal" can be rejected at the 5% level) and how many times they had a similar behavior (the null hypothesis cannot be rejected at the 5% level).

It is also possible to observe that the version that uses an ϵ-population size equal to 90% of the main population obtained better results in problems with few objective functions (three and four). On the other hand, the version with an ϵ-population size equal to 10% of the main population obtained better results in problems with more objective functions (five and six). In fact, the use of large ϵ-populations promotes the exploration, in a better way, of the search space corresponding to the problem of finding the optimal ϵ value. It is worth noticing that in problems with few objective functions, the number of iterations used was sufficient to approximate the optimal ϵ value. However, in problems with more objective functions, the problem of approximating the optimal ϵ value is much more difficult, and the ϵ-population cannot converge to a good value in a fast way.

Therefore, we hypothesized that the use of small ϵ-population sizes in these test problems would not be able to explore the whole search space and it is very likely that the ϵ-population will converge to a local optimum. To validate our hypothesis, we plotted the convergence of the ϵ-population. Figures 4 and 5 show the convergence plots for DTLZ4, DTLZ7, WFG2, and WFG6 with three and six objective functions and we can see that in most cases the version that uses a smaller ϵ-population achieves convergence to an ϵ-individual with the best fitness (0). Conversely, when we use a larger population, we cannot converge to an ϵ-individual with the best fitness in problems with six objective functions. Moreover, in these plots, we can see that the convergence of smaller ϵ-populations is faster than the one of larger ϵ-populations.

For the above reasons and in order to perform a fair comparison, we use an ϵ-population size equal to ten percent

of the main population size for all the comparisons reported in the following.

TABLE 2: Results obtained in the DTLZ and WFG test problems with up to six objective functions. We group the test problems by the number of objective functions (k). We compare two versions of CoGDE-MOEA: the first one uses an ϵ-population equal to 10% and the other one uses an ϵ-population equal to 90%. For the comparison, we used the hypervolume indicator and Wilcoxon's rank sum. The format is as follows: number of times that the algorithm won / number of times that the algorithm tied / number of times that the algorithm lost.

k	CoGDE-MOEA (10%)	CoGDE-MOEA (90%)
3	0 / 5 / 2	**2 / 5 / 0**
4	0 / 6 / 1	**1 / 6 / 0**
5	**1 / 6 / 0**	0 / 6 / 1
6	0 / 7 / 0	0 / 7 / 0

DTLZ test problems

k	CoGDE-MOEA (10%)	CoGDE-MOEA (90%)
3	0 / 6 / 1	**1 / 6 / 0**
4	0 / 6 / 1	**1 / 6 / 0**
5	**3 / 4 / 0**	0 / 4 / 3
6	**2 / 5 / 0**	0 / 5 / 2

WFG test problems

B. COMPARISON OF MOEAS BASED ON ϵ-DOMINANCE

In this section, we compare the proposed CoGDE-MOEA concerning a version of GDE-MOEA that uses the ϵ values found by EVOCA (we call this version Evoca GDE-MOEA) and with respect to the original GDE-MOEA. EVOCA was employed to find good-performing ϵ values for problems with three, four, five, and six objective functions considering both the DTLZ and the WFG test problems. See Table 3.

TABLE 3: ϵ values found by EVOCA considering the DTLZ and WFG test problems. k denotes the number of objective functions.

k	ϵ values
3	$[0.04, 0.012, 0.045]$
4	$[0.012, 0.5, 0.014, 0.018]$
5	$[0.035, 0.5, 0.142, 0.09, 0.1]$
6	$[0.026, 0.033, 0.052, 0.023, 0.01, 0.02]$

Tables 4(a) and 4(b) summarize the results obtained over 30 independent runs for each of the 56 adopted test problems. We used the hypervolume indicator and Wilcoxon's rank sum in the same way that in the previous experimental study. In Table 4(a), we can see that the proposed CoGDE-MOEA was better than evoca GDE-MOEA in fifteen DTLZ test problems.

Evoca GDE-MOEA and CoGDE-MOEA showed similar behavior in eight problems, and Evoca GDE-MOEA outperforms CoGDE-MOEA in five problems. Also, we can observe that our CoGDE-MOEA is better than the original GDE-MOEA in twelve DTLZ test problems. In eight DTLZ test problems both MOEAs had similar behavior. The original GDE-MOEA is better than our CoGDE-MOEA in eight DTLZ test problems. Regarding the WFG test problems (see Table 4(b)), CoGDE-MOEA is better than Evoca GDE-MOEA in twenty-four problems, and they had similar behavior in four problems. Concerning the original GDE-MOEA, CoGDE-MOEA outperforms it in seventeen problems, CoGDE-MOEA was outperformed by it in three problems, and both MOEAs had similar behavior in eight problems.

From all the above results, we can corroborate that the ϵ values depend on several factors, and then, it is not viable to employ a specialized technique to tune them considering a set of different problems. In this experimental study, we grouped the MOPs by the number of objective functions, and the results are not good. In conclusion, it is necessary to use a specialized technique per problem, and this has a much higher computational cost. Moreover, we can see that our co-evolutionary scheme to set the ϵ values was able to outperform the scheme proposed in [5] in most of the test problems. An essential difference between both schemes is that the co-evolutionary scheme proposed here assumes that the optimal ϵ values can be different per objective function. Therefore, we can say that a different ϵ value for each objective should be considered when ϵ-dominance is used to solve MOPs.

In Table 5, we can see that GDE-MOEA required at most two seconds to solve the adopted MOPs while CoGDE-MOEA required up to seven seconds. Although GDE-MOEA seems to be better regarding computational time, we must consider that GDE-MOEA uses one ϵ value for all objectives while CoGDE-MOEA finds a suitable ϵ value for each objective function. This is reflected in the quality of the $\mathcal{PF}$ approximations achieved by our proposed CoGDE-MOEA.

