A Multi-Region Differential Evolution Approach for Continuous Optimization Problems

Guillermo Leguizamón
LIDIC - Universidad Nacional de San Luis,
San Luis, ARGENTINA
and UMI LAFMIA 3175 CNRS CINVESTAV-IPN
Departamento de Computación
Av. IPN No. 2508. Col. San Pedro Zacatenco
México D.F. 07300, MÉXICO,
email: legui@unsl.edu.ar

Carlos A. Coello Coello
CINVESTAV-IPN
(Evolutionary Computation Group)
and UMI LAFMIA 3175 CNRS at CINVESTAV-IPN
Departamento de Computación
Av. IPN No. 2508. Col. San Pedro Zacatenco
México D.F. 07300, MÉXICO
email: ccoello@cs.cinvestav.mx

***Abstract*—This paper presents a Multi-Region Differential Evolution (MRDE) algorithm as an extension of a classical version of differential evolution (DE) (i.e., as an extension of DE/rand-to-best/1/exp). MRDE is designed to simultaneously search on different and evenly distributed sub-regions on the whole search space. The number and extent of the search regions change during the execution of the algorithm, in such a way that, at the final stage of the evolutionary process, only one region remains (i.e., the whole search space). Our proposed MRDE is compared with respect to the classical DE algorithm on a set of well-known benchmark problems. The results achieved show enough evidence of the benefits of distributing the population of vectors when dealing with large-scale optimization problems.**

I. Introduction

Differential Evolution (DE) is a metaheuristic that was originally proposed in the mid-1990s by Kenneth Price and Rainer Storn [11], [9], [10], [12]. More recent perspectives about DE and its different variations and applications are provided in Feoktistov [5] and Chakraborty [3].

DE has been widely accepted as one of the most efficient algorithms for solving continuous optimization problems. In that regard, several recent research efforts have been devoted to use DE to solve large scale optimization problems. The previous CEC 2008 and CEC 2010 competitions included several DE-based proposals which were relatively successful when compared to other types of metaheuristics. For example, the proposals of Brest et al. [1] and Zamuda et al. [16] showed competitive results in the CEC 2008 competition. Brest et al. [1] presented a Self-Adaptive Differential Evolution algorithm (*jDEdynNP-F*) in which parameters F and C_r are self-adapted, together with the parameter NP (reduction on the population size). In addition, a mechanism for changing the sign of F is also incorporated. *jDEdynNP-F* reaches high quality solutions for the benchmark adopted. On the other hand, Zamuda et al. [16] designed *DEwSAcc*, a DE-based algorithm extended by a log-normal self-adaptation mechanism for its main parameters (i.e., F and C_r). *DEwSAcc* also includes a cooperative co-evolution mechanism aimed to decompose the dimensions of the problem. Although *DEwSAcc* was less competitive with respect to the other DE-based algorithm available at the CEC 2008 competition (*jDEdynNP-F*), its results were competitive when compared to those of the other algorithms that participated in the contest.

More recently (at the CEC 2010 contest) Brest et al. [2] presented *jDElsgo*, an extended version of *jDEdynNP-F*, which was developed by the same author in order to deal with large scale optimization problems. Wang et al. [15], for the same contest, presented a sequential DE algorithm enhanced by neighborhood search (*SDENS*). This algorithm involves two main steps: 1) from each individual, two trial vector are generated by applying, respectively, local and global neighborhood search, and 2) the fittest vector among the target one and the two trial vectors is selected to be part of the next generation. In addition, a DE with one array is used to promote convergence. Both, jDElsgo and SDENS achieved good quality solutions for the benchmark problems adopted. However, none of them outperformed the other algorithms presented at this contest.

Regarding the use of multi-region (or multi-trajectory) mechanisms to explore by regions the whole search space in continuous domains, we can mention the works of Tseng and Chen [13], Zhao et al. [17], and Hu et al. [6]. It is worth remarking that none of the above algorithms are based on DE. Tseng and Chen [13] presented a multiple trajectory search (MTS) algorithm which achieved the highest ranking in the CEC 2008 contest. MTS uses multiple agents that apply one of three candidate of iterated local search on different regions distributed according to orthogonal arrays. MTS also includes the concept of "foreground" solutions. These are high quality solutions found at earlier iterations and used as base solutions to find newer and better solutions in further iterations of the algorithm. Another interesting PSO based algorithm is DMS-PSO-SHS presented in the CEC 2010 contest by Zhao et al. [17]. DMS-PSO-SHS is an extension of DMS-PSO (see [19]), which implements a multiple trajectory search mechanism hybridized with sub-regional harmony search. The whole swarm is divided into a large number of sub-swarms (with dynamic size) to conform each one the population of the harmony search algorithm. The results from DMS-PSO-SHS showed to be highly competitive on the benchmark problems considered. Regarding the Ant Colony Optimization

(ACO) metaheuristic [4], an orthogonal search algorithm was presented by Hu et al. [6]. The algorithm, called Continuous Orthogonal Ant Colony (COAC), distributes a set of ants on different regions of the search according to an orthogonal array. To enhance diversity, COAC implements and adaptive regional radius mechanism by either expanding or shrinking the area of exploration of each ant. The performance of COAC was compared with two ACO-based algorithms (API and CACO) on a set of seventeen continuous optimization problems. Although the results obtained by COAC seem to be encouraging, it is not clear from the contents of the paper which was the dimensionality of the problems considered by the authors for their experimental study.

