Noname manuscript No.
(will be inserted by the editor)

Diversidad en los Algoritmos Evolutivos

Carlos Segura · Joel Chacón Castillo

Received: date / Accepted: date

Resumen Keywords Diversidad · Multi-objetivo · Convergencia Prematura · Evolución Diferencial

1. Introducción

*Usos de los algoritmos evolutivos. *Definición de convergencia prematura. * *Como esta organizado el trabajo.

Evolutionary Algorithms (EAs) are one of the most widely used techniques to deal with complex optimization problems. Several variants of these strategies have been devised [?] and applied in many fields, such as in science, economic and engineering [?]. Among them, Differential Evolution (DE) [?] is one of the most effective strategies to deal with continuous optimization. In fact, it has been the winning strategy of several optimization competitions [?]. Similarly to other EAs, DE is inspired by the natural evolution process and it involves the application of mutation, recombination and selection. The main peculiarity of DE is that it considers the differences among vectors that are present in the population to explore the search space. In this sense it is similar to the Nelder-Mead [?] and the Controlled Random Search (CRS) [?] optimizers.

In spite of the effectiveness of DE, there exists several weaknesses that have been detected and partially solved by extending the standard variant [?]. Among them, the sensitivity to its parameters [?], the appearance of stagnation due to the reduced exploration capabilities [?, ?] and premature convergence [?] are some of the most well-known issues. This last one issue is tackled

Carlos Segura
Centro de Investigación en Matemáticas, Guanajuato, Mexico
E-mail: carlos.segura@cimat.mx

Joel Chacón Castillo
Centro de Investigación en Matemáticas, Guanajuato, Mexico
E-mail: joel.chacon@cimat.mx

in this paper. Note that, attending to the proper design of population-based meta-heuristics [?], special attention must be paid to attain a proper balance between exploration and exploitation. A too large exploration degree prevents the proper intensification of the best located regions, usually resulting in a too slow convergence. Differently, an excessive exploitation degree provokes loss of diversity meaning that only a limited number of regions are sampled.

Since the appearance of DE, some criticism appeared because of its incapability to maintain a large enough diversity due to the use of a selection with high pressure [?]. Thus, several extensions of DE to deal with premature convergence have been devised such as parameter adaptation [?], auto-enhanced population diversity [?] and selection strategies with a lower selection pressure [?]. Some of the last studies on design of population-based meta-heuristics [?] show that explicitly controlling the diversity to properly balance the exploration and intensification degree is particularly useful. Specifically, in the field of combinatorial optimization some novel replacement strategies that dynamically alter the balance between exploration and exploitation have appeared [?]. The main principle of such proposals is to use the stopping criterion and elapsed generations to bias the decisions taken by the optimizers with the aim of promoting exploration in the initial stages and exploitation in the last ones. Probably their main weakness is that the time required to obtain high-quality solution increases. Our novel proposal, which is called DE with Enhanced Diversity Maintenance (DE-EDM), integrates a similar principle into DE. However, in order to avoid the excessive growth of computational requirements typical of diversity-based replacement strategies, the method was designed with the aim of inducing a larger degree of intensification.

The rest of the paper is organized as follows. Some basic concepts of DE and a review of works related to diversity within DE are given in section 2. Section ?? presents an analysis about the algorithms with best performance on the last continuous optimization contests held at the IEEE Congress on Evolutionary Computation. More emphasis is given on the variants based on DE. Our proposal is described in section ??. The experimental validation, which includes comparisons against state-of-the-art approaches, is shown in section ??. Finally, our conclusions and some lines of future work are given in section 5.

2. Mecanismos para evitar la convergencia prematura

3. Diseño de Evolución Diferencial basado en diversidad

long term [1].

3.1. Evolución Diferencia: Conceptos Básicos

This section is devoted to summarize the classic DE variant and to introduce some of the most important terms used in the DE field. The classic DE scheme is called the DE/rand/1/bin and it has been extensively used to generate more complex DE variants [?]. In fact, our proposal also extends the classic variant. However, in order to perform fair comparisons, our experimental validation takes into account state-of-the-art approaches that incorporate more complex components and even algorithms not belonging to the DE field.

DE was originally proposed as a direct search method for single-objective continuous optimization. The variables governing a given problem performance are given as a vector like $\mathbf{X} = [x_1, x_2, ..., x_D]$, where D is the dimension of the problem. In continuous optimization, each x_i is a real number and usually box-constraints are given, i.e. there is a lower bound (a_i) and upper bound (b_i) for each variable. The aim of the optimization process is to obtain the vector $\mathbf{X}^*$ which minimizes a given objective function, mathematically denoted by $f : \Omega \subseteq \Re^D \to \Re$. In the box-constrained case $\Omega = \prod_{j=1}^{D} [a_j, b_j]$.

DE is a population-based stochastic algorithm, so it iteratively evolves a set of candidate solutions. In DE such candidate solutions are usually called vectors. In the basic DE variant for each member of the population — they are called *target vectors* — a new *mutant vector* is created. Then, the mutant vector is combined with the target vector to generate a *trial vector*. Finally, a selection phase is applied to choose the survivors. In this way, several generations are evolved until a stopping criterion is reached. The ith vector of the population at the generation G is denoted as $\mathbf{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, ..., X_{D,i,G}]$. In the following more details are given for each component of DE.

3.1.1. Initialization

DE usually starts the optimization process with a randomly initiated population of NP vectors. Since there is commonly no information about the performance of different regions, uniform random generators are usually applied. Hence, the jth component of the ith vector is initialized as $x_{j,i,0} = a_j + rand_{i,j}[0,1](b_j - a_j)$, where $rand_{i,j}[0,1]$ is an uniformly distributed random number lying between 0 and 1.

3.1.2. Mutation

For each target vector a mutant vector is created and several ways of performing such a process have been proposed. In the classic DE variant the rand/1 strategy is applied. In this case, the mutant vector $V_{i,G}$ is created as follows:

$$\mathbf{V}_{i,G} = \mathbf{X}_{r1,G} + F \times (\mathbf{X}_{r2,G} - \mathbf{X}_{r3,G}) \quad r1 \neq r2 \neq r3 \tag{1}$$

The indices $r1, r2, r3 \in [1, NP]$ are different integers randomly chosen from the range $[1, NP]$. In addition, they are all different from the index i. It is important to take into account that the difference between vectors is scaled with the number F, which is usually defined in the interval [0,4, 1]. The scaled difference is added to a third vector, meaning that when diversity decreases and consequently differences are low, mutant vectors are similar to target vectors. As a result, maintaining some degree of diversity is specially important in DE.

3.1.3. Crossover

In order to combine information of different candidate solutions and with the aim of increasing diversity, the crossover operator is applied. Specifically, each target vector $\mathbf{X}_{i,G}$ is mixed with its corresponding mutant vector $V_{i,G}$ to generate the trial vector $\mathbf{U_{i,G}} = [u_{1,i,G}, u_{2,i,G}, ..., u_{D,i,G}]$. The most typical crossover is the *binomial* one, which operates as follows:

$$\mathbf{U}_{j,i,G} = \begin{cases} \mathbf{V}_{j,i,G}, & \text{if}(rand_{i,j}[0,1] \leq CR \quad or \quad j = j_{rand}) \\ \mathbf{X}_{j,i,G}, & \text{otherwise} \end{cases} \tag{2}$$

where $rand_{i,j}[0,1]$ is a uniformly distributed random number, j_{rand} is a randomly chosen index which ensures that $\mathbf{U}_{i,G}$ inherits at least one component from $\mathbf{V}_{i,G}$ and $CR \in [0,1]$ is the crossover rate.

3.1.4. Selection

Finally, a greedy selection is performed to determine the survivors of the next generation. Each trial vector is compared with its corresponding target vector and the best one survives:

$$\mathbf{X}_{j,i,G+1} = \begin{cases} \mathbf{U}_{i,G}, & \text{if} \quad f(\mathbf{U}_{i,G}) \leq f(\mathbf{X}_{i,G}) \\ \mathbf{X}_{i,G}, & \text{otherwise} \end{cases} \tag{3}$$

Hence, each population member either gets better or remains with the same objective value in each generation. Since members never deteriorate, it is considered to be a selection with high pressure. Note that in case of a tie, the trial vector survives.

3.2. Diversity in Differential Evolution

DE is highly susceptible to the loss of diversity due to the greedy strategy applied in the selection phase. However, several analyses to better deal with this issue have been carried out. Since the general implications of each parameter on the diversity are known, one of the alternatives is to theoretically estimate proper values for the DE parameters [?]. Differently, some analyses regarding the effects of the norm of the difference vectors used in the mutation have also been performed [?]. Such analyses and additional empirical studies regarding the crossover allowed to conclude that some kind of movements might be disallowed to delay the convergence [?]. In this last study the kind of accepted movements varies along the run. Specifically, it discards movements with a size below a threshold and this threshold decreases taking into account the elapsed generations. Other ways of altering the kind of accepted movements have been proposed [?]. Note that these kinds of methods have similarities with our proposal in the sense that decisions are biased by the number of elapsed generations. However, our method operates on the replacement strategy and not on the mutation phase. Moreover, these methods do not consider explicitly the differences appearing on the whole population. Instead, the restrictions apply to the differences appearing in the reproduction phase.