C. COGDE-MOEA VS MOEAS NOT BASED ON ϵ-DOMINANCE

In this section, we compare CoGDE-MOEA with respect to two well-known MOEAs: MOEA/D and SMS-EMOA. We chose MOEA/D because it is a viable alternative to deal with many-objective optimization problems, e.g., its computational cost is very low. MOEA/D was proposed by Zhang et al. in [6], and it decomposes the MOP into N scalar optimization subproblems which are then simultaneously solved using an evolutionary algorithm. For our experiments, we use PBI (Penalty Boundary Intersection) to decompose the MOP. To generate the convex weights, we used the technique proposed in [43] and after that, we applied clustering (k-means) to obtain a specific number of weights.[4] In our experiments, we used a neighborhood size equal to 20. We also compared results against SMS-EMOA [7] because it is the most popular hypervolume-based MOEA. In this study, we used a version of SMS-EMOA that approximates the contribution to the

[4]Note that we cannot directly use the convex weights generated by the technique proposed in [43] because the number of weights grows very quickly as we increase the number of objective functions.

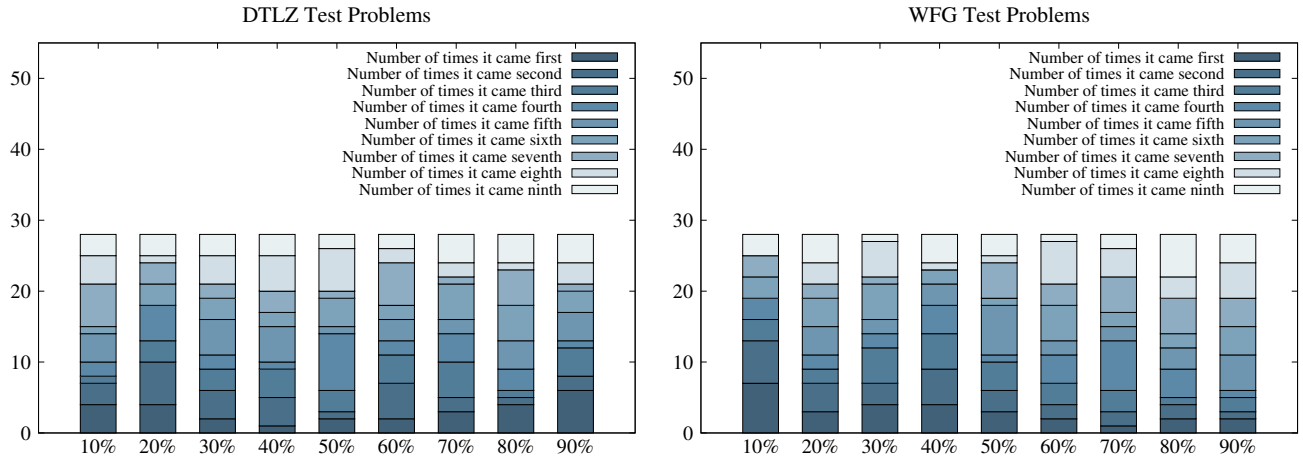

FIGURE 3: **Results obtained in the DTLZ and WFG test problems with three, four, five and six objective functions**. Histograms show the number of times that each version of CoGDE-MOEA (depending on the size of ϵ-population that it uses) came in certain place (first, second,..., ninth). We compare CoGDE-MOEA using an ϵ-population size equal to 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80% and 90% of the main population.

hypervolume using the technique proposed by Bader et al. in [8] adopting 10^4 as the number of samples. We called this version "hypeSMS-EMOA". It is worth mentioning that the original SMS-EMOA was not employed because it can require up to five hours to solve a problem with five objectives, and to solve a problem with six objective functions is impractical.

Tables 4(c) and 4(d) show the results regarding I_H. From Table 4(c), we can see that in nineteen DTLZ test problems, CoGDE-MOEA was better than MOEA/D. In seven cases, both algorithms had similar behavior. MOEA/D was better than CoGDE-MOEA in only two test problems. For the WFG test problems (see Table 4d), we can observe that only in one problem both algorithms obtained similar behavior. However, in three problems MOEA/D was better than CoGDE-MOEA, and in twenty-four test problems, CoGDE-MOEA was better than MOEA/D. From these results, we can say that CoGDE-MOEA outperformed MOEA/D.

On the other hand, we can see that hypeSMS-EMOA outperformed CoGDE-MOEA in twenty-three DTLZ test problems, they obtained similar behavior in four problems, and only in one case, CoGDE-MOEA overcame hypeSMS-EMOA. However, concerning the WFG test problems, the proposed CoGDE-MOEA outperformed hypeSMS-EMOA in nine test problems, they have similar behavior in eight cases, and in eleven problems hypeSMS-EMOA was better than CoGDE-MOEA. Therefore, we can say that in the DTLZ test problems hypeSMS-EMOA outperformed our CoGDE-MOEA, but in the WFG test problems, our proposed approach was competitive. In Table 5 we can observe the maximum time required by the three MOEAs to solve an MOP with a specific number of objective functions. MOEA/D was the fastest algorithm, followed by CoGDE-MOEA. Clearly, hypeSMS-EMOA was the slowest algorithm. If we consider the worst case, MOEA/D was approximately five times faster than CoGDE-MOEA and CoGDE-MOEA was approximately eleven times faster than hypeSMS-EMOA.

To corroborate the results obtained by I_H, in Table 6, we present the results obtained by the algorithms using I_{SC} and I_S[5]. We applied Wilcoxon's rank sum in the same way as in the previous sections but now using I_S instead of I_H. In Table 6(a), we can observe that only in two cases (WFG1(3) and WFG5(5)) CoGDE-MOEA was better than MOEA/D. Moreover, CoGDE-MOEA was also better regarding I_S in these two problems. There are five problems (WFG2(3), WFG4(3), WFG2(4), WFG2(5), and WFG2(6)) in which CoGDE-MOEA covered almost one hundred percent of the solutions found by MOEA/D, and MOEA/D covered less than one percent of the solutions found by CoGDE-MOEA. There was no problem where MOEA/D was better than CoGDE-MOEA. We can also see that CoGDE-MOEA achieved a better distribution than MOEA/D in nineteen WFG test problems, while MOEA/D achieved a better distribution than CoGDE-MOEA in four WFG test problems. From these results, we corroborate that CoGDE-MOEA outperformed MOEA/D.