The rest of the paper is organized as follows. The next section describes a basic DE algorithm which will be the baseline algorithm of our proposal (MRDE) presented in Section III. Our experimental study is described in Section IV where the benchmark problems adopted are introduced, as well as a performance comparison between DE and MRDE. Finally, the conclusions and lines of future research are drawn in Section V.

II. Basics on DE

Following the early concepts from Price et al. [10], DE can be described as a population-based metaheuristic that evolves a set of NP vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_{NP}\}$ which undergo differential mutation and crossover operators as well as a one-to-one selection operator used to bias the search towards solutions of higher quality. More precisely, the DE algorithm includes the main following components:

i) **Population**
$P(t) = [\mathbf{x}_1(t), \ldots, \mathbf{x}_{NP}(t)]$ for $t = 0, 1, \ldots, t_{max}$ and $\mathbf{x}_i(t) = \{x_{i,1}(t), \ldots, x_{i,D}(t)\}$, where t indicates the current generation and D the number of problem parameters or dimensionality.

ii) **Initialization process**
It is usually achieved by generating a uniform random number subject to the corresponding lower and upper limits for each problem dimension:

$$x_{i,j}(t) = U[0,1) \cdot (u_j - l_j) + l_j$$

where l_j and u_j are, respectively, the lower and upper limits for dimension j and $U(0,1)$ returns a uniformly distributed random number in the range $[0,1)$, different for each parameter.

iii) **Differential Mutation**
To apply this operator a base vector $\mathbf{x}_{r_1}(t) \in P(t)$ is perturbed by a difference vector based mutation in order to generate the mutated vector $\mathbf{m}$ as follows:

$$\mathbf{m} = \mathbf{x}_{r_1}(t) + F \cdot (\mathbf{x}_{r_3}(t) - \mathbf{x}_{r_2}(t)), \qquad (1)$$

where r_1, r_2, and r_3 are three different indexes randomly chosen form the set $\{1, \ldots, NP\}$. There exist several alternatives to define eq. (1), e.g., changing the way in which r_1, r_2, and r_2 are selected or by using additional difference vectors for mutation, among others.

iv) **Crossover**
This operator which is aimed at enhancing the diversity, is a binary operator that mixes the resulting vector $\mathbf{m}$ from the mutation operator and the so-called target vector $\mathbf{x}_i(t)$. The resulting vector ($\mathbf{u}$) can be obtained as:

$$\mathbf{u} = (u_1, u_2, \ldots, u_D), \text{ where}$$

$$u_j = \begin{cases} m_j & \text{if } (U[0,1) \leq C_r) \\ x_{i,j}(t) & \text{otherwise.} \end{cases} \qquad (2)$$

$$\text{for } j \in \{1, \ldots, D\}.$$

Similarly to eq. (1), other alternatives are also possible for the crossover operator as described in [5], [10].

v) **Selection**
In order to bias the search towards high quality solutions, a simple one-to-one selection process is applied where the recently generated trial vector $\mathbf{u}$ competes against the target vector $\mathbf{x}_i$, i.e., the best performer vector regarding the objective function value survives for the next generation. Basically, the process is as simple as follows (for a minimization problem):

$$\mathbf{x}_i(t) = \begin{cases} \mathbf{u} & \text{if } eval(\mathbf{u}) \leq eval(\mathbf{x}_i) \\ \mathbf{x}_i(t) & \text{otherwise.} \end{cases} \qquad (3)$$

The above description corresponds to a classical DE algorithm. However, several other variants can be found in literature. The usual notation to indicate a particular variant is given by the expression DE/*x*/*y*/*z*, where *x* indicates the base vector (r_1 in eq. (1)), *y* represents the number of difference vectors considered (this also affects eq. (1)), and *z* denotes the crossover method (variant of eq. (3)). In our work, we adopted the variant DE/rand-to-best/1/exp which is extended to obtain our proposed algorithm MRDE as further explained in this paper. The general outline of DE/rand-to-best/1/exp is presented in Algorithm 1 which will be the base pseudo-code to present and better understand our proposal in the next section. The components of Algorithm 1 are self-explained: a) Population initialization (line 1), b) mutation operator according eq. (1) (lines 4 and 5), c) crossover (line 6), and finally d) one-to-one selection (lines 7-9).