A different alternative operates by altering the selection operator [?]. Particularly, the selection pressure is relaxed through a probabilistic selection to maintain the population diversity and consequently to allow escaping from basin of attraction of local optima. Since it considers the fitness to establish the acceptance probabilities, it is very sensitive to scale transformations. In this case, decisions are not biased by the elapsed generations.

Finally, in the *Auto-Enhanced Population Diversity* (AEPD), the diversity is explicitly measured and it triggers a mechanism to diversify the population when a too low diversity is detected [?]. Strategies with similar principles but with different disturbance schemes have also been devised [?].

Note that DE variants with best performance in competitions do not apply these modifications and that most of these extensions have not been implemented in the most widely used frameworks. As a result, these extensions are not so widely used in the community in spite of their important benefits for some cases.

3.3. Propuesta

Our proposal is motivated by two main works in the area of control of diversity in EAs. The first one is the empirical study developed by Montgomery et al [?], which presents several empirical analyses that confirm issues related to premature convergence in DE. The second work, by Segura et al. [?], provides significant improvements in the combinatorial optimization field by developing a novel replacement strategy called *Replacement with Multi-objective based Dynamic Diversity Control* (RMDDC) that relates the control of diversity with the stopping criterion and elapsed generations. Important benefits were attained by methods including RMDDC, so given the conclusions of these previous works, the proposal of this paper is a novel DE variant that includes an explicit mechanism that follows some of the principles of RMDDC. This novel optimizer is called *Differential Evolution with Enhanced Diversity Maintenance* (DE-EDM) and its source code is freely available [1].

The core of DE-EDM (see Algorithm **??**) is quite similar to the standard DE. In fact the way of creating new trial solutions is not modified at all (lines 5 and 6). The novelty is the incorporation of an elite population (E) and a novel diversity-based replacement strategy. In order to select the members of the

[1] The code in C++ can be downloaded in the next link `https://github.com/joelchaconcastillo/Diversity_DE_Research.git`

Algorithm 1 General scheme of DE-EDM

```
1: Randomly initialize the population of $NP$ individuals, where each one is uniformly distributed.
2: $G = 0$
3: while stopping criterion is not satisfied do
4:   for $i = 1$ to $NP$ do
5:     Mutation: Generate the mutant vector ($V_{i,G}$) according to Eq. (1).
6:     Crossover: use recombination to generate the trial vector ($U_{i,G}$) according to Eq. (2).
7:     Selection: Update the elite vector ($E_{i,G}$ instead of $X_{i,G}$) according to Eq. (3).
8:   Replacement: Select the target vectors ($X_{G+1}$) according to Algorithm ?? .
9:   $G = G + 1$
```

elite population, the original greedy replacement of DE is used (line 7). On the other way, the replacement strategy (line 8), which is in charge of selecting the next population members, follows the same principle that guided the design of RMDDC, i.e. individuals that contribute too little to diversity should not be accepted as members of the next generation. In this way, the greedy selection strategy of DE is not used to maintain the parent population (X). In order to establish the minimum acceptable diversity contribution to be selected, the stopping criterion and elapsed generations are taken into account. One of the main weaknesses of RMDDC is that its convergence is highly delayed. Thus, in order to promote a faster convergence than in RMDDC two modifications are performed. First, no concepts of the multi-objective field are applied, instead a more greedy selection is taken into account. Second, the elite population is also considered as an input of the replacement strategy.

Our replacement strategy (see Algorithm **??**) operates as follows. It receives as input the parent population (target vectors), the offspring population (trial vectors), and the elite population. In each generation it must select the NP vectors of the next parent population. First, it calculates the desired minimum distance D_t given the current number of elapsed function evaluations (line 2). Then, it joins the three populations in a set of current members (line 3). The current members set contains vectors that might be selected to survive. Then, the set of survivors and penalized individuals are initialized to the empty set (line 4). In order to select the NP survivors (next parent population) an iterative process is repeated (lines 5 - 13). In each step the best individual in the *Current set*, i.e. the one with best objective function is selected to survive, i.e. it is moved to the *Survivor set* (line 6 - 8). Then, individuals in the *Current set* with a distance metric lower than D_t are transferred to the *Penalized set* (line 9). The way to calculate the distance between two individuals is by using the normalized Euclidean distance described in Eq. **??**, where D is the dimension of the problem, and a_d, b_d are the minimum and maximum bounds of each dimension (d). In cases where the *Current set* is empty previous to the selection of NP individuals, the *Survivor set* is filled by selecting in each step the individual in *Penalized* with the largest distance to the closest individual in the *Survivor set* (lines 10 - 13).

Algorithm 2 Replacement Phase

```
1: Input: $Population$ ($target\ vectors$), $Offspring$ ($trial\ vectors$), and $Elite$
2: Update $D_t = D_I - D_I * (nfes/(0{,}95 * max\_nfes))$
3: $Current = Population \cup Offspring \cup Elite$.
4: $Survivors = Penalized = \emptyset$.
5: while $|Survivors| < NP$ And $|Current| > 0$ do
6:     $Selected$ = Select the best individual of $Current$.
7:     Remove $Selected$ from $Current$.
8:     Copy $Selected$ to $Survivors$.
9:     Find the individuals from $Current$ with a distance to $Selected$ lower than $D_t$ and move
       them to $Penalized$. Normalized distance is considered (Eq. ??).
10: while $|Survivors| < NP$ do
11:     $Selected$ = Select the individual from $Penalized$ with the largest distance to the closest
        individual in $Survivors$.
12:     Remove $Selected$ from $Penalized$.
13:     Copy $Selected$ to $Survivors$.
14: return $Survivors$
```

$$distance(x_i, x_j) = \frac{\sqrt{\sum_{d=1}^{D} \left(\frac{x_i^d - x_j^d}{b_d - a_d} \right)^2}}{\sqrt{D}} \tag{4}$$

In order to complete the description it is important to specify the way to calculate D_t and the methodology to update the elite individuals. All the remaining steps are maintained as in the classic DE variant. The value of D_t is used to alter the degree between exploration and explotation so it should depend on the optimization stage. Specifically, this value should be reduced as the stopping criterion is reached with the aim of promoting explotation. In our scheme, an initial value for D_t (D_I) must be set. Then, similarly than in [?], a linear reduction of D_t is performed by taking into account the elapsed function evaluations and stopping criterion. Particularly, in this work, the stopping criterion is set by function evaluations. The reduction is calculated in such a way that by the 95 % of maximum number of evaluations the resulting D_t value is 0. Therefore, in the remaining 5 % diversity is not considered at all. Thus, if max_nfes is the maximum number of evaluations and $nfes$ is the elapsed number of evaluations, D_t can be calculated as $D_t = D_I - D_I * (nfes/(0{,}95 * max\_nfes))$.

The initial distance (D_I) heavily affects the performance of DE-EDM. If this parameter is fixed large enough, then at the first optimization stages the algorithm aims to maximize the diversity of the population, so a proper exploration is performed which is very important in some kinds of problems such as highly multimodal and deceptive ones. Thus, the effect of premature convergence might be alleviated. A too large D_I might induce too much exploration so a proper exploitation phase is not performed. In the opposite case, a too low D_I might avoid the exploration phase, so avoiding local optima is more difficult. Depending on the kind of fitness landscape and stopping criterion, the optimal D_I might vary. For instance, deceptive and highly multi-modal problems usually require larger values than unimodal problems. However, in our proposal, D_I is not adapted to each problem, instead an experimental study to check the robustness of different D_I value is attached in the experimental validation section.

Similarly that the standard DE, in DE-EDM the crossover probability (CR) and the mutation factor (F) must be set. The first one is perhaps the most important for the performance according to several studies developed by Montgomery et al. [1]. These authors empirically proved that extremes CR values leads to vastly different search behaviors. They explained that low CR values result in a search that is aligned with a small number of search space axes and induce small displacements. This provokes a gradual and slow convergence that in some scenarios might result in a robust behavior. Additionally, high CR values might generate higher quality solutions with a lower probability. However, these transformations provoke large displacements that could improve significantly the solutions when successful. According to this, we employ both high and low CR values as it is showed in Eq. **??**.

$$CR = \begin{cases} Normal(0{,}2, 0{,}1), & \text{if} \quad rand[0,1] \leq 0{,}5 \\ Normal(0{,}9, 0{,}1), & \text{otherwise} \end{cases} \tag{5}$$

Following the principles of several SHADE variants [**?**, **?**], the function evaluations are considered in the random generation of the mutation factor F. Particularly, each F is sampled through a Cauchy distribution (Eq. **??**).

$$Cauchy(0{,}5, 0{,}5 * nfes/max\_nfes) \tag{6}$$

Therefore, at the first optimization stages, F values near to 0,5 are generated. Then, as the execution advances, the density function suffers a gradual transformation and the variance is increased, meaning that values outside the interval [0,0, 1,0] are generated with a higher probability. In the cases when values larger than 1,0 are generated, the value 1,0 is used. In the case of generating a negative value, the F is resampled. One of the effects of this approach is to increase the probability of generating large F-values as the execution progresses with the aim of avoiding a fast convergence.