Concerning hypeSMS-EMOA, in two WFG test problems (WFG5(4) and WFG5(5)), CoGDE-MOEA was better than hypeSMS-EMOA and only in one problem (WFG1(3)) occurred the opposite, see Table 6(b). Regarding I_S, we can see that in ten WFG test problems, CoGDE-MOEA achieved a better distribution than hypeSMS-EMOA and in thirteen WFG test problems, hypeSMS-EMOA achieved a better distribution.

To summarize this section, we can say that CoGDE-

[5]Note that we only present results for the WFG test problems, in the case of the DTLZ test problems, the results were similar.

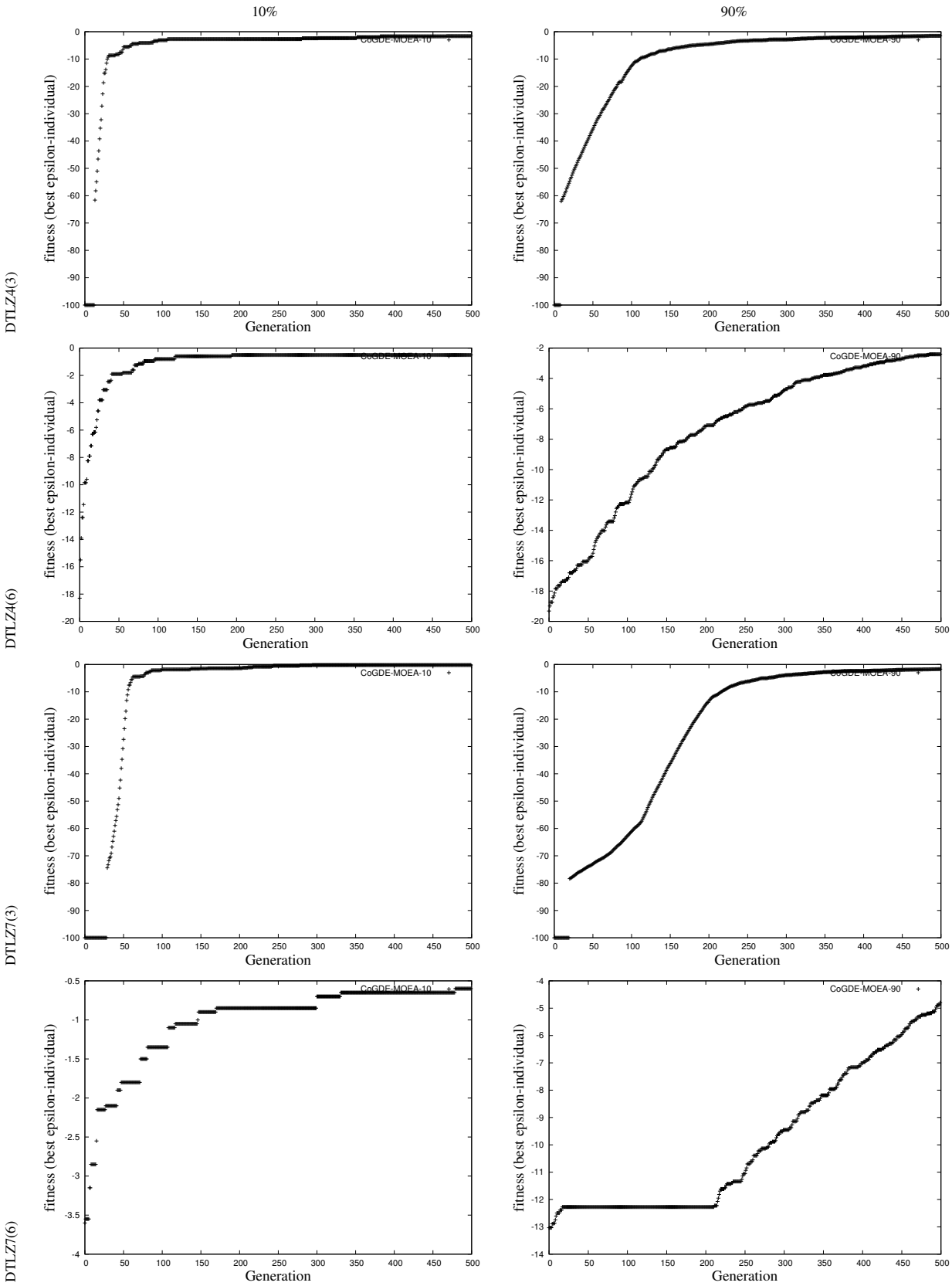

FIGURE 4: **Convergence graph (ϵ-population) obtained by CoGDE-MOEA**. We use an ϵ-population sizes equal to 10 and 90 percent of the main population, in the median (concerning the hypervolume indicator) of its independent runs for the test problems DTLZ4 and DTLZ7 with three and six objective functions.