III. The proposed Multi-region DE (MRDE)

Our proposed MRDE algorithm consists in evolving a set NP of vectors distributed in NR initial regions of NPR vectors each, i.e., $NP = NR \times NPR$. Before presenting the pseudo-code for MRDE, it is necessary explaing some basic ideas behind its design. The initialization process involves the determination of a set of NR anchor points. These so-called anchor points determine the position of the respective region regarding the whole search space. In our case, we adopted a Latin Hypercube Sampling to evenly distribute the anchor points and afterwards determine the respective lower and upper limits of each region. We have introduced a parameter Ext

Algorithm 1 General outline of DE/rand-to-best/1/exp for a minimization problem.

```
1: Init(x_1(0), ..., x_NP(0));
2: for t ∈ 1 : t_max do
3:    for i ∈ 1 : NP do
4:       Obtain(r_2,r_3); [r_1 = best_it]
5:       m ← MutateTargetVector(x_i(t), best_it, r_2, r_3, F);
6:       u ← Crossover(x_i(t), m, C_r);
7:       if (cost(u) < cost(x_i(t)) ) then
8:          x_i(t + 1) ← u; [replace the target by the trial
            vector]
9:       end if
10:   end for
11: end for
```

which defines the extent of search of each region around the corresponding anchor point. Figure 1 shows a 2-dimensional search space on which 6 anchor points have been sampled. It can be observed that 3 regions have been highlighted, which correspond to the anchor points p_1, p_4, and p_6. The region determined by anchor point p_4 covers on all dimensions to the maximum extent, whereas for anchor points p_1 and p_6, the extent of the regions is limited by the lower an upper limits originally specified for the corresponding dimension. More precisely, Ext indicates that the ratio of search is no greater than $Ext \times ((u_j - l_j)/NR)$ for $j \in \{1, \ldots, D\}$. For example, in Figure 1 the extent of search for each region is $Ext \approx 2$.

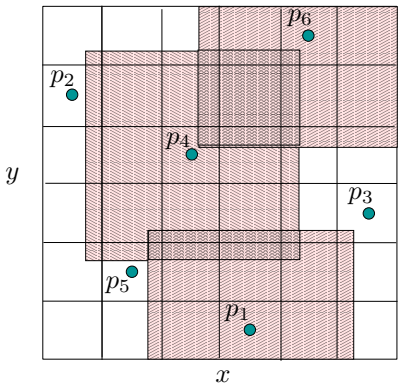

Fig. 1. Search space in $\mathbb{R}^2$ divided in six regions of search. The anchor points representing each region are a product of a Latin Hypercube sampling

After the process of determining the extent of each of the NR regions, an initialization process must take place to sample the NPR vectors on every region. The sampling of the vectors is also based on a Latin Hypercube to better distribute the points. However, any other mechanism could be used here as well as in the step of sampling the anchor points explained above. Figure 2 shows an example of 5 vectors sampled in the region determined by anchor point p_4 (first shown in Figure 1) with the corresponding lower and upper bounds.

Algorithm 2 describes the main steps involved in MRDE. It should be noticed that MRDE extends Algorithm 1 in

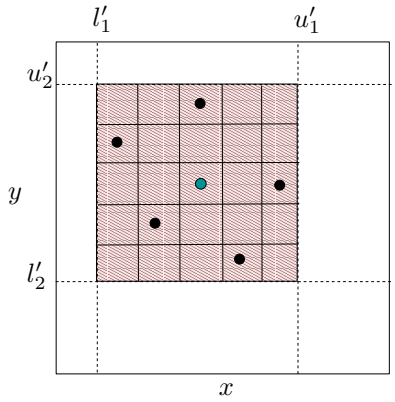

Fig. 2. Latin Hypercube sampling of five points in one of the six search regions previously determined in Fig. 1 where l'_1,l'_2 and u'_1, u'_2 are respectively the (temporary) lower and upper limits for dimensions 1 and 2 in the region determined by anchor point p_4.

order to keep fixed the basic components of a classical DE and only incorporate those components that are necessary to manage the multiple regions. This will let us better assess the potential benefits of the new components. Function `GetAnchorPoints()` generates NR anchor points spread over the whole search space. After that (line 2-5) the respective lower and upper limits are calculated for each region and each dimension (it should be noticed that those boundaries can be different for each dimension) and then, NPR points are sampled on each region. Thus, at the end of this iteration, NP vectors are sampled. The inner iteration found in Algorithm 1 is here decomposed in two iterations (line 12-21). The outer one is governed by the number of regions whereas the inner one controls the application of the basic operators in each region of NPR vectors. Notice that $best\_it^k$ (line 5) represents the best-so-far solution found in region k. In addition, a condition to proceed with the merge of regions is incorporated (lines 7-11). This step aims to increase the extent of the search area as the algorithm execution progresses. Function `Merge()` merges the regions based on the distance between the original anchor points. We considered to change from NR to $NR/2$ regions when the merge process takes place. Consequently, the number of vector in each region is duplicated at this point. As will explained in Section IV, this mechanism works for $NR = 2^m$, for $m \in \mathbb{N}$.