3.4. Resultados

In this section the experimental validation is presented. Specifically, we show that by explicitly controlling the diversity in DE, results of state-of-the-art algorithms are improved further. Particularly, the benchmarks of CEC 2016 and CEC 2017 are considered. Each one of them is composed of thirty different problems. The state-of-the-art is composed by the algorithms that attained the first places of each year competition. Additionally, the standard DE was included. Thus, the algorithms considered from the CEC 2016 are UMOEAs-II [**?**] and L-SHADE-EpSin [**?**] that achieved the first and second place respectively. Similarly, the top algorithms from CEC 2017 are EBOwithCMAR [**?**] and jSO [**?**]. It is interesting to remark that EBOwithCMAR is considered as an improvement of the UMOEAs-II. Additionally, jSO and L-SHADE-EpSin belong to the SHADE's family. All these algorithms are tested with both benchmarks as it is suggested by [?].

Given that all of them are stochastic algorithms, each execution was repeated 51 times with different seeds. In every case, the stopping criterion was set to $25,000,000$ functions evaluations. We performed our evaluation following the guidelines of CEC benchmark competitions. Thus, if the gap between the values of the best solution found and the optimal solution was 10^{-8} or smaller, the error is treated as 0. The parameterization indicated by the authors was used in every algorithm and it is as follows:

- **EBOwithCMAR**: For EBO, the maximum population size of $S_1 = 18D$, minimum population size of $S_1 = 4$, maximum population size of $S_2 = 146{,}8D$, minimum population size of $S_2 = 10$, historical memory size H=6. For CMAR Population size $S_3 = 4+3log(D)$, $\sigma = 0{,}3$, CS = 50, probability of local search $pl = 0{,}1$ and $cfe_{ls} = 0{,}4 * FE_{max}$.
- **UMOEAs-II**: For MODE, maximum population size of $S_1 = 18D$, minimum population size of $S_1 = 4$, size memory H=6. For CMA-ES Population size $S_2 = 4 + \lfloor 3log(D) \rfloor$, $\mu = \frac{PS}{2}$, $\sigma = 0{,}3$, CS = 50. For local search, $cfe_{ls} = 0{,}2 * FE_{max}$.
- **jSO**: Maximum population size = $25log(D)\sqrt{D}$, historical memory size H= 5, initial mutation memory $M_F = 0{,}5$, initial probability memory $M_{CR} = 0{,}8$, minimum population size = 4, initial p-best = $0{,}25 * N$, final p-best = 2.
- **L-SHADE-EpSin**: Maximum population size = $25log(D)\sqrt{D}$, historical memory size H= 5, initial mutation memory $M_F = 0{,}5$, initial probability memory $M_{CR} = 0{,}5$, initial memory frequency $\mu_F = 0{,}5$, minimum population size = 4, initial p-best = $0{,}25 * N$, final p-best = 2, generations of local search $G_{LS} = 250$.
- **DE-EDM**: $D_I = 0{,}3$, population size = 250.
- **Standard-DE**: population size = 250 (operators as DE-EDM).

Our experimental analyses have been performed in base of the difference between the optimal solution and the best obtained solution. In order to statistically compare the results, a similar guideline than the one proposed in [?] was used. First a Shapiro-Wilk test was performed to check whatever or not the values of the results followed a Gaussian distribution. If, so, the Levene test was used to check for the homogeneity of the variances. If samples had equal variance, an ANOVA test was done; if not, a Welch test was performed. For non-Gaussian distributions, the non parametric Kruskal-Wallis test was used to test whether samples are drawn from the same distribution. An algorithm X is said to win algorithm Y when the differences between them are statistically significant, and the mean and median obtained by X are higher than the mean and median achieved by Y.

In tables **??** and **??** a summary of the results obtained for CEC 2016 and CEC 2017 are shown, respectively. The column tagged with "Always Solved" shows the number of functions where a zero error was obtained in the 51 runs. Additionally, column tagged with "At least one time solved" shows the number of functions that were solved to optimality at least in one run. Practically all functions (28 of them) of the CEC 2017 benchmark were solved with our proposal at least one time. Additionally, 21 functions of the CEC 2016 were also solved. This constrast with the results obtained by state-of-the-art algorithms. They were able to reach optimal values in significantly less functions. In order to confirm the superioriy of DE-EDM, pair-wise statisticall test were used. The column tagged with the symbol $\uparrow$ shows the number of times that the superiority of each method could be confirmed, whereas the column tagged with the symbol $\downarrow$ count the number of cases where the method was inferior. Finally, the number comparisons whose differences were not significant are shown in the column tagged with the symbol $\longleftrightarrow$. The statistical tests indicate that the DE-EDM attained the best results in both years. The number of wins in CEC 2016 and CEC 2017 were 77 and 88 respectively. Also the number of losts were of 25 and 6 respectively. Additionally, the last place attained in both years was by the L-SHADE-Epsilon with 20 wins in 2016 and 7 wins in 2017. The last column tagget with "Score" shows the analyses proposed in the CEC's competitions. Particularly, the evaluation method combines two scores defined in the equation (**??**). Thus the final score is composed by the sum $Score = Score_1 + Score_2$.

$$
\begin{aligned}
Score_1 &= \left(1 - \frac{SE - SE_{min}}{SE}\right) \times 50, \\
Score_2 &= \left(1 - \frac{SR - SR_{min}}{SR}\right) \times 50,
\end{aligned}
\tag{7}
$$

Here, SE_{min} is the minimal sum of errors from all the algorithms, and SE is the sum of error values $SE = \sum_{i=1}^{30} error\_f_i$. Also, SR_{min} is the minimal sum of ranks from all the algorithms, namely the sum of each rank in each function for the considered algorithms $SE = \sum_{i=1}^{30} error\_f_i$. Principally, our proposal attained the best scores (100,00) in both years, showing its superiority. Additionally, the Standard-DE attained good enough results, in fact it got the third and second places in CEC 2016 and CEC 2017 respectively. This shows that the performance of the state-of-the-art algorithms is different considering long-term executions. Specifically, although that in CEC 2017 the L-SHADE-Epsilon algorithm got the lowest number of wins in the statistical test it showed a competitive score. This might occurs since that the statistical scores considers both mean and median errors. Morever, the score considers a rank and mean based in the error.

Giving that our proposal is based in the explicitly control of the diversity and with the aim of a better understanding of its behavior in the figure **??** is showed the diversity through the elapsed function evaluatons. Particularly, the DE-EDM was executed with the functions f_1 and f_{30}. Specifically, based in their properties the first function is easily solved (unimodal) and the second is one of the most difficult (hybrid). In the left side is showed the diversity of the Elite population. Although there are not constraints in the Elite population to lost the diversity, it seems that in both functions f_1 and f_{30} the diversity is

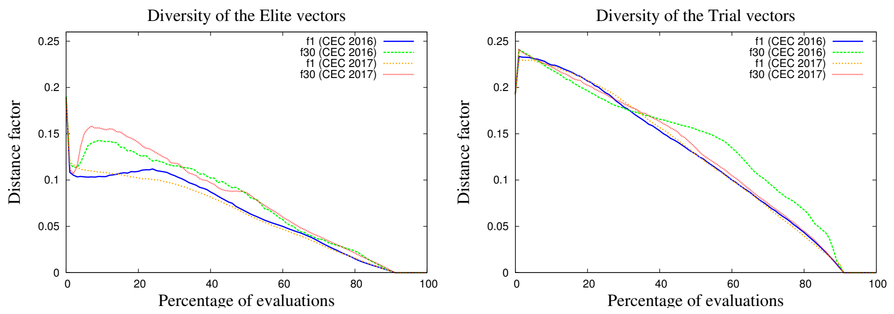

Figura 1 Average DCN of the 51 executions with the problems f_1 and f_{30} (CEC 2016 and CEC 2017). The initial distance factor considered corresponds to $D_I = 0{,}3$.