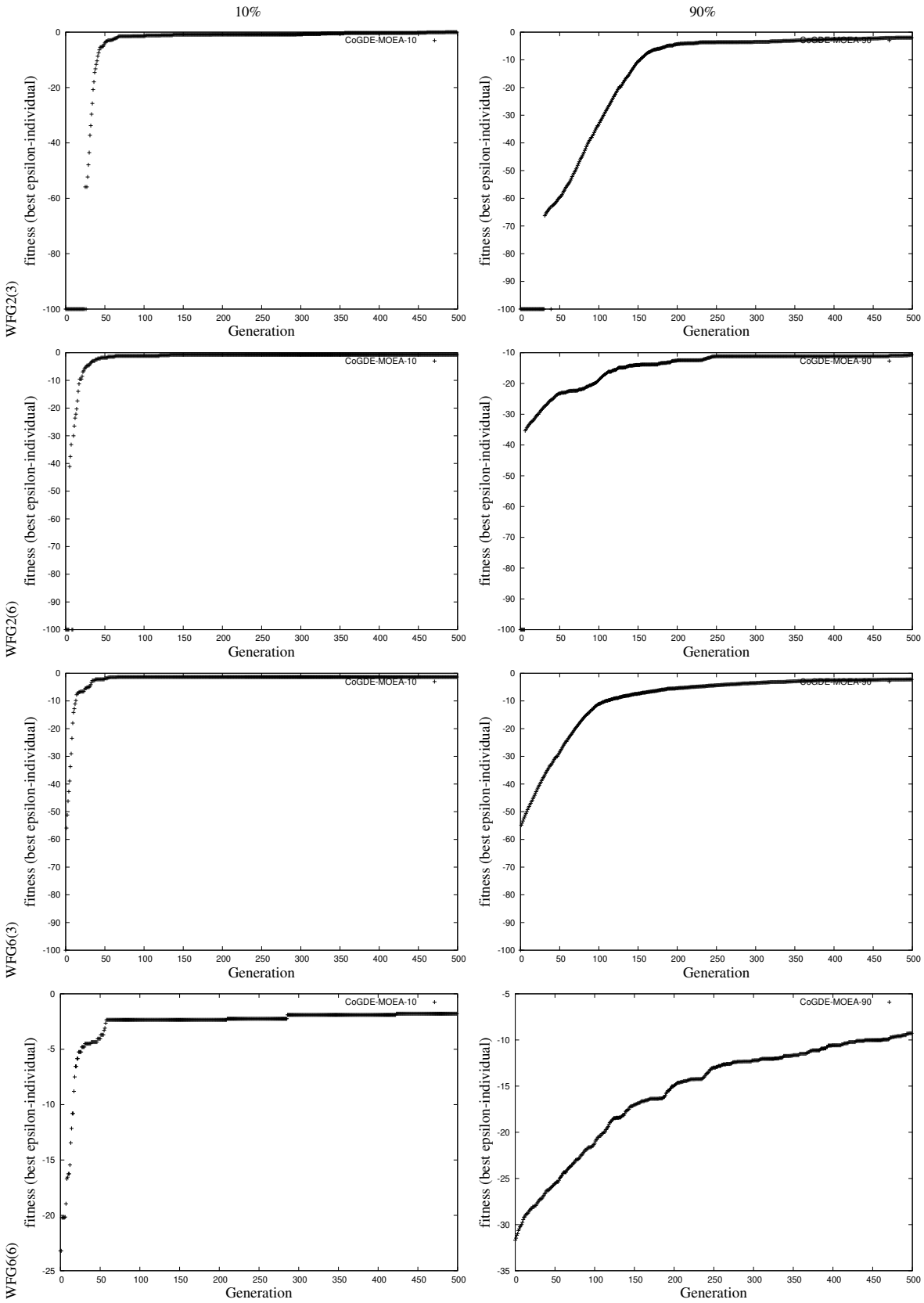

FIGURE 5: **Convergence graph (ϵ-population) obtained by CoGDE-MOEA**. We use an ϵ-population sizes equal to 10 and 90 percent of the main population, in the median (concerning the hypervolume indicator) of its independent runs for the test problems WFG2 y WFG6 with three and six objective functions.

TABLE 4: Results obtained by Evoca GDE-MOEA, the original GDE-MOEA, MOEA/D, hype SMS-EMOA and CoGDE-MOEA in the DTLZ and WFG test problems. We group the test problems by the number of objective functions (k). For the comparison, we used the hypervolume indicator and Wilcoxon's rank sum. The format is as follows: number of times that the algorithm won / number of times that the algorithm tied / number of times that the algorithm lost.

k	Co GDE-MOEA	evoca GDE-MOEA
3	2 / 2 / 3	**3 / 2 / 2**
4	**3 / 4 / 0**	0 / 4 / 3
5	**5 / 2 / 0**	0 / 2 / 5
6	**5 / 0 / 2**	2 / 0 / 5

DTLZ test problems

k	Co GDE-MOEA	original GDE-MOEA
3	1 / 1 / 5	**5 / 1 / 1**
4	2 / 2 / 3	**3 / 2 / 2**
5	**4 / 3 / 0**	0 / 3 / 4
6	**5 / 2 / 0**	0 / 2 / 5

DTLZ test problems

(a)

k	Co GDE-MOEA	evoca GDE-MOEA
3	**6 / 1 / 0**	0 / 1 / 6
4	**6 / 1 / 0**	0 / 1 / 6
5	**6 / 1 / 0**	0 / 1 / 6
6	**6 / 1 / 0**	0 / 1 / 6

WFG test problems

k	Co GDE-MOEA	original GDE-MOEA
3	2 / 2 / 3	**3 / 2 / 2**
4	**3 / 4 / 0**	0 / 4 / 3
5	**5 / 2 / 0**	0 / 2 / 5
6	**7 / 0 / 0**	0 / 0 / 7

WFG test problems

(b)

k	Co GDE-MOEA	MOEA / D
3	**4 / 3 / 0**	0 / 3 / 4
4	**4 / 2 / 1**	1 / 2 / 4
5	**5 / 2 / 0**	0 / 2 / 5
6	**6 / 0 / 1**	1 / 0 / 6

DTLZ test problems

k	Co GDE-MOEA	hype SMS-EMOA
3	0 / 2 / 5	**5 / 2 / 0**
4	1 / 1 / 5	**5 / 1 / 1**
5	0 / 1 / 6	**6 / 1 / 0**
6	0 / 0 / 7	**7 / 0 / 0**