IV. Experimental Study

This section presents an experimental study on the selected benchmark problems (Section IV-A) to assess the performance of our proposed algorithm. A preliminary experimental study was conducted to select the strategy to apply among the ten strategies originally implemented in the classical DE source code [8] where five strategies apply 'exp' crossover and five apply 'bin' crossover. The results on the benchmark problems ($D = 500$ was used at this stage) showed that DE/*/*/exp outperform DE/*/*/bin variants. Particularly, DE/rand-to-best/1/exp showed an acceptable performance along the whole set of problems. For that reason,

Algorithm 2 General outline of MRDE/rand-to-best/1/exp for a minimization problem. It must be noticed that NP (in Algorithm 1) is equivalent to $NPR \times NR$ in this algorithm.

GetAnchorPointsLHS(NR);
for $k \in 1 : NR$ **do**
  (L^k, U^k) $\leftarrow$ GetLowerUpper(k, Ext);
  Init_LHS($\mathbf{x}_1^k(t), \ldots, \mathbf{x}_{NPR}^k(t)$); [$k$ denotes a region number]
end for
for $t \in 1 : t_{max}$ **do**
  if (merge condition is TRUE) **then**
    Merge(P);
    $NR \leftarrow NR/2$;
    $NPR = NPR * 2$;
  end if
  for $k \in 1 : NR$ **do**
    for $i \in 1 : NPR$ **do**
      Obtain(r_2,r_3);
      $\mathbf{m}$ $\leftarrow$MutateTargetVector($\mathbf{x}_i(t)$, $best\_it^k$, r_2, r_3, F);
      $\mathbf{u}$ $\leftarrow$Crossover($\mathbf{x}_i(t)$, $\mathbf{m}$, C_r);
      if (cost($\mathbf{m}$) $<$ cost($\mathbf{x}_i(t)$)) **then**
        $\mathbf{x}_i(t+1) \leftarrow \mathbf{u}$; [replace]
      end if
    end for
  end for
end for

we have chosen this DE variant; however, our proposed MRDE could be applied and studied on any of the other De variants currently available. Indeed, as will be further indicated, several alternatives of the core DE algorithm can be considered for more advanced versions of MRDE. For the sake of simplicity, further references to classical DE/rand-to-best/1/exp will be referred to as simply DE. Their corresponding extensions will be called MRDE.

All the experiments reported in this paper were run on an Intel Pentium (R) 4, CPU 3.00Gz, and 1Gb RAM; OS Linux version 2.6.23.17-88.fc7 (Red Hat 4.1.2-27). DE (taken from [8]) and its extension MRDE were implemented in the C programming language.

A. Test Problems

We selected 11 test problems from the benchmark prepared for the "Special Issue of Soft Computing: A Fusion of Foundations, Methodologies and Applications on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems" [7]. The problems represent a set of scalable functions for high-dimensional optimization. See Table IV-A for a description of these problems and their corresponding optimum values. Particularly, the objective of this special session was to bring to the research community newer and more challenging problems to assess current nature-inspired optimization algorithms as well as other, novel optimization algorithms.

TABLE I
A SUBSET OF THE TEST SUITE ORIGINALLY PROPOSED FOR THE "SPECIAL ISSUE OF SOFT COMPUTING: A FUSION OF FOUNDATIONS, METHODOLOGIES AND APPLICATIONS ON SCALABILITY OF EVOLUTIONARY ALGORITHMS AND OTHER METAHEURISTICS FOR LARGE SCALE CONTINUOUS OPTIMIZATION PROBLEMS" [7]

