Enhancing Evolutionary Algorithms by Efficient Population Initialization for Constrained Problems

Saber Elsayed*, Ruhul Sarker*, Noha Hamza*, Carlos A. Coello Coello† and Efrén Mezura-Montes‡
*School of Engineering and Information Technolog
University of New South Wales Canberra, Australia
Email: {s.elsayed,r.sarker,n.hamza
@unsw.edu.au}
†Depto. de Computación, CINVESTAV-IPN, Mexico
Email: ccoello@cs.cinvestav.mx
‡Artificial Intelligence Research Center
University of Veracruz, Mexico
Email:emezura@uv.mx

***Abstract*—One of the challenges that appear in solving constrained optimization problems is to quickly locate the search areas of interest. Although the initial solutions of any optimization algorithm have a significant effect on its performance, none of the existing initialization methods can provide direct information about the objective function and constraints of the problem to be solved. In this paper, a technique for generating initial solutions is proposed, which provides useful information about the behavior of both the objective function and the constraints. Based on such information, an automatic mechanism for selecting individuals, from the search areas of interest, is introduced. The proposed method is adopted with different evolutionary algorithms and tested on the CEC2006 and the CEC2010 test problems. The results obtained show the benefits of the proposed method in enhancing the performance, and reducing the average computational time, of several algorithms with respect to their versions adopting other initialization techniques**

***Index Terms*—constrained optimization problems, population initialization, evolutionary algorithms**

I. Introduction

Many engineering, business, computer science and defense decision processes require solving optimization problems in the presence of constraints. Such problems are known as constrained optimization problems (COPs). A COP may contain different types of variables and constraints. These problems become more challenging if they possess difficult characteristics, such as multi-modality, high dimensionality, and small feasible regions [1]. Formally, a COP can be expressed as:

$$\text{minimize } f(\overrightarrow{x})$$

$$\begin{aligned} \text{subject to: } & c_k(\overrightarrow{x}) \leq 0,\ k = 1, 2, \ldots, K \\ & h_e(\overrightarrow{x}) = 0,\ e = 1, 2, \ldots, E \\ & L_j \leq x_j \leq U_j,\ j = 1, 2, \ldots, D \end{aligned} \tag{1}$$

where $\overrightarrow{x} = [x_1, x_2, ..., x_D]$ is a vector with D decision variables, $f(\overrightarrow{x})$ the objective function, $c_k(\overrightarrow{x})$ the k^{th} inequality constraint, $h_e(\overrightarrow{x})$ the e^{th} equality constraint and each x_j has a lower limit (L_j) and an upper limit (U_j).

Over the years, the solution of COPs has attracted a considerable amount of research. Among the currently available approaches to deal with COPs, evolutionary algorithms (EAs), such as genetic algorithms (GAs) [2] and differential evolution (DE) [3], have become very popular. Normally, the first step in such algorithms is to generate an initial set of solutions to evolve. Due to the influence of such solutions in the performance of an EA, a considerable number of new initialization methods has been developed, with the main aim of uniformly cover the search space.

The most popular technique for generating a population of individuals is the pseudo-random number generator (PRNG) [4] which generates a sequence of of random numbers [4], in which the solutions are scattered according to a uniform distribution, or to any other statistical distribution. This initialization method is simple, but it has difficulties when the dimensionality increases [5] because it tends to fail in the generation of points that are fully distributed [4], [6]. Based on chaos theory [7], the chaotic number generator (CNG) [8] has been proposed for its use with EAs [8]. Among seven chaotic maps used with DE, the variant with the sinus map outperformed all the other variants [8]. As a type of space-filling method, uniform experimental design (UED) [9] searches for points to be uniformly distributed in a given range. However, evaluating such a large population is expensive for both small- and large-scale problems. This shortcoming was the motivation for introducing orthogonal design. An orthogonal array aims to specify a set of combinations spread uniformly over the space of all possible combinations. In the literature, such an initialization method enhanced the performance of several optimization approaches, such as DE [10]. Latin hypercube sampling (LHS) [11] divides the variables into a fixed number of intervals (creating grids)

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

and then generates a random value within each grid with the restriction that only one value exists in each row or column. Opposition-based learning (OBL) initialization method has demonstrated success in improving optimization algorithms [12]. In it, a set of initial solutions (original) is generated by any of the above methods, and subsequently a heuristic is used to generate opposite solutions$(\widetilde{x})$ from the original ones. After that, the best individuals from both populations are selected. Several variations of this method have been introduced, such as center-based sampling [13], generalized OBL [14] and current optimum OBL [15].

Generally, existing approaches have some shortcomings; for instance, none takes into consideration the behavior of the objective function and constraints of a problem in hand which may give vital indications to determining the approximate areas of interest for exploration, and subsequently may save computational efforts. In fact, this motivated Elsayed et al. [16] to introduce a heuristic space-filling approach to generate initial solutions in EAs.