DTLZ test problems

(c)

k	Co GDE-MOEA	MOEA / D
3	**6 / 0 / 1**	1 / 0 / 6
4	**6 / 0 / 1**	1 / 0 / 6
5	**6 / 0 / 1**	1 / 0 / 6
6	**6 / 1 / 0**	0 / 1 / 6

WFG test problems

k	Co GDE-MOEA	hype SMS-EMOA
3	**3 / 3 / 1**	1 / 3 / 3
4	3 / 1 / 3	3 / 1 / 3
5	1 / 2 / 4	**4 / 2 / 1**
6	2 / 2 / 3	**3 / 2 / 2**

WFG test problems

(d)

MOEA is an excellent alternative to solve MOPs with low (three objective functions) and high (four, five, and six objective functions) dimensionality. As we showed, the proposed CoGDE-MOEA was better than MOEA/D, and it was quite competitive with respect to hypeSMS-EMOA but at a much lower computational cost (CoGDE-MOEA is eleven times faster than hypeSMS-EMOA).

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have studied MOEAs based on ϵ-dominance, finding that in most cases, they select individuals in the same way as ϵ-MOEA. The main disadvantage of this selection is that setting the ϵ values is a hard task: it is necessary to know the true Pareto front ($\mathcal{PF}$) and to state the number of nondominated solutions required. Since there are several MOPs of which we do not know their $\mathcal{PF}$s, the first point is infeasible.

There are works in the literature that address this problem, and all of them agree that the optimal value of ϵ depends on the features of the MOP, e.g., geometrical characteristics of the true $\mathcal{PF}$, correlation between objective functions, number of objective functions, and size of the search space. For this reason, we adopted the following hypothesis: "The optimal value of ϵ depends on the MOP that we want to solve. Additionally, it is possible to have different ϵ values for each objective function".

The previous related works present some disadvantages. The most important are the following: (i) none of the previous proposals conducted a preliminary study to validate that the ϵ values indeed depend on the MOP that we want to solve, (ii) some of these assume that a single value of ϵ is sufficient for all the objective functions of the MOP, and (iii) take into consideration the spread of the Pareto front but not its geometrical characteristics to set the ϵ value.

The main contributions of this work are two. First, we validated the above hypothesis using a specialized algorithm to set parameters, called "Evolutionary Calibrator (EVOCA)". For this task, we considered test problems with different characteristics. The goal of EVOCA was to find the ϵ values that maximized the hypervolume indicator of the approximate $\mathcal{PF}$s found by a recent MOEA based on ϵ-dominance called "Generational Distance and ϵ-dominance-based Multi-Objective Evolutionary Algorithm (CoGDE-MOEA)". Since EVOCA found different values of ϵ for each objective function and there is no relation between the ϵ values for each problem, we can assume that our hypothesis is correct.

Second, we proposed a new co-evolutionary scheme to set the ϵ values in any MOEA based on ϵ-dominance. To validate our co-evolutionary scheme, we incorporated it into the GDE-MOEA. The resulting approach was called "Co-evolutionary Generational Distance and ϵ-dominance - based Multi-Objective Evolutionary Algorithm (CoGDE-MOEA)". CoGDE-MOEA coevolves a population of ϵ values which let the algorithm look for the best setting of such values along the search. We proposed specialized operators for this special ϵ's population. Therefore, our co-evolutionary scheme is implicitly taking into account all the points mentioned before (geometrical features, number of objective functions, etc.). Our results indicate that our CoGDE-MOEA was able to outperform the original GDE-MOEA as well as MOEA/D. Our proposed approach also showed to be competitive concerning a version of SMS-EMOA that uses a fitness assignment mechanism based on the approximation of the hypervolume (hypeSMS-EMOA) but at a much lower computational cost (CoGDE-MOEA is eleven times faster than hypeSMS-EMOA).

As part of our future work, we want to incorporate our co-evolutionary scheme into other MOEAs based on ϵ-dominance with the aim of measuring the impact that our scheme has on their performance. We also want to study other ways to evolve the ϵ-population with the aim of improving our $(\mu + 1)$-ES to find the optimal ϵ values.

TABLE 5: Results obtained in the DTLZ and WFG test problems with up to six objective functions. We compare GDE-MOEA, CoGDE-MOEA, MOEA/D and hypeSMS-EMOA concerning the running time required by each MOEA to obtain the approximation of the Pareto optimal set. We group the test problems by the number of objective functions, and we show the maximum time required for each group. The results are in seconds. We show average values over 30 independent runs. The values in parentheses correspond to the standard deviations.

k	GDE-MOEA time	Co GDE-MOEA time	MOEA/D time	hype SMS-EMOA time	GDE-MOEA time	Co GDE-MOEA time	MOEA/D time	hype SMS-EMOA time
3	0.73(0.44)	1.98(0.47)	**0.58(0.00)**	19.38(0.60)	1.33(0.47)	6.14(0.96)	**1.31(0.21)**	28.46(0.39)
4	0.86(0.34)	2.46(0.62)	**0.65(0.00)**	27.02(1.11)	1.30(0.46)	4.67(1.06)	**1.60(0.01)**	37.61(0.54)
5	0.70(0.46)	3.52(0.39)	**0.69(0.00)**	34.95(1.85)	1.53(0.50)	5.04(0.50)	**1.94(0.05)**	46.20(0.44)
6	0.76(0.43)	4.08(0.72)	**0.74(0.00)**	42.46(3.11)	1.93(0.44)	4.84(0.84)	**1.82(0.00)**	54.64(0.50)
	DTLZ test problems				WFG test problems			

TABLE 6: Results in the WFG test problems using I_{SC} and I_S. In this case, $\boldsymbol{A}$ is the set composed by all solutions found by CoGDE-MOEA considering all 30 independent runs. In (a), $\boldsymbol{B}$ is the set composed by all solutions found by MOEA/D considering all 30 independent runs. In (b), $\boldsymbol{B}$ is the set composed by all solutions found by hypeSMS-EMOA considering all 30 independent runs. In the case of I_S, we applied Wilcoxon's rank sum, and the column indicates which MOEA won. When we put "-", it means that both MOEAs had similar behavior.