Cuadro 1 Summary results - CEC 2016

Algorithm	Always solved	At least one time solved	Statistical Tests			Score
			$\uparrow$	$\downarrow$	$\longleftrightarrow$	
EBOwithCMAR	8	14	35	56	59	50.28
jSO	9	17	47	51	52	55.43
UMOEAs-II	9	14	51	31	68	62.45
L-SHADE-Epsilon	7	13	20	71	59	50.12
DE-EDM	13	21	77	25	48	100.00
Standard-DE	11	19	50	46	54	56.29

Cuadro 2 Summary results - CEC 2017

Algorithm	Always solved	At least one time solved	Statistical Tests			Score
			$\uparrow$	$\downarrow$	$\longleftrightarrow$	
EBOwithCMAR	9	18	34	46	70	37.14
jSO	8	15	29	55	66	29.30
UMOEAs-II	11	15	43	40	67	26.89
L-SHADE-Epsilon	8	19	7	81	62	32.78
DE-EDM	21	28	88	6	56	100.00
Standard-DE	12	21	56	29	65	42.91

implicitly maintained. Similarly, in the right side is showed the diversity of the trial vectors. It shows that the diversity is explicitly maintained as is desired (i.e. until the 95 % of the total function evaluations).

In order, to provide comparable results of our proposal, in the tables **??** and **??** are reported the best, worst, median, mean, standard deviation and success ratio. Particularly, these tables show that the uni-modal were solved by our proposal. Also, several simple multi-modal functions were adequatelly approximated. Principally, our proposal solved several complex functions (e.g. Composition Functions) that were not solved by the state-of-the-art.

Cuadro 3 Results for DE based diversity CEC 2016 problems

	Best	Worst	Median	Mean	Std	Succ. Ratio
f_1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_6	0.00E+00	3.60E-02	4.00E-03	7.39E-03	1.15E-02	3.92E-01
f_7	2.00E-02	1.02E-01	5.90E-02	5.77E-02	4.93E-02	0.00E+00
f_8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{10}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{11}	0.00E+00	6.00E-02	0.00E+00	5.88E-03	1.90E-02	9.02E-01
f_{12}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{13}	1.00E-02	8.00E-02	5.00E-02	4.67E-02	2.60E-02	0.00E+00
f_{14}	1.00E-02	5.00E-02	3.00E-02	2.82E-02	2.13E-02	0.00E+00
f_{15}	0.00E+00	4.70E-01	2.20E-01	1.99E-01	1.55E-01	1.96E-02
f_{16}	4.00E-02	1.50E-01	8.00E-02	8.47E-02	4.96E-02	0.00E+00
f_{17}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{18}	0.00E+00	2.00E-02	1.00E-02	7.65E-03	6.32E-03	3.14E-01
f_{19}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{20}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{21}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{22}	0.00E+00	3.00E-02	0.00E+00	3.73E-03	2.76E-02	7.65E-01
f_{23}	0.00E+00	1.00E+02	0.00E+00	2.55E+01	5.10E+01	7.45E-01
f_{24}	0.00E+00	6.90E-01	0.00E+00	2.61E-02	1.33E-01	9.61E-01
f_{25}	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00	0.00E+00
f_{26}	8.00E-02	1.00E+02	5.29E+01	5.20E+01	3.19E+01	0.00E+00
f_{27}	2.50E-01	9.10E-01	5.40E-01	5.60E-01	2.92E-01	0.00E+00
f_{28}	0.00E+00	3.57E+02	3.43E+02	2.76E+02	1.60E+02	1.96E-01
f_{29}	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00	0.00E+00
f_{30}	1.84E+02	1.84E+02	1.84E+02	1.84E+02	3.25E-02	0.00E+00

3.5. Empirical analyses of the initial distance factor

In our proposal the diversity is explicitly promoted through several stages, which are controlled with the initial distance factor D_I. Therefore, the effect of this parameter is analysed in detail. Particularly, the general configuration of the experimental validation is taken into account. Thus, several initial distance factors were considered ($D_I = \{0{,}0, 0{,}1, 0{,}2, 0{,}3, 0{,}4, 0{,}5, 0{,}6, 0{,}7, 0{,}8, 0{,}9, 1{,}0, 1{,}1\}$).

In the figure **??** is showed the average success ratio vs. the initial distance factor D_I. The most relevant points are described as follows:

- If the diversity is not promoted ($D_I = 0{,}0$) the performance of the algorithms is seriously implicated.
- In this scenario the ideal configuration is $D_I = 0{,}3$, although that the range $[0{,}1, 0{,}4]$ also provides quality solutions.
- If the diversity of the solutions increases (after a range) the quality of solutions is implicated.

Finally, its important stand out that the solutions are less affected by the population size, however there is still present a relation between the D_I and the population size.

Cuadro 4 Results for DE based diversity CEC 2017 problems

	Best	**Worst**	**Median**	**Mean**	**Std**	**Succ. Ratio**
f_1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_7	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{10}	0.00E+00	1.20E-01	0.00E+00	1.65E-02	3.39E-02	7.45E-01
f_{11}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{12}	0.00E+00	2.20E-01	0.00E+00	6.37E-02	1.76E-01	6.67E-01
f_{13}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{14}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{15}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{16}	0.00E+00	2.10E-01	0.00E+00	2.47E-02	7.27E-02	8.82E-01
f_{17}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{18}	0.00E+00	1.00E-02	0.00E+00	1.96E-03	4.47E-03	8.04E-01
f_{19}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{20}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{21}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{23}	0.00E+00	3.00E+02	0.00E+00	3.49E+01	1.03E+02	8.82E-01
f_{24}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{25}	0.00E+00	1.00E+02	0.00E+00	3.92E+00	2.00E+01	9.61E-01
f_{26}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{27}	0.00E+00	3.87E+02	3.87E+02	2.05E+02	2.68E+02	1.96E-02
f_{28}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{29}	1.45E+02	2.26E+02	2.18E+02	1.99E+02	4.21E+01	0.00E+00
f_{30}	3.95E+02	3.95E+02	3.95E+02	3.95E+02	2.10E-01	0.00E+00

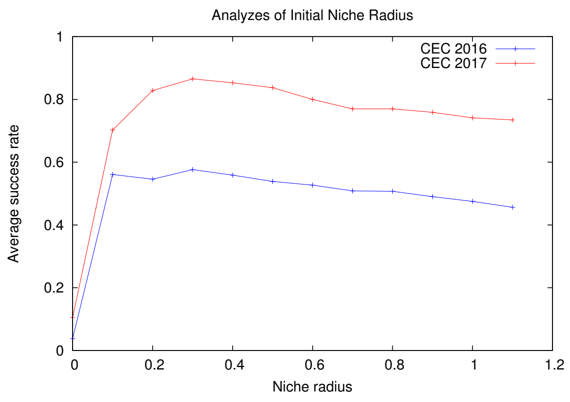

Figura 2 Average success rate with different initial distance factors in the benchmark of CEC 2016 and CEC 2017, is considered a population size of 250 and $25,000,000$ function evaluations.

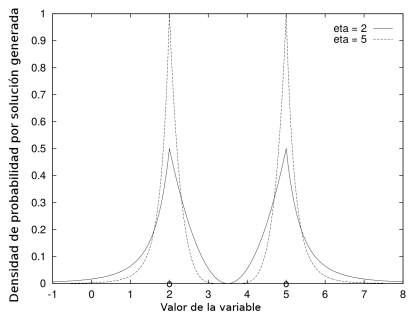

Figura 3 Función de densidad del operador SBX con índices de distribución 2 y 5.

4. Diseño de operadores de cruce basados en diversidad

4.1. Conceptos Básicos

Es uno de los operadores de cruce más utilizados en los MOEAs y es clasificado como centrado en los padres, particularmente tiene la propiedad de poder crear cualquier valor en el espacio de búsqueda combinando dos individuos padres. En el SBX en base a dos individuos padres (p_1, p_2) se generan dos hijos (c_1, c_2) considerando a una distribución de probabilidad. Para tener más control la forma de la distribución de la probabilidad se puede controlar con un índice de distribución η_c que determina la apertura de la distribución, y por tanto, la capacidad de exploración. Específicamente, un índice de distribución pequeño induce una mayor probabilidad de construir soluciones hijas alejadas de los padres, mientras que con un índice de distribución elevado, la probabilidad de crear soluciones hijas más similares a los padres se incrementa. El efecto de η_c es ilustrado en la Figura 1, donde se muestran las funciones de densidad de dos índices de distribución distintos. Específicamente los círculos representan a los padres, y se puede apreciar que con $\eta_c = 5$, la probabilidad de crear soluciones más cercanas a los padres es mayor que con $\eta_c = 2$.

El SBX tiene la propiedad de preservar una relación entre la media de los valores padres e hijos ($c_1 + c_2 = p_1 + p_2$), con lo que se define el factor de dispersión $\beta = |c_1 - c_2|/|p_1 - p_2|$, y en base al valor de β, podemos determinar donde quedarían localizados los hijos. En el SBX la probabilidad de distribución para $\beta \in [0, \infty]$ viene dada por:

$$P(\beta) = \begin{cases} 0{,}5(\eta_c + 1)\beta^{\eta_c}, & \text{si} \quad \beta \leq 1 \\ 0{,}5(\eta_c + 1)\frac{1}{\beta^{\eta_c+2}}, & \text{de otra forma} \end{cases} \tag{8}$$

En base a esto, se cumplen las siguientes propiedades:

- Los valores de las soluciones hijas son equidistantes de los padres.
- Existe una probabilidad no nula de generar una solución hija en cualquier parte en el espacio independientemente de donde se localicen los padres.