Benchmark Problems	Search Range	$f(\mathbf{x}^*)$
$F_1(\mathbf{x}) = \sum_{j=1}^{D} z_j + f\_bias_1, \mathbf{z} = \mathbf{x} - \mathbf{o}$ $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_1 = -450$	[-100,100]	-450
$F_2(\mathbf{x}) = max_j\{\lvert z_j \rvert, 1 \leq j \leq D\} + f\_bias_2, \mathbf{z} = \mathbf{x} - \mathbf{o}$ $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_2 = -450$	[-100,100]	-450
$F_3(\mathbf{x}) = \sum_{j=1}^{D-1}(100 \cdot (z_j^2 - z_j)^2 + (z_j - 1)^2) + f\_bias_3$, $\mathbf{z} = \mathbf{x} - \mathbf{o} + \mathbf{1}$; $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_3 = 390$	[-100,100]	390
$F_4(\mathbf{x}) = \sum_{j=1}^{D}(z_j^2 - 10 \cdot \cos(2\pi z_j) + 10) + f\_bias_4, \mathbf{z} = \mathbf{x} - \mathbf{o}$ $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_4 = -330$	[-5,5]	-330
$F_5(\mathbf{x}) = \sum_{j=1}^{D} \frac{z_j^2}{4000} - \prod_{j=1}^{D} \cos(\frac{z_j}{\sqrt{j}}) + 1 + f\_bias_5, \mathbf{z} = \mathbf{x} - \mathbf{o}$ $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_5 = -180$	[-600,600]	-180
$F_6(\mathbf{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D} z_j^2}) - \exp(\frac{1}{D}\sum_{j=1}^{D}\cos(2\pi z_j)) + 20 + f\_bias_6)$ $\mathbf{z} = \mathbf{x} - \mathbf{o}$; $\mathbf{o} = (o_1, o_2, \ldots, o_D)$; the shifted global optimum and $f\_bias_6 = -140$	[-32,32]	-140
$F_7(\mathbf{x}) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D}$	[-10,10]	0
$F_8(\mathbf{x}) = \sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$	[-65.536,65.636]	0
$F_9(\mathbf{x}) = (\sum_{i=1}^{D-1} f_{10}(x_i, x_{i+1})) + f_{10}(x_D, x_1)$ where $f_{10}(x, y) = (x^2 + y^2)^{0.25} \cdot (sin^2(50 \cdot (x^2 + y^2)^{0.1}) + 1)$	[-100,100]	0
$F_{10}(\mathbf{x}) = \sum_{i=1}^{D}(x_i^2 + 2x_{i+1}^2 - 0.3cos(3\pi x_i) - 0.4cos(4\pi x_{i+1}) + 0.7)$	[-15,15]	0
$F_{11}(\mathbf{x}) = \sum_{i=1}^{D-1}((x_i^2 + x_{i+1}^2)^{0.25} \cdot (sin^2(50(x_i^2 + x_{i+1}^2)^{0.1}) + 1))$	[-100,100]	0

B. DE vs MRDE

The parameter setting chosen for both algorithms is as follows: $F = 0.85$, $C_r = 0.9$, the maximum number of FES (function evaluations) was set to $D \times 5000$, where D is the problem's dimensionality. As we aim to determine the scaling properties of the algorithms, we considered $D \in \{500, 1000, 1500\}$. For MRDE, two additional parameters have to be considered: $NR = 4$ (number of initial regions) and $Ext = 1$ (extent of the region around the

TABLE II
COMPARISON OF DE AND MRDE BASED ON THE MANN-WHITNEY-WILCOXON TEST FOR DIMENSION $D = 500$. SYMBOLS '+', '-', AND '=' INDICATE, RESPECTIVELY, IF THERE EXIST SIGNIFICANT STATISTICAL DIFFERENCES OF MRDE WITH RESPECT TO DE.

Prob.	DE	MRDE	p-value	+/-/=
1	0 (0)	0 (0)	1	=
2	122.2123 (2.5196)	**44.4528** (1.1610)	1.41e-09	+
3	441.4288 (15.1254)	**412.2743** (55.9263)	6.96e-5	+
4	375.2653 (11.9744)	**305.1120** (17.0310)	1.41e-9	+
5	0 (0)	0 (0)	1	=
6	0 (0)	0 (0)	1	=
7	0 (0)	0 (0)	1	=
8	2.0256e+05 (0.1076e+05)	**0.3311e+05** (0.3070e+04)	1.4157e-9	+
9	0.2055 (0.0134)	**0.0141** (0.0086)	1.4157e-9	+
10	0 (0)	0 (0)	1	=
11	0.2015 0.0078	**0.0083** 0.0041	1.4157e-9	+

TABLE III
COMPARISON OF DE AND MRDE BASED ON THE MANN-WHITNEY-WILCOXON TEST FOR DIMENSION $D = 1000$. SYMBOLS '+', '-', AND '=' INDICATE, RESPECTIVELY, IF THERE EXIST SIGNIFICANT STATISTICAL DIFFERENCES OF MRDE WITH RESPECT TO DE.