f	$I_{SC}(\mathcal{A},\mathcal{B})$	$I_{SC}(\mathcal{B},\mathcal{A})$	I_S
WFG1 (3)	**0.0050**	0.0000	$\mathcal{A}$
WFG2 (3)	**0.9740**	0.0403	-
WFG3 (3)	**0.5990**	0.0153	$\mathcal{A}$
WFG4 (3)	**0.9787**	0.0020	$\mathcal{A}$
WFG5 (3)	**0.0690**	0.0047	$\mathcal{A}$
WFG6 (3)	**0.4127**	0.0333	$\mathcal{A}$
WFG7 (3)	**0.2087**	0.0450	$\mathcal{A}$
WFG1 (4)	0.0000	0.0000	$\mathcal{A}$
WFG2 (4)	**0.9993**	0.0010	-
WFG3 (4)	**0.3317**	0.0153	$\mathcal{A}$
WFG4 (4)	**0.6557**	0.0013	$\mathcal{A}$
WFG5 (4)	**0.0160**	0.0003	$\mathcal{A}$
WFG6 (4)	**0.3107**	0.0143	$\mathcal{A}$
WFG7 (4)	0.0053	**0.0067**	-
WFG1 (5)	0.0000	0.0000	-
WFG2 (5)	**0.9483**	0.0057	-
WFG3 (5)	**0.1053**	0.0180	$\mathcal{A}$
WFG4 (5)	**0.2950**	0.0057	$\mathcal{A}$
WFG5 (5)	**0.0060**	0.0000	$\mathcal{A}$
WFG6 (5)	**0.1857**	0.0217	$\mathcal{A}$
WFG7 (5)	0.0000	0.0000	$\mathcal{B}$
WFG1 (6)	0.0000	0.0000	$\mathcal{B}$
WFG2 (6)	**0.8730**	0.0130	$\mathcal{B}$
WFG3 (6)	**0.0330**	0.0010	$\mathcal{A}$
WFG4 (6)	**0.0970**	0.0047	$\mathcal{A}$
WFG5 (6)	0.0000	0.0000	$\mathcal{A}$
WFG6 (6)	**0.1880**	0.0040	$\mathcal{A}$
WFG7 (6)	0.0000	0.0000	$\mathcal{B}$

(a)

$I_{SC}(\mathcal{A},\mathcal{B})$	$I_{SC}(\mathcal{B},\mathcal{A})$	I_S
0.0000	**0.0007**	$\mathcal{A}$
0.1840	**0.6477**	$\mathcal{B}$
0.2490	0.0657	$\mathcal{A}$
0.5980	0.0670	$\mathcal{A}$
0.0243	0.0027	$\mathcal{A}$
0.0317	**0.2460**	$\mathcal{B}$
0.3220	0.0307	$\mathcal{A}$
0.0000	0.0000	$\mathcal{A}$
0.3290	**0.4150**	$\mathcal{B}$
0.0643	0.0567	$\mathcal{B}$
0.1023	0.0703	$\mathcal{A}$
0.0377	0.0000	-
0.0157	**0.1810**	$\mathcal{B}$
0.0187	0.0103	-
0.0000	0.0000	$\mathcal{A}$
0.1123	**0.3740**	$\mathcal{B}$
0.0237	**0.2657**	$\mathcal{B}$
0.0453	0.0387	$\mathcal{A}$
0.0357	0.0000	-
0.0213	**0.5290**	-
0.0010	**0.0023**	$\mathcal{B}$
0.0000	0.0000	-
0.1260	**0.2160**	$\mathcal{B}$
0.0370	**0.0940**	$\mathcal{B}$
0.0337	0.0300	$\mathcal{A}$
0.0137	0.0050	$\mathcal{B}$
0.0877	**0.1870**	$\mathcal{B}$
0.0000	0.0000	$\mathcal{B}$

(b)

ACKNOWLEDGMENT

The first author acknowledges support from PAPIIT project no. IA105918. Second author acknowledges support from Fondecyt project no. 11150787. Fifth author gratefully acknowledges support from CONACyT project no. 221551

REFERENCES

[1] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems. New York: Springer, second ed., September 2007. ISBN 978-0-387-33254-3.

[2] M. Farina and P. Amato, "On the optimal solution definition for many-criteria optimization problems," in Proceedings of the NAFIPS-FLINT International Conference'2002, (Piscataway, New Jersey), pp. 233–238, IEEE Service Center, June 2002.

[3] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of pareto-

optimal solutions," Evolutionary Computation, vol. 13, pp. 501–525, Winter 2005.

[4] M. C. Riff and E. Montero, "A new algorithm for reducing metaheuristic design effort," in IEEE Congress on Evolutionary Computation (CEC 2013), (Cancún, México), pp. 3283–3290, June 2013.

[5] A. Menchaca-Mendez and C. A. Coello Coello, "Gde-moea : A new moea based on the generational distance indicator and ϵ-dominance," in 2015 IEEE Congress on Evolutionary Computation (CEC'2015), (Sendai, Japan), pp. 947–955, IEEE Press, 25-28 May 2015. ISBN 978-1-4799-7492-4.

[6] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," IEEE Transactions on Evolutionary Computation, vol. 11, pp. 712–731, December 2007.

[7] M. Emmerich, N. Beume, and B. Naujoks, "An EMO algorithm using the hypervolume measure as selection criterion," in Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005 (C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, eds.), (Guanajuato, México), pp. 62–76, Springer. Lecture Notes in Computer Science Vol. 3410, March 2005.

[8] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," Evolutionary Computation, vol. 19, pp. 45–76, Spring 2011.