- La probabilidad de crear un par de soluciones hijas dentro del rango de las soluciones padres es idéntica a la probabilidad de crear dos soluciones hijas fuera de dicho rango.

A la hora de implementar esta distribución para dos valores de los individuos padres (p_1, p_2), se crean los valores de los hijos (c_1, c_2) como combinación lineal de los valores padres con un número aleatorio $u \in [0,1]$ de la forma ([**?**]):

$$\begin{aligned} c_1 &= 0{,}5(1+\beta(u))p_1 + 0{,}5(1-\beta(u))p_2 \\ c_2 &= 0{,}5(1-\beta(u))p_1 + 0{,}5(1+\beta(u))p_2 \end{aligned} \tag{9}$$

Así, para realiza la simulación del parámetro $\beta(u)$, primero se genera un número aleatorio $u \in [0,1]$, y se utiliza en la siguiente fórmula:

$$\beta(u) = \begin{cases} (2u)^{\frac{1}{\eta_c+1}}, & \text{si} \quad u \leq 0{,}5, \\ (\frac{1}{2(1-u)})^{\frac{1}{\eta+1}}, & \text{de otra forma} \end{cases} \tag{10}$$

Se puede observar que la forma de la distribución no está afectada por los límites de cada variable, en base a este inconveniente, los autores [**?**] modificaron la ecuación (6) de forma que exista una probabilidad cero de crear individuos fuera del dominio de la variable, por lo tanto se puede calcular $\beta(u,a)$ en función de un número aleatorio $u \in [0,1]$ de la forma:

$$\beta(u,a) = \begin{cases} (2u(1-\rho_a))^{\frac{1}{\eta_c+1}}, & \text{si} \quad u \leq 0{,}5/(1-\rho_a), \\ (\frac{1}{2(1-u(1-\rho_a))})^{\frac{1}{\eta+1}}, & \text{de otra forma} \end{cases} \tag{11}$$

donde $a = x_i^{(L)}$ es el límite inferior y $b = x_i^{(U)}$ es el límite superior, y $\rho_a = 1/(2\beta_a^{\eta_c+1})$, donde $\beta_a = 1 + (p_1 - a)/(p_2 - p_1)$. Similarmente, ρ_b es calculado remplazando β_a por $\beta_b = 1 + (b - p_2)/(p_2 - p_1)$, así $\beta(u,b)$ es calculado con la ecuación (7), generando así dos individuos hijo en base a sus respectivas funciones de distribución $\beta(u,a)$ y $\beta(u,b)$.

En la práctica, se trabajan con funciones de múltiples variables, con lo que hay que generalizar lo anterior. Una forma sencilla sería aplicar lo mismo a cada variable, pero se ha visto que eso es muy disruptivo, con lo que habitualmente no se cambian todas las variables. Por ejemplo, en la implementación que más se usa en la práctica (jMETAL, NSGA-II, etc.) cuando se aplica el SBX cada variable sufre el proceso de cruce con una probabilidad igual a 0.5, mientras que en el resto de casos los valores se heredaran sin ser alterados ([**?**, **?**]). Además, cuando se produce un cruce también aplican intercambios entre las valores generadas con probabilidad 0.5, lo que resulta en reflexiones que son analizados posteriormente. Por último, cabe destacar que no se ha podido encontrar variantes del operador SBX en las que en lugar de usar dichas probabilidades fijas en 0.5, estas sean cambiadas a lo largo de la ejecución.

Algorithm 3 Operador de Cruce Binario Simulado (SBX)

```
Entrada: Individuos Padre ($P_1, P_2$), Indice de distribución ($\eta_c$), Probabilidad de cruza ($P_c$).
Salida: Individuos hijo($C_1, C_2$).
$r_1 \leftarrow U[0,1]$.
if $r_1 \le P_c$ then
    for cada variable d do
        if $U[0,1] \le 0{,}5$ then
            $a = LowBound(d)$.
            $b = UpperBound(d)$.
            $r_2 \leftarrow U[0,1]$.
            $\beta_a = 1 + (p_1 - a)/(p_2 - p_1)$.
            $\rho_a = 1/(2\beta_a^{\eta_c+1})$.
            Utilizar $r_2$ y $\rho_a$ en la ecuación 7 para generar $\beta(u,a)$.
            Generar a $C_1(d)$ utilizando $\beta(u,a)$ en la ecuación 5.
            $\beta_b = 1 + (b - p_2)/(p_2 - p_1)$.
            $\rho_b = 1/(2\beta_b^{\eta_c+1})$.
            Utilizar $r_2$ y $\rho_b$ en la ecuación 7 para generar $\beta(u,b)$.
            Generar a $C_2(d)$ utilizando $\beta(u,b)$ en la ecuación 5.
            if $U[0,1] \le 0{,}5$ then
                Intercambiar los valores de $C_1(d)$ con $C_2(d)$.
        else
            $C_1(d) = P_1(d)$.
            $C_2(d) = P_2(d)$.
else
    $C_1 = P_1$
    $C_2 = P_2$
```

4.2. Análisis operador de cruce SBX

La implementación más utilizada del operador SBX se encuentra integrada en el NSGA-II publicada en ([?]). Este procedimiento está descrito en el algoritmo 2, donde destacan dos aspectos clave. El primero está relacionado con la similitud entre los individuos padre y los hijos (líneas 22 y 23). En dicha implementación los valores de las soluciones hijas son heredadas directamente de las soluciones padre con una probabilidad fija igual a 0.5, mientras que el resto de las variables son modificadas mediante la distribución de probabilidad propia del SBX. En consecuencia la similitud que existe entre los padres y los hijos depende del número de variables que se consideran en el problema de optimización, pues el incremento de la dimensionalidad involucra la creación de soluciones más distantes.

El segundo punto clave, que nunca ha sido analizado en detalle, está relacionado con el conjunto de reflexiones que se realizan en las líneas 18 - 20. Después de generar los dos valores que deben ser heredados en los dos hijos, dichos valores son intercambiados entre sí con una probabilidad fija del 0.5. En consecuencia, cada vez que las variables son intercambiadas se realiza una reflexión, que puede inducir grandes distancias entre los padres y los hijos, a pesar de que el SBX es considerado como un cruce centrado en los padres. La parte izquierda de la Figura 5 ilustra este comportamiento en el operador SBX para dos y tres variables. En esta figura, los padres están identificados con dos puntos rojos, y se ejecutó el operador SBX diez mil veces. Cada uno de los puntos de color negro es una solución hija, con lo que esta figura ilustra las zonas en las que se tienden a generar a los hijos. Se puede ver que los valores de cada variable siempre están cercanos a uno de los valores asociados a las variables de los padres. Sin embargo, debido al proceso de intercambio de valores, las soluciones hijas no siempre se encuentran cercanas a los padres. Particularmente, en el caso de dos dimensiones mostrado, se crean soluciones en la esquina superior izquierda y en la esquina inferior derecha, mientras que los padres están es las esquinas opuestas. A medida que aumenta el número de dimensiones d, la probabilidad de que siempre o nunca haya intercambios, y que por tanto, la nueva solución esté cercana a uno de los padres es $k^d + (1-k)^d$, donde k es la probabilidad de realizar un intercambio, con lo que se produce un decremento exponencial respecto al número de dimensiones. En consecuencia, las reflexiones provocan un alto grado de exploración. En algunos MOPs con alta dimensionalidad en el espacio de las variables, esto podría representar un inconveniente porque las reflexiones localizan soluciones hijas en cada esquina del hipercubo mínimo que contiene a las soluciones padre, lo que significa que el operador original podría inducir un nivel muy bajo de intensificación.

El inconveniente relacionado con las reflexiones puede ser manejado parcialmente implementando restricciones de emparejamiento que traten de cruzar exclusivamente a soluciones similares. Bajo esta condiciones, el hipercubo sería de menor tamaño, con lo que se induciría un mayor grado de intensificación. Esta puede ser una de las razones por las que el MOEA/D, que incorpora restricciones de apareamiento, ha sido capaz de resolver muchos problemas de forma exitosa. En este trabajo, se trata de resolver esta problemática eliminando el intercambio de variables, así como realizando otras modificaciones en el SBX que se describen en la siguiente sección. La ventaja de esta segunda alternativa es que se puede incorporar de forma sencilla en cualquier MOEA.

This section is devoted to review some of the most important works that are highly related to the research presented in this paper. First, the most important paradigms are defined. Thereafter, some relevant classifications of crossover operators are introduced. Finally, the popular operator, which is used extensively in this paper, is discussed.