Prob.	DE	MRDE	p-value	+/-/=
1	0 (0)	0 (0)	1	=
2	164.8760 (1.8935)	**68.5130** (1.1992)	1.4157e-09	+
3	986.4420 (15.8624)	**911.3352** (56.2462)	6.9619e-05	+
4	657.3982 (13.9735)	**640.2191** (32.4988)	1.4157e-09	+
5	0 (0)	0 (0)	1	=
6	0.0644 6.6442	**0** 0	8.8575e-05	+
7	0 (0)	0 (0)	1	=
8	8.8434e+05 (0.3460e+05)	**2.0081e+05** (1.312e+04)	1.4157e-09	+
9	0.4748 (0.0211)	**0.0240** (0.0111)	1.4157e-09	+
10	0 (0)	0 (0)	1	=
11	0.4955 (0.0144)	**0.0249** (0.0111)	3.1615e-09	+

TABLE IV
COMPARISON OF DE AND MRDE BASED ON THE MANN-WHITNEY-WILCOXON TEST FOR DIMENSION $D = 1500$. SYMBOLS '+', '-',AND '=' INDICATE, RESPECTIVELY, IF THERE EXIST SIGNIFICANT STATISTICAL DIFFERENCES OF MRDE WITH RESPECT TO DE.

Prob.	DE	MRDE	p-value	+/-/=
1	0 (0)	0 (0)	1	=
2	157.6307 (1.9033)	**67.6965** (1.0661)	1.4157e-09	+
3	930.7854 (7.5215)	**890.5772** (57.6894)	3.5302e-06	+
4	773.4208 (20.5676)	**647.0049** (31.5415)	1.4157e-09	+
5	0 (0)	0 (0)	 1	+ =
6	0.0236 (6.3681)	**0** (0)	1.4638e-08	+
7	0 (0)	0 (0)	1	=
8	8.6701e+05 (0.3280e+05)	**2.0245e+05** (0.1492e+05)	1.4157e-09	+ +
9	0.4722 (0.0157)	**0.0270** (0.0305)	1.4638e-08	+
10	0 (0)	0 (0)	1	=
11	0.4718 (0.0143)	0.0238 (0.0165)	1.4157e-09	+

corresponding anchor points). These parameter values were determined on a preliminary experimental study on the test suite problems with $D = 500$. Both algorithms were run 25 times under different random seeds on each problem and parameters setting. The "merge condition" in Algorithm 2 (line 7) will become TRUE when the current generation is equal to: $(t\%(t_{max}/(NR-1) == 0)))$. In this way, MRDE promotes first the search on the initial NR regions separately and then merges the regions until searching in only one region (the whole original search space). In our case, as $NR = 4$, MRDE will search on four regions during the first generations. After the first 'merge', it will search on two regions and, finally, in one region.

Tables II, III, and IV show, for each problem, and for the algorithms DE and MRDE, the median of the error measure out of 25 runs and the corresponding standard deviation (between parentheses). To assess the statistical significance based on the medians, the non-parametric Mann-Whitney-Wilcoxon test at a level of 95% of confidence was applied. Thus, a p-value < 0.05 indicates that, based on the median values, MRDE outperforms DE. An additional column was included to indicate, based on both, the p-values and medians, if MRDE outperforms DE (+) or DE outperforms MRDE (-), i.e., significant differences were found. On the other hand, both have similar performance (=), i.e., no significant differences were found.

First of all, is can be highlighted that DE performs fairly well on all the test problems adopted. It also can scale up for some problems (1,5,7, and 10) from dimension 500 to 1500 without losing the capacity to find the optimal solutions. However, we can observe, on the one hand, that MRDE also scales up to the optimum for the same problems previously mentioned and also for test problem 6. On the other hand, we see that MRDE was capable of improving at a different degree the results from DE in the remaining problems (indicated by the corresponding p-values and symbol "+"). The best improvement of MRDE was on test problem 8 for all dimensions, as well as on test problems 2, 9, and 11. Although on test problems 3 and 4, MRDE also outperforms DE, the extent of degree achieved is lesser. Finally, we can see the capacity of both algorithms to scale up from dimension 1000 to 1500 as the solutions found, when compared one-to-one for these two dimensionalities, have similar objective values.

Since it is important to analyze the performance of MRDE with respect to state-of-the-art algorithms for large continuous

TABLE V
COMPARISON OF MRDE AND SADE-MMTS [18] ON THE TEST PROBLEMS CONSIDERED HERE (WITH $D = 1000$) BASED ON THE MEDIAN VALUES REPORTED FOR EACH ALGORITHM.

Prob.	MRDE	SaDE-MMTS
1	0	0
2	6.8513e+01	**4.6414e+01**
3	9.1133e+02	**0**
4	6.4021e+02	**3.0655e+01**
5	0	0
6	0	0
7	0	0
8	2.0081e+05	**1.2564e+03**
9	**0.0240**	8.5612e+01
10	0	0
11	**0.0249**	9.3347e+01

optimization problems, we have chosen *SaDE-MMTS* [18], a recently proposed algorithm that also implements, as in our MRDE, a multi-trajectory approach to explore the search space. Although the comparison does not include any statistical test, it gives us some useful indications about the comparative performance of these algorithms. Table V displays the results (in terms of median values) for algorithms MRDE and SaDE-MMTS. On the one hand, a similar behavior of both algorithms can be observed for test problems 1, 5, 6, 7, and 10. On the other hand, SaDE-MMTS shows a superior performance on test problems 3 (the biggest difference), 4, 8, and finally 2, showing for the last one (test problem 2) the smallest difference when compared to the remaining problems of this group—i.e., problems 3, 4, 8 and 2. Interestingly, MRDE was able to outperform SaDE-MMTS in two problems, namely test problems 9 and 11. This short analysis show us a potential for MRDE (by improving our current version) to cope with large continuous optimization problems