[9] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multi-objective optimization," Evolutionary Computation, vol. 10, pp. 263–282, Fall 2002.

[10] H. Xia, J. Zhuang, and D. Yu, "Multi-objective unsupervised feature selection algorithm utilizing redundancy measure and negative epsilon-dominance for fault diagnosis," Neurocomputing, vol. 146, pp. 113–124, December 25 2014.

[11] S. Bandyopadhyay, U. Maulik, and R. Chakraborty, "Incorporating ϵ-dominance in amosa: Application to multiobjective 0/1 knapsack problem and clustering gene expression data," Applied Soft Computing, vol. 13, no. 5, pp. 2405 – 2411, 2013.

[12] C. Angione, J. Costanza, G. Carapezza, P. LiÓ, and G. Nicosia, "Pareto epsilon-dominance and identifiable solutions for biocad modeling," in Proceedings of the 50th Annual Design Automation Conference, DAC '13, (New York, NY, USA), pp. 43:1–43:9, ACM, 2013.

[13] K. Xu, J. Zhou, Y. Zhang, and R. Gu, "Differential evolution based on ϵ-domination and orthogonal design method for power environmentally-friendly dispatch," Expert Syst. Appl., vol. 39, pp. 3956–3963, Mar. 2012.

[14] M. Li, L. Liu, and D. Lin, "A fast steady-state ϵ-dominance multi-objective evolutionary algorithm," Comput. Optim. Appl., vol. 48, pp. 109–138, Jan. 2011.

[15] S.-Y. Shin, I.-H. Lee, and B.-T. Zhang, "Microarray probe design using ϵ-multi-objective evolutionary algorithms with thermodynamic criteria," in Applications of Evolutionary Computing. EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC (F. R. et al., ed.), (Budapest, Hungary), pp. 184–195, Springer, Lecture Notes in Computer Science Vol. 3907, April 2006.

[16] M. Reyes Sierra and C. A. Coello Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance," in Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005 (C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, eds.), (Guanajuato, México), pp. 505–519, Springer. Lecture Notes in Computer Science Vol. 3410, March 2005.

[17] H. Sato, H. E. Aguirre, and K. Tanaka, "Controlling dominance area of solutions and its impact on the performance of moeas," in Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007 (S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, eds.), (Matshushima, Japan), pp. 5–20, Springer. Lecture Notes in Computer Science Vol. 4403, March 2007.

[18] H. Sato, H. E. Aguirre, and K. Tanaka, "Self-controlling dominance area of solutions in evolutionary many-objective optimization," in Simulated Evolution and Learning, 8th International Conference, SEAL 2010 (K. Deb, A. Bhattacharya, N. Chakraborti, P. Chakroborty, S. Das, J. Dutta, S. K. Gupta, A. Jain, V. Aggarwal, J. Branke, S. J. Louis, and K. C. Tan, eds.), (Kanpur, India), pp. 455–465, Springer. Lecture Notes in Computer Science Vol. 6457, December 1-4 2010.

[19] A. G. Hernández-DÍaz, L. V. Santana-Quintero, C. A. Coello Coello, and J. Molina, "Pareto-adaptive ϵ-dominance," Evolutionary Computation, vol. 15, pp. 493–517, Winter 2007.

[20] H. Aguirre and K. Tanaka, "Adaptive ϵ-ranking on mnk-landscapes," in 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM'2009), (Nashville, TN, USA), pp. 104–111, IEEE Press, March 30 - April 2 2009. ISBN 978-1-4244-2764-2.

[21] H. E. Aguirre and K. Tanaka, "Space partitioning with adaptive ϵ-ranking and substitute distance assignments: a comparative study on many-objective mnk-landscapes," in 2009 Genetic and Evolutionary Computation Conference (GECCO'2009), (Montreal, Canada), pp. 547–554, ACM Press, July 8–12 2009. ISBN 978-1-60558-325-9.

[22] H. Aguirre and K. Tanaka, "Many-objective optimization by space partitioning and adaptive ϵ-ranking on mnk-landscapes," in Evolutionary Multi-Criterion Optimization. 5th International Conference, EMO 2009 (M. Ehrgott, C. M. Fonseca, X. Gandibleux, J.-K. Hao, and M. Sevaux, eds.), pp. 407–422, Nantes, France: Springer. Lecture Notes in Computer Science Vol. 5467, April 2009.

[23] N. Kowatari, A. Oyama, H. E. Aguirre, and K. Tanaka, "A study on large population moea using adaptive ϵ-box dominance and neighborhood recombination for many-objective optimization," in Learning and Intelligent Optimization, 6th International Conference, LION 6 (Y. Hamadi and M. Schoenauer, eds.), (Paris, France), pp. 86–100, Springer. Lecture Notes in Computer Science Vol. 7219, January 16-20 2012.

[24] C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems. New York: Kluwer Academic Publishers, May 2002. ISBN 0-3064-6762-3.

[25] D. A. V. Veldhuizen and G. B. Lamont, "Evolutionary computation and convergence to a pareto front," in Late Breaking Papers at the Genetic Programming 1998 Conference (J. R. Koza, ed.), (Stanford University, California), pp. 221–228, Stanford University Bookstore, July 1998.

[26] D. A. V. Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm test suites," in Proceedings of the 1999 ACM Symposium on Applied Computing (J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, and G. B. Lamont, eds.), (San Antonio, Texas), pp. 351–357, ACM Press, 1999.

[27] S. Bandyopadhyay, R. Chakraborty, and U. Maulik, "Priority based epsilon dominance: A new measure in multiobjective optimization," Information Sciences, vol. 305, pp. 97–109, June 1 2015.

[28] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," Evolutionary Computation, vol. 8, no. 2, pp. 149–172, 2000.

[29] G. G. Yen and H. Lu, "Dynamic multiobjective evolutionary algorithm: Adaptive cell-based rank and density estimation," IEEE Transactions on Evolutionary Computation, vol. 7, pp. 253–274, June 2003.