4.3. Multi-objective Evolutionary Algorithms

In the last years, a large number of following different design principles have been devised. In order to better classify them, several taxonomies have been proposed [?]. Attending to the principles of design, can be based on Pareto dominance, indicators and/or decomposition [?]. All of them have quite competitive representatives, so in this paper belonging to the different groups are taken into account. Particularly, the experimental validation has been carried out by including the Non-Dominated Sorting Genetic Algorithm (NSGA-II) [?], the MOEA based on Decomposition () [?], and the S-Metric Selection Evolutionary Multi-objective Optimization Algorithm () [?]. They are representative methods of the domination-based, decomposition-based and indicator-based paradigms, respectively. The following subsections briefly describe each one of these paradigms and introduce the selected methods.

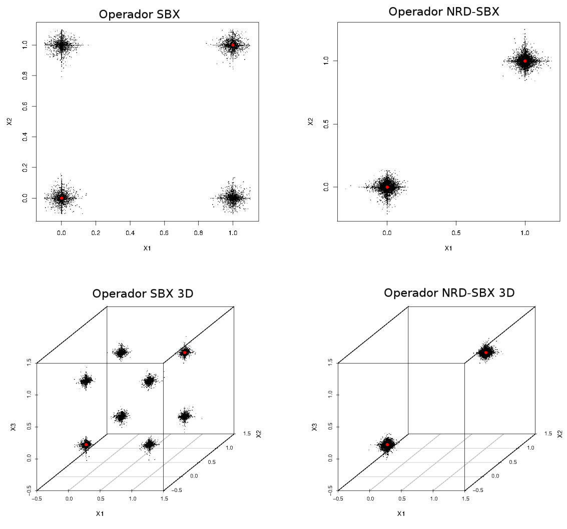

Figura 4 En la columna izquierda se presenta la simulación del operador SBX y en la columna de la derecha se presenta el operador NRD-SBX, ambas distribuciones con un índice de distribución de 20.

4.3.1. Domination Based MOEAs - NSGA-II

One of the most recognized paradigms is the domination based approach. belonging to this category are based on the application of the dominance relation to design different components of the EAs, specially the selection phase. Given that the dominance relation does not inherently promotes diversity in the objective space, auxiliary techniques such as niching, crowding and/or clustering are usually integrated to obtain an acceptable spread and diversity in the objective space. A critical drawback of methods based on the dominance relation is its scalability in terms of the dimensionality of the objective space. In fact the selection pressure is substantially reduced as the number of objectives increases. Although some strategies have been developed to deal with this issue [**?**] it remains as an important drawback for this kind of algorithms.

One of the most popular techniques of this group is the . This algorithm [**?**] considers a special selection operator based on non-dominated sorting and crowding. Non-dominated sorting is used to provide convergence to the Pareto front whereas crowding promotes the preservation of diversity in the objective space.

4.3.2. Decomposition Based MOEAs - MOEA/D

Decomposition-based [**?**] transform a in a set of single-objective optimization problems that are tackled simultaneously. This transformation can be achieved through several approaches. The most popular of them is applying a weighted Tchebycheff function, thus requiring a set of well distributed weights to attain well-spread solutions. An important drawback of this kind of approaches is related to the dependency between the Pareto front geometry and the weights required to attain proper solutions.

[?] is a recently designed decomposition-based . It includes several features such as problem decomposition, weighted aggregation of objectives and mating restrictions based on neighborhood definitions. Particularly, the neighborhoods are considered in the mating selection. A popular variant of is the , which uses the DE operators [?] and the polynomial mutation operator [?] in the reproduction phase. Additionally, it has two extra mechanisms for maintaining the population diversity [?].

4.3.3. Indicator Based MOEAs - SMS-EMOA

In multi-objective optimization several quality indicators have been developed to compare the performance of MOEAs. Since these indicators measure the quality of the approximations attained by , a paradigm based on the application of these indicators was proposed. Particularly, the indicators replace the Pareto dominance relation with the aim of guiding the optimization process. Among the different indicators, hypervolume is a widely accepted Pareto-compliance quality indicator [?]. One of the main advantages of these algorithms is that indicators usually take into account both the quality and diversity of the solutions, so no additional mechanisms to preserve diversity are required.

A popular and extensively used indicator-based algorithm is the [?]. This algorithm might be considered as hybrid, since it involves both indicators and Pareto dominance concepts. Essentially, it integrates the non-dominated sorting method with the use of the hypervolume metric. Thus, uses the hypervolume as a density estimator which results in a computationally expensive task. Particularly, the replacement phase erases the individual of the worst ranked front with the minimum contribution to the hypervolume. Taking into account the promising behavior of , it has been used in our experimental validation.

4.4. Crossover operators

The crossover operators are designed to generate offspring solutions using information of the parent solutions. They combine features of two or more parent solutions to generate new candidate solutions. Since several crossover operators have been proposed, some taxonomies have also been provided. The taxonomies are based on features such as the location of new generated solutions or the kinds of relations among the variables.

A popular taxonomy classifies crossover operators into variable-wise operators and vector-wise operators. In the variable wise category, each variable from parent solutions is recombined independently with a certain pre-specified

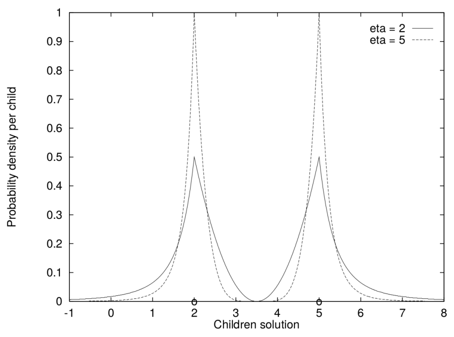

Figura 5 Probability density function of the operator with indexes of distribution 2 and 5. The parents are located in 2 and 5 respectively.

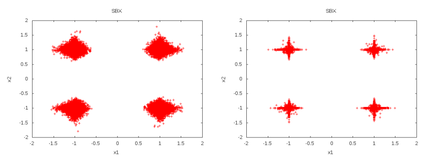

Figura 6 Simulations of the operator with a distribution index equal to 20. Parents are located in $P_1 = (-1{,}0, -1{,}0)$ and $P_2 = (1{,}0, 1{,}0)$. The left simulation corresponds to a probability of altering a variable (δ_1 in Algorithm 2) equal to 1,0 and in the right it corresponds to 0,1.

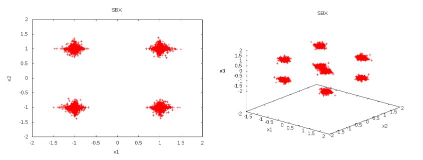

Figura 7 Simulations of the operator with a distribution index equal to 20. Parents are located in $P_1 = (-1{,}0, -1{,}0)$ and $P_2 = (1{,}0, 1{,}0)$ and $P_1 = (-1{,}0, -1{,}0, -1{,}0)$ and $P_2 = (1{,}0, 1{,}0, 1{,}0)$ for two and three variables respectively.

probability to create new values. These operators are specially suitable to deal with separable problems. Some operators belonging to this category are the Blend Crossover () [?], and the [?]. Alternatively, the vector-wise recombination operators are designed to take into account the linkage among variables. They usually perform a linear combination of the variable vectors. Some operators belonging to this category are the Unimodal Normally Distributed Crossover () [?], and the simplex crossover () [?]. Additionally, crossover operators can be classified as Parent-Centric and Mean-Centric [?]. In Parent-Centric operators, children solutions are created around one of the parent solutions, whereas in Mean-Centric operators, children solutions tend to be created mostly around the mean of the participating parent solutions. Among the crossover operators, is probably the most frequently used operator, so this research focuses on this crossover.

4.4.1. Simulated Binary Crossover - SBX

The reproduction operators are one of the most relevant components that influence the search process of EAs. Specifically, the crossover and mutation operators are highly related with the diversity of solutions. Hence, the quality of solutions are highly affected by the applied operators.

Simulated Binary Crossover () [?] is probably the most popular operator for continuous domains and most have been extensively tested with such an operator [?,?]. is classified as Parent-Centric, meaning that two children values (c_1 and c_2) are created around the parent values (p_1 and p_2). The process of generating the children values is based on a probability distribution. This distribution controls the spread factor $\beta = |c_1 - c_2|/|p_1 - p_2|$ defined as the ratio between the spread of the children values and parent values. In order to define this density function a distribution index η_c (a user-defined control parameter) alters the exploration capability of the operator. Specifically, a small index induces a larger probability of building children values distant to the parent values, whereas high indexes tend to create solutions very similar to the parents as is shown in Figure 3.

The probability distribution to create an offspring value is defined as a function of $\beta \in [0, \infty]$ as follows:

$$P(\beta) = \begin{cases} 0{,}5(\eta_c + 1)\beta^{\eta_c}, & \text{if} \quad \beta \leq 1 \\ 0{,}5(\eta_c + 1)\frac{1}{\beta^{\eta_c+2}}, & \text{otherwise} \end{cases} \tag{12}$$

Based in the mean-preserving property of children values and parent values, has the following properties:

- Both offspring values are equi-distant from parent values.
- There exist a non-zero probability to create offspring solutions in the entire feasible space from any two parent values.
- The overall probability of creating a pair of offspring values within the range of parent values is identical to the overall probability of creating two offspring values outside the range of parent values.