To conclude this section we have selected two problems to show some features of the behavior of DE and MRDE with respect to the velocity of convergence and diversity of the population. The selected problems are test problem 8 (a difficult problem for our algorithms) and test problem 6, an easier one. To measure the degree of population diversity we adopted the following function [14]:

$$Div(T) = \frac{1}{N_{diag} \cdot NP} \sum_{i=1}^{NP} \sqrt{\sum_{j=1}^{D} (x_{i,j} - \bar{x}_j)^2} \qquad (4)$$

where N_{diag} is the length of the diagonal of the search space determined by the corresponding upper and lower limits for each decision variable, $x_{i,j}$ is the value at dimension j of solution i in $P(t)$, and $\bar{x}^j$ is the average of all the values in dimension j. Function Div (eq. (4)) returns a value in the range $[0.0, 0.5]$. Therefore, the higher the values returned, the more diversity is detected in the population of vectors.

Figures 3 and 4 display the convergence plots[1] for test problems 8 and 6 ($D = 500$) throughout the whole run. Test problem 6 is solved by the two algorithms; however, we can see that MRDE reaches the optimal solution earlier than DE. Although test problem 8 is not optimally solved by these algorithms, a similar situation as the one described above is also observed, i.e., MRDE shows an accelerated (but not premature) velocity of converge.

[1]It must be noted that, at each iteration, both DE and MRDE evaluate the same number of solutions.

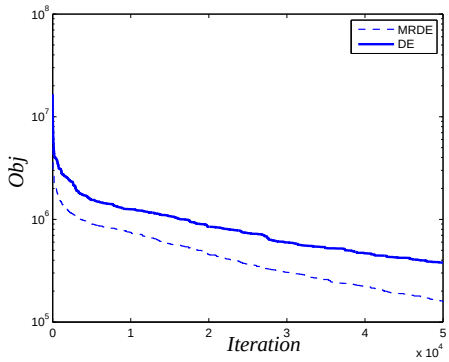

Fig. 3. Convergence plot (Objective Value) for test problem 8, $D = 500$.

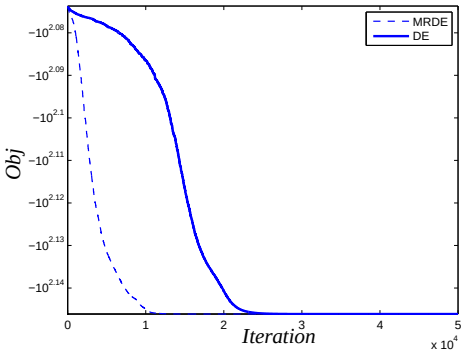

Fig. 4. Convergence plot (Objective Value) for test problem 6, $D = 500$.

The behavior previously observed can also be seen from the perspective of a diversity metric (eq. (4)). Figures 5 and 6 display for the same test problems (8 and 6) the behavior of DE and MRDE regarding the diversity metric. It can be clearly seen that MRDE has a faster loss of diversity. However, this does not mean performance degradation (Tables II, III, and IV support this claim). We hypothesize that the fast convergence of MRDE is because this algorithm rapidly finds promising regions of the search space that are exploited first by the solutions belonging to those regions. Thus, as the execution of MRDE progresses, the solutions (vectors) in unsuccessful regions displace the most successful ones when the merge process takes place.

V. CONCLUSIONS AND FUTURE WORK

We presented in this paper the algorithm MRDE, which is an extension of a classical DE algorithm, aimed to explore simultaneously on different regions of the search space. The results reported here show a clear potential of this alternative approach to solve large scale dimensional problems.

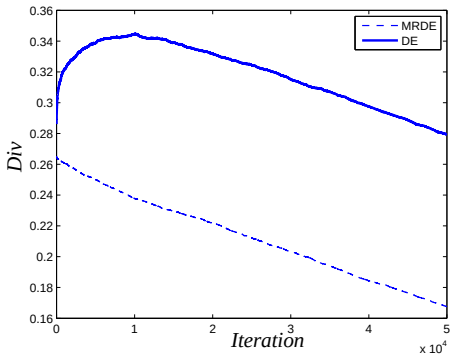

Fig. 5. Convergence plot (Diversity metric) for test problem 8, $D = 500$.