[30] L. Rachmawati and D. Srinivasan, "Dynamic resizing for grid-based archiving in evolutionary multi-objective optimization," in 2007 IEEE Congress on Evolutionary Computation (CEC'2007), (Singapore), pp. 3975–3982, IEEE Press, September 2007.

[31] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," IEEE Transactions on Evolutionary Computation, vol. 17, pp. 721–736, October 2013.

[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," IEEE Transactions on Evolutionary Computation, vol. 6, pp. 182–197, April 2002.

[33] E. Montero, M.-C. Riff, and B. Neveu, "A beginner's guide to tuning methods," Applied Soft Computing, vol. 17, pp. 39–51, 2014.

[34] E. Montero and M.-C. Riff, "Towards a Method for Automatic Algorithm Configuration: A Design Evaluation Using Tuners," in Parallel Problem Solving from Nature - PPSN XIII, vol. 8672 of Lecture Notes in Computer Science, pp. 90–99, Springer International Publishing, 2014.

[35] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," IEEE Transactions on Evolutionary Computation, vol. 7, pp. 117–132, April 2003.

[36] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in Evolutionary Multiobjective Optimization. Theoretical Advances and Applications (A. Abraham, L. Jain, and R. Goldberg, eds.), pp. 105–145, USA: Springer, 2005.

[37] K. Tan, Y. Yang, and C. Goh, "A distributed cooperative coevolutionary algorithm for multiobjective optimization," Evolutionary Computation, IEEE Transactions on, vol. 10, pp. 527–549, October 2006.

[38] L. Miguel Antonio and C. A. Coello Coello, "A non-cooperative game for faster convergence in cooperative coevolution for multi-objective optimization," in 2015 IEEE Congress on Evolutionary Computation (CEC'2015), (Sendai, Japan), pp. 109–116, IEEE Press, 25-28 May 2015. ISBN 978-1-4799-7492-4.

[39] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," IEEE Transactions on Evolutionary Computation, vol. 10, pp. 477–506, October 2006.
[40] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms—a comparative study," in Parallel Problem Solving from Nature V (A. E. Eiben, ed.), (Amsterdam), pp. 292–301, Springer-Verlag, September 1998.
[41] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," Evolutionary Computation, vol. 8, pp. 173–195, Summer 2000.
[42] J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.
[43] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," SIAM J. on Optimization, vol. 8, pp. 631–657, Mar. 1998.

ADRIANA MENCHACA-MÉNDEZ is a Professor at the National Autonomous University of Mexico, ENES Campus Morelia, Michoacán since 2016. She received the M.Sc. and the Ph.D. degree in computer science from CINVESTAV-IPN, México, in 2008 and 2015, respectively. Her current research interests include multi-objective evolutionary optimization, hypervolume-based multi-objective evolutionary algorithms, combinatorial optimization, and parameter setting problems. She has published 13 technical papers in international conferences and journals. She has also served as a reviewer of different international journals and conferences in the field of evolutionary computation.

ELIZABETH MONTERO (M'08) is a Professor in Computing Science at the University Andres Bello, Chile. She holds a PhD from the University of Nice-Sophia Antipolis, France. Her research interests lie in the foundations and application of heuristic search methods, parameter setting problems and applications to combinatorial optimization. She has published over 30 technical papers in high level heuristic search conferences such as GECCO, PPSN, and CEC.

LUIS MIGUEL ANTONIO received a Bachelors degree in computer systems from the Escuela Superior de Cómputo (ESCOM), IPN, in Mexico City, México, in 2011. He also received a Masters and a PhD degree in computer science from the Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV-IPN), in Mexico City, México, in 2013 and 2017 respectively. His current research interests include evolutionary multi-objective optimization, large scale multi-objective optimization and machine learning.

SAÚL ZAPOTECAS-MARTÍNEZ (M'10) received the M.Sc. and the Ph.D. degree in computer science from CINVESTAV-IPN, Mexico, in 2007 and 2013, respectively. He was an Assistant Professor at the Department of Electrical and Electronic Engineering at Shinshu University, Nagano, Japan, from 2014 to 2016. Since 2017, Dr. Zapotecas is a visiting Professor at the Department of Applied Mathematics and Systems at the Metropolitan Autonomous University, Cuajimalpa unit (UAM-C), México City, México. Dr. Zapotecas has authored and co-authored over 40 technical papers and book chapters. His publications currently report over 400 citations in Google Scholar, and his H-index is 12. He has served in the program committee of several international conferences, and as a reviewer of several international journals in the field of evolutionary computation. Dr. Zapotecas is a Member of the IEEE and the ACM. His current research interests include: multi-objective evolutionary optimization, multi-objective benchmarking, and expensive multi-objective optimization.

CARLOS A. COELLO COELLO (M'98–SM'04–F'11) received PhD degree in computer science from Tulane University, USA, in 1996. He is currently Professor (CINVESTAV-3F Researcher) at the Computer Science Department of CINVESTAV-IPN, in Mexico City, México.

He is a Professor (CINVESTAV-3F Researcher) with the Department of Computer Science of CINVESTAV-IPN, Mexico City, Mexico. He has authored and co-authored over 450 technical papers and book chapters. His publications currently report over 40000 citations in Google Scholar with an H-index of 79. His current research interests include evolutionary multiobjective optimization and constraint-handling techniques for evolutionary algorithms.

Dr. Coello Coello was a recipient of the 2007 National Research Award from the Mexican Academy of Sciences in the area of exact sciences, the 2013 IEEE Kiyo Tomiyasu Award, the 2012 National Medal of Science and Arts in the area of Physical, Mathematical and Natural Sciences and the 2016 The World Academy of Sciences (TWAS) Award in Engineering Sciences. He is an Associate Editor of the IEEE Transactions on Evolutionary Computation and serves on the editorial board of several other international journals. He is a member of ACM, Sigma Xi, and the Mexican Academy of Science.