Therefore, considering two participating parent values (p_1 and p_2), two offspring values (c_1 and c_2) can be created as linear combination of parent values with a uniform random number $u \in [0, 1]$, as follows:

$$\begin{aligned} c_1 &= 0{,}5(1 + \beta(u))p_1 + 0{,}5(1 - \beta(u))p_2 \\ c_2 &= 0{,}5(1 - \beta(u))p_1 + 0{,}5(1 + \beta(u))p_2 \end{aligned} \tag{13}$$

The parameter $\beta(u)$ depends on the random number u, as follows:

$$\beta(u) = \begin{cases} (2u)^{\frac{1}{\eta_c+1}}, & \text{if} \quad u \leq 0{,}5, \\ (\frac{1}{2(1-u)})^{\frac{1}{\eta_c+1}}, & \text{otherwise} \end{cases} \tag{14}$$

Algorithm 4 Simulated Binary Crossover ()

```
Input: Parents ($P_1, P_2$), Distribution index ($\eta_c$), Crossover probability ($P_c$).
Output: Children ($C_1, C_2$).
if $U[0,1] \leq P_c$ then
    for each variable d do
        if $U[0,1] \leq \delta_1$ then
            Generate $C_{1,d}$ with Equations (11) and (12).
            Generate $C_{2,d}$ with Equations (11) and (13).
            if $U[0,1] \leq (1-\delta_2)$ then
                Swap $C_{1,d}$ with $C_{2,d}$.
        else
            $C_{1,d} = P_{1,d}$.
            $C_{2,d} = P_{2,d}$.
else
    $C_1 = P_1$.
    $C_2 = P_2$.
```

The above equation considers an optimization problem with no variable bounds. In most practical problems, each variable is bounded within a lower and upper bound. Thus, the modification of the probability distribution shown in Equation (11) was proposed [?] with the aim of taking into account such bounds. This last variant is extensively used nowadays.

$$\beta(u) = \begin{cases} (2u(1-\gamma))^{\frac{1}{\eta_c+1}}, & \text{if} \quad u \leq 0{,}5/(1-\gamma), \\ (\frac{1}{2(1-u(1-\gamma))})^{\frac{1}{\eta_c+1}}, & \text{otherwise} \end{cases} \tag{15}$$

$$c_1 = 0{,}5(1+\beta(u))p_1 + 0{,}5(1-\beta(u))p_2 \tag{16}$$

$$c_2 = 0{,}5(1+\beta(u))p_1 + 0{,}5(1-\beta(u))p_2 \tag{17}$$

In this case, the child c_1 which is nearest to p_1 is calculated according to the Equation (12). Considering that $p_1 < p_2$ and with a lower bound equal to a, $\gamma = 1/(\alpha^{\eta_c+1})$, where $\alpha = 1 + (p_1 - a)/(p_2 - p_1)$. Similarly, the second child c_2 is computed with $\alpha = 1 + (b - p_2)/(p_2 - p_1)$, where b correspond to the upper bound. Then, the second child is computed as is indicated in Equation (13).

Note that as reported in [?] several extensions of the to problems with multiple variables might be provided. Authors considered a simple strategy for choosing the variables to cross [?]. Specifically, each variable is crossed with probability 0.5, following the principles of uniform crossover. Authors recognized the important implications on the linkage among variables of such decisions. In any case, this is the most typical way of applying in problems with multiple variables nowadays.

4.4.2. Implementation and analyses of SBX operator

This section discusses some of the main characteristics of the most currently used implementation of the operator for problems with multiple variables. Essentially, three key components that might affect its performance are discussed. Firstly, as already mentioned it alters each variable with a fixed probability equal to 0,5. If this probability value is increased, the children tend

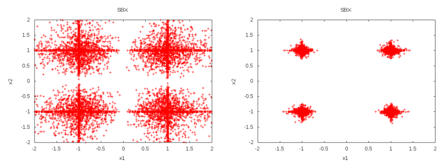

Figura 8 Simulation of the operator sampling 10, 000 children values, the parents are located in $P_1 = (-1{,}0, -1{,}0)$ and $P_2 = (1{,}0, 1{,}0)$. The left and right are with a distribution index of 2 and 20 respectively.

to be more distant to the parents, since in average more variables are modified simultaneously. In separable problems, altering only one variable might be adequate. However, for non-separable problems altering several variables simultaneously seems more promising. The implications of varying this probability is illustrated in Figure 4, where a problem with two variables is taken into account. In the right side a low probability is used and it provokes a bias to explore by keeping some values intact creating a figure similar to a cross in the two-dimensional case. This feature might be suitable for separable problems. Alternatively, the left side shows that when using a high probability this bias disappears, which could be more suitable for non-separable problems. Note that this probability is somewhat related with the distribution index in the sense that both have a direct effect on the similarity between parents and children.

The second key issue is that after generating the two child values with the distribution, such values are interchanged with a fixed probability that is usually set to 0,5, i.e. the value closer to parent p_1 is not always inherited by c_1. This is a feature that is not usually discussed but it is important for the obtained performance. In some contexts this probability is known as "Variable uniform crossover probability" [**?**] or "Discrete Recombination" [**?**]. Since in multi-objective optimization more diversity is maintained these swaps might produce a high disruptive operator. In fact, in some sense due to this action it is not so clear that can be categorized as a parent-centric operator. These interchanges between the children has the effect of performing multiple "reflections" in the search space. When increasing the dimensions of the decision variables the number of regions covered increases exponentially as is illustrated in Figure 5 where cases with two and three decision variables are taken into account. Note also that this feature has a considerable effect on the distance between parents and offspring.

Finally, the last component is the distribution index, which is probably the most well known feature of the . A low index results in greater exploration levels. In fact, a distribution index equal to one has a similar effect to the Fuzzy Recombination Operator [**?**]. The effect of applying different indexes is illustrated in Figure 6 where the left side considers a low index value whereas the right side takes into account a higher index value, which creates new candidate solutions that are more similar to the parents.

The implementation is shown in Algorithm 2. This pseudocode is based on the implementation that is integrated in the NSGA-II code published by Deb et al. [?] and which is the most popular variant nowadays. As an input it requires two parents (P_1 and P_2) and it creates two children (C_1 and C_2). The first and second key components commented previously correspond to the lines 5 and 8, respectively. As is usual, for the basic case, is configured with $\delta_1 = \delta_2 = 0{,}5$ and $\eta_c = 20$. It is important take into account that this implementation does not consider the dimension of the decision variables or the stopping criteria to set any of its internal parameters.

4.5. Propuesta

Based on the previous analyses and with the aim of inducing an appropriate balance between exploration and intensification, the following modifications are proposed. First, the probability to modify a variable (δ_1) is dynamically modified during the execution. The rationality behind this modification is to increase the exploration capability in the initial stages by altering simultaneously several variables and then, as the evolution proceeds reduce the number of variables that are modified. The value of δ_1 is changed in base of a linear decreasing model, where initially it is fixed to 1,0 and then it is decreased so that at the half of total generations is equal to 0,5. This last value is maintained until the end of the execution, i.e. from the half of the execution it behaves as the traditional implementation. Equation (14) is the one used to set the value of δ_1, where $G_{Elapsed}$ is the current generation and G_{End} is the total number of generations.

In a similar way, the second change is related to the probability of performing reflections ($1 - \delta_2$). In this case δ_2 is also updated as in Equation (14), meaning that the probability of performing a reflection increases from 0,0 to 0,5 during the execution. This modification is performed with the aim of avoiding the disruptive behavior of interchanging the variables at the first generations because this might result in very drastic modifications. Once that the individuals converge to certain degree it might make more sense to perform such reflections. Thus, this probability is increased to 0,5 which is the value used in the standard implementation of .

$$\delta_1 = \delta_2 = max\left(0{,}5, 1{,}0 - \frac{G_{Elapsed}}{G_{End}}\right) \tag{18}$$

Finally, the distribution index is also changed during the execution. At the first stages a low distribution index is induced with the aim of increasing the exploration capabilities of . Then, it is linearly incremented which has the effect of closing the distribution curve, meaning that more intensification is promoted. The linear increment is governed by Equation (15), meaning that the distribution index is altered from 2 to 22. Note that modifications similar to this last one have been explored previously [?], [?].