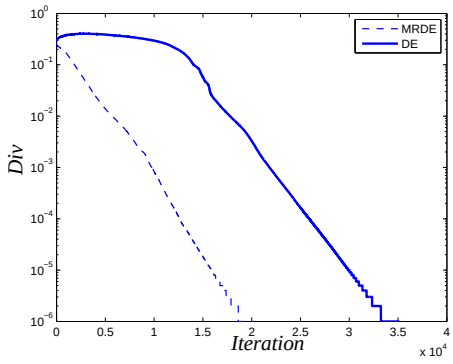

Fig. 6. Convergence plot (Diversity metric) for test problem 6, $D = 500$.

Nevertheless, we consider that a deeper investigation of this multi-trajectory approach is still necessary. In the case of MRDE, it is necessary an in-depth study and a further analysis of performance of the effect of NR and Ext in the algorithm as well as the characteristics observed in the velocity of convergence as well as the loss of diversity when varying them. In addition, we devise the development of an alternative MRDE approach by considering more advanced versions of the core DE algorithm as our search engine. We believe that such an approach could constitute a viable alternative for the efficient and effective solution of large scale dimensional problems.

Acknowledgments

The first author acknowledges the support from the UMI-LAFMIA 3175 CNRS at CINVESTAV-IPN and from the Universidad Nacional de San Luis, Argentina. The second author gratefully acknowledges support from CONACyT project no 103570.

References

[1] J. Brest, A. Zamuda, B. Boskovic, M. S. Maučec, and V. Zumer. High-dimensional real-parameter optimization using Self-Adaptive Differential Evolution algorithm with population size reduction. In *IEEE Congress on Evolutionary Computation*, pages 2032–2039, 2008.

[2] J. Brest, A. Zamuda, I. Fister, and M. S. Maučec. Large scale global optimization using self-adaptive differential evolution algorithm. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 3097–3104, Barcelona, Spain, July 18-23 2010.

[3] Uday K. Chakraborty, editor. *Advances in Differential Evolution*. Springer, 2008.

[4] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. The MIT Press, 2004. ISBN 0-262-04219-3.

[5] V. Feoktistov. *Differential Evolution: In Search of Solutions (Springer Optimization and Its Applications)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[6] X-M. Hu, J. Zhang, and Y. Li. Orthogonal Methods Based Ant Colony Search for Solving Continuous Optimization Problems. *Journal of Computer Science and Technology*, 23:2–18, 2008.

[7] M. Lozano and F. Herrera. Call for Papers Special Issue of Soft Computing: A Fusion of Foundations, Methodologies and Applications on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems. Available at http://sci2s.ugr.es/eamhco/CFP.php, 2009.

[8] DE Home Page. http://www.icsi.berkeley.edu/~storn/code.html.

[9] Kenneth V. Price. An Introduction to Differential Evolution. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 79–108. McGraw-Hill, London, UK, 1999.

[10] Kenneth V. Price, Rainer M. Storn, and Jouni A. Lampinen. *Differential Evolution. A Practical Approach to Global Optimization*. Springer, Berlin, 2005. ISBN 3-540-20950-6.

[11] Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient adaptative scheme for global optimization over continuous spaces. Technical Report TR-95- 12, International Computer Science, Berkeley, California, March 1995.

[12] Rainer Storn and Kenneth Price. Differential Evolution: A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[13] L. Tseng and C. Chen. Multiple trajectory search for Large Scale Global Optimization. In *IEEE Congress on Evolutionary Computation*, pages 3052–3059, 2008.

[14] R. K. Ursem. Diversity-Guided Evolutionary Algorithms. In *Proceedings of Parallel Problem Solving from Nature VII (PPSN-2002)*, pages 462–471. Springer Verlag, 2002.

[15] H. Wang, Z. Wu, S. Rahnamayan, and D. Jiang. Sequential DE enhanced by neighborhood search for Large Scale Global Optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 4056–4062, Barcelona, Spain, July 2010.

[16] A. Zamuda, J. Brest, B. Boskovic, and V. Zumer. Large Scale Global Optimization using Differential Evolution with self-adaptation and cooperative co-evolution. In *IEEE Congress on Evolutionary Computation*, pages 3718–3725, 2008.

[17] S-Z. Zhao, J.J. Liang, P.N. Suganthan, and M.F. Tasgetiren. Dynamic multi-swarm particle swarm optimizer with local search for Large Scale Global Optimization. In *IEEE Congress on Evolutionary Computation*, pages 3845–3852, 2008.

[18] S-Z. Zhao, P. Suganthan, and S. Das. Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. *Soft Computing - A Fusion of Foundations, Methodologies and Applications - Special Issue on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems*, 2010. 10.1007/s00500-010-0645-4.

[19] S-Z. Zhao, P.N. Suganthan, and S. Das. Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1984–1990, Barcelona, Spain, July 2010.