Cuadro 5 References points for the HV indicator

Instances	**Reference Point**
WFG1-WFG9	$[2{,}1, ..., 2m + 0{,}1]$
DTLZ 1, 2, 4	$[1{,}1, ..., 1{,}1]$
DTLZ 3, 5, 6	$[3, ..., 3]$
DTLZ7	$[1{,}1, ..., 1{,}1, 2m]$
UF 1-10	$[2, ..., 2]$

Cuadro 6 Statistical Information of Metrics with two objectives

	NSGA-II						MOEA/D						SMS-EMOA					
	1	2	3	4	5	DE	1	2	3	4	5	DE	1	2	3	4	5	DE
Average HV	0.88	0.90	0.90	0.91	0.93	**0.94**	0.87	0.87	0.87	0.90	**0.91**	**0.91**	0.88	0.89	0.87	0.91	0.92	**0.93**
Average IGD+	0.12	0.09	0.11	0.07	0.06	**0.05**	0.14	0.12	0.14	0.09	0.08	**0.07**	0.13	0.11	0.14	0.08	0.07	**0.05**

Cuadro 7 Statistical Information of Metrics with three objectives

	NSGA-II						MOEA/D						SMS-EMOA					
	1	2	3	4	5	DE	1	2	3	4	5	DE	1	2	3	4	5	DE
Average HV	**0.87**	0.84	**0.87**	**0.87**	**0.87**	0.85	0.84	0.84	0.84	**0.86**	**0.86**	0.85	0.90	0.89	0.88	**0.91**	**0.91**	**0.91**
Average IGD+	0.13	0.16	0.13	**0.12**	**0.12**	0.13	0.15	0.14	0.15	**0.11**	**0.11**	0.13	0.11	0.11	0.13	**0.09**	**0.09**	0.13

Cuadro 8 Summary of Statistical Tests

NSGA-II															
	1			2			3			4			5		
	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$
HV-2obj	16	29	47	6	61	25	28	19	45	31	23	38	**54**	3	35
HV-3obj	15	19	42	12	50	14	17	15	44	**33**	10	33	26	9	41
IGD-2obj	14	30	48	4	60	28	25	17	50	33	19	40	**52**	2	38
IGD-3obj	14	18	44	13	44	19	18	15	43	**33**	15	28	23	9	44

MOEA/D															
	1			2			3			4			5		
	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$
HV-2obj	15	33	44	10	60	22	25	26	41	39	18	35	**57**	9	26
HV-3obj	10	22	44	12	39	25	11	19	46	24	10	42	**38**	5	33
IGD-2obj	16	31	45	9	60	23	23	27	42	37	17	38	**57**	7	28
IGD-3obj	12	22	42	13	43	20	13	24	39	30	9	37	**40**	10	26

SMS-EMOA															
	1			2			3			4			5		
	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$	$\uparrow$	$\downarrow$	$\longleftrightarrow$
HV-2obj	9	35	48	7	43	42	16	31	45	41	9	42	**53**	8	31
HV-3obj	7	21	48	9	35	32	13	21	42	27	6	43	**31**	4	41
IGD-2obj	10	34	48	15	48	29	12	33	47	41	12	39	**55**	6	31
IGD-3obj	8	20	48	13	30	33	9	19	48	22	5	49	**27**	5	44

$$\eta_c = 2 + 20 \times \left(\frac{G_{Elapsed}}{G_{End}} \right) \tag{19}$$

4.6. Resultados

This section is devoted to analyze the results obtained with the dynamic variants of (). The novel crossover operator was integrated with , and . First, three variants that alter only one of each of the components previously discussed are analyzed. Then, a case that alters two of them simultaneously is taken into account. The WFG [?], DTLZ [?] and UF [?] test problems have been used for our purpose. Our experimental validation also includes the variant of Differential Evolution known as DEMO [?] with the aim of comparing our extension of with other well-known operators.

Given that all the methods are stochastic algorithms, each execution was repeated 35 times with different seeds. The common configuration in all of them was the following: the stopping criterion was set to $25,000$ generations, the population size was fixed to 100, WFG test problems were configured with two and three objectives, and 24 variables were considered, where 20 of them are distance parameters and 4 of them are position parameters. In the case of the DTLZ test instances, the number of decision variables were set to $n = M+r-1$, where $r = \{5, 10, 20\}$ for DTLZ1, DTLZ2 to DTLZ6 and DTLZ7 respectively, as is suggested in [?]. In the UF benchmark set the number of decision variables were set to 10. Finally, the polynomial mutation was used with a mutation probability equal to $1/n$ and with a distribution index equal to 50, whereas for the cases that used the , the crossover probability was set to 0,9 and the distribution index was set to 20. The additional parameterization of each algorithm was as follows:

- **DEMO**: CR = 0.3 and F = 0.5.
- **SMS-EMOA**: offset = 100.
- **MOEA/D**: size of neighborhood = 10, max updates by sub-problem (nr) = 2 and δ = 0,9.

In order to compare the fronts obtained by the different methods the normalized hypervolume (HV) and IGD+ was taken into account. The reference points used for the hypervolume indicator are shown in the Table 1 and are similar to the ones used in [?, ?].

In order to statistically compare the results (IGD+ and HV values), the following statistical tests were performed. First a Shapiro-Wilk test was performed to check whatever or not the values of the results followed a Gaussian distribution. If, so, the Levene test was used to check for the homogeneity of the variances. If samples had equal variance, an ANOVA test was done; if not, a Welch test was performed. For non-Gaussian distributions, the non-parametric Kruskal-Wallis test was used to test whether samples are drawn from the same distribution. An algorithm X is said to win algorithm Y when the differences between them are statistically significant, and the mean and median obtained by X are higher (in HV) or lower (in IGD+) than the mean and median achieved by Y.

4.7. Analysis of isolated components

In this section we discuss about the independent effect of each component that is dynamically modified. The effect of each component is analyzed through four cases, based in the Algorithm 2. Each case is described as follows:

- **Case 1**: The standard SBX operator where $\delta_1 = \delta_2 = 0{,}5$ and $\eta_c = 20$.
- **Case 2**: The value δ_1 is updated according to Equation (14), $\delta_2 = 0{,}5$ and $\eta_c = 20$.
- **Case 3**: The value δ_2 is updated according to Equation (14), $\delta_1 = 0{,}5$ and $\eta_c = 20$.
- **Case 4**: The distribution index is updated according to Equation (15), $\delta_1 = \delta_2 = 0{,}5$.

In order to analyze the performance of each Case (Case 5 is discussed later), Tables 2 and 3 shows information about the Normalized Hyper-volume (HV) [**?**] and about the Inverted Generational Distance Plus (IGD+) [**?**]. Specifically, the mean of the HV and IGD+ for all considered problems are shown for two and three objectives. It is clear that case 4 outperforms case 1, case 2 and case 3 both with two and three objectives in all the tested algorithms. Therefore, increasing the distribution index during the execution seems to be the most beneficial action. This occurs because the initially open distribution curve leads to a higher degree of exploration, whereas as the evolution proceeds more intensification is promoted. On the other hand, case 2 presented a lower performance than case 1 when taking into account three objectives. Thus, it seems that altering almost all the variables convert the new approach into a too disruptive operator. Perhaps, altering δ_1 in a different way might provide better results, but this is left as a future work.

Previous analyses are only based on the mean obtained for all the problems. However, depending on the problem the performance might vary. This is analyzed in the following section. Additionally, more detailed results are available[2].

4.8. Simultaneous modification of several components

Based on the previously discussed results, a variant of the is proposed where the case 3 and case 4 are mixed, i.e. both δ_2 and the distribution index are updated dynamically. Since case 2 did not report significant benefits, the updating mechanism for δ_1 was discarded. Specifically in our case 5, Algorithm 2 is configured as follows. The parameter δ_1 is fixed to 0,5, i.e. in a similar way than the standard . Following the case 3, δ_2 is updated according to Equation (14). Finally, according to case 4 η_c is updated in base of Equation (15).

Attending to the mean HV and IGD+ obtained by the case 5 (see Tables 2 and 3) it is clear that integrating case 3 and case 4 is beneficial. The advantages are clearer in the case of two objectives, whereas in the case of three objectives, case 4 and case 5 are similar in terms of mean performance. Moreover, results attained with case 5 are superior to the ones obtained with DE in three objectives, whereas when using the traditional results deteriorate.

[2] https://github.com/joelchaconcastillo/SBX_CEC2018.

Thus, when properly configuring a results similar or superior to DEMO could be obtained.

Finally, since previous analyses only consider the mean among all the benchmark problems, an additional analyses was developed to better understand the contributions of the different cases. Particularly, pair-wise statistical tests among all the five cases that consider and were carried out. This was performed independently for , and . Results of these statistical tests are shown in Table 4. For each algorithm and case, the column "↑" reports the number of comparisons where the statistical tests confirmed the superiority of the corresponding case, whereas the column "↓" reports the number of cases where it was inferior and "⟷" indicates the number of comparisons where differences were not statistically significant. The advantages of case 5 are quite clear. Only in the case of with three objectives, case 4 could outperform the results obtained by case 5. Thus, by properly combining several dynamic modification, results can be improved further. Moreover, results confirm the advantages of our proposals when compared to the standard (case 1). The only case that is not clearly superior to the standard is the case number 2, as it was previously discussed.

5. Conclusiones

Referencias

1. J. Montgomery, S. Chen, in *Evolutionary Computation (CEC), 2010 IEEE Congress on* (IEEE, 2010), pp. 1–8