Therefore, in this paper, we propose a new initialization approach that capitalizes the behavior of both objective function and constraints in deciding the initial individuals. The proposed technique divides the entire search space into a predefined number of unit spaces in a deterministic sequence within the decision variables' boundaries. Individuals are then produced by taking the corner points of those unit spaces. By considering the fitness values and constraints violation of these individuals, interesting behaviors of both of them are gained. Consequently, an automatic selection mechanism is introduced to select a subset of points to be used to form the initial population for an EA.

The proposed method was incorporated into five optimization algorithms and was used to solve a set of 24 well-known COPs. The results demonstrated the ability of the proposed method to uncover interesting information about the objective function and constraints. In addition, it is shown that the proposed approach had the ability to enhance the performance of all the algorithms considered by obtaining high-quality solutions and saving the average fitness evaluations and computational time up to 9.00% and 12.64%, respectively, in comparison with the same algorithm adopting a uniform distribution. Based on the above mentioned, the best two algorithms were selected, with their performance with different initialization methods tested on the CEC2010 test problems. The results confirmed the benefits of the proposed method in improving such algorithms by obtaining good solutions and saving the average number of fitness evaluations.

The remainder of this paper is organized as follows. The proposed approach is illustrated in Section II, followed by the experimental results and conclusions in Sections III and IV, respectively.

II. EA WITH SEQUENCE-BASED INITIALIZATION

It is fundamental that any EA starts with a set of PS initial solutions, i.e., $X = \{\overrightarrow{x}_1, \overrightarrow{x}_2, ..., \overrightarrow{x}_z, ..., \overrightarrow{x}_{PS}\}$. Then, each individual is evaluated, and the number of current fitness evaluations (cfe) is increased. A new population $\left(X'\right)$ is generated by applying the evolutionary operators. A selection operation is then carried out to determine the solutions that should survive to the next generation. The process continues until a stopping criterion is met.

As starting with good initial solutions may help achieving high-quality solutions, our focus is to develop an initialization method that can help getting information about the problem at hand and hence may determine the search areas of interest that an EA can focus on. The details of such a procedure is described below.

A. Sequence-based Initialization

Initially, the search domain $\left[\overrightarrow{L}, \overrightarrow{U}\right]$ is divided into $\gamma + 1$ segment vectors ($\overrightarrow{S}_\beta$, $\forall \beta = 1, 2, ..., \gamma + 1$), with the first and last ones being the lower and upper range vectors of all the decision variables, respectively. Firstly, the interval (I) of all $\overrightarrow{S}$ can be determined using equation (2).

$$\overrightarrow{I} = \frac{\left(\overrightarrow{U} - \overrightarrow{L}\right)}{\gamma} \tag{2}$$

Then, each $\overrightarrow{S}_\beta$ is generated as follows:

$$\overrightarrow{S}_\beta = \begin{cases} \overrightarrow{L} & \beta = 1 \\ \overrightarrow{S}_{\beta-1} + \overrightarrow{I} & 2 \leq \beta \leq \gamma \\ \overrightarrow{U} & \beta = \gamma + 1 \end{cases} \tag{3}$$

Then, a set of possible solutions is generated as described in Algorithm 1. For the β^{th} vector in S $\left(\overrightarrow{S}_\beta\right)$, some possible individuals are produced, such that starting with the last variable, i.e., $j = D$, by changing its value to all possible $S_{\beta,j}$ $\forall$ $\beta = \{1, 2, ...\gamma + 1\}$, no redundant points should be generated. It is worth mentioning that the sequence in which the points are generated is vital. Based on this approach, the maximum number of solutions (PS_{total}) that can be generated is $(\gamma + 1) \times ((D \times \gamma) + 1)$. It is worthy to mention that even for large scale problems, i.e., $D = 1000$, the total number of solutions that can be generated assuming that $\gamma = 10$, represents only 4% of the total evolutionary process.

By using this initialization method, it is possible to get important information about the problem landscape. For example, consider the following COP:

$$\text{Minimize } f(\overrightarrow{x}) = \sum_{j=1}^{D} - x_j \sin\left(|x_j|\right)$$

subject to:

$$c_1(\overrightarrow{x}) = \left(\sum_{j=1}^{D} \left(\left(\frac{x_j}{100}\right)^2 - 10\cos\left(2\pi\frac{x_j}{100}\right) + 10\right)\right) - 30 \leq 0$$

$$c_2(\overrightarrow{x}) = \sum_{j=1}^{D} (x_j) - 7.5D \leq 0$$

Algorithm 1 Population Initialization

define $\overrightarrow{\Omega} = [0, 0, ..., 0]_{\gamma+1}$ //to avoid generating similar individuals;
$i \leftarrow 0$;
for $\beta = 1{:}\gamma + 1$ **do**
  for $j = D : -1 : 1$ // for each decision variable **do**
    for $\alpha = 1{:}\gamma + 1$ **do**
      if $(\beta == \alpha)$ **and** $(\Omega(\beta) == 0)$ **then**
        $\Omega(\beta) \leftarrow 1$;
        $i \leftarrow i + 1$;
        $\overrightarrow{x}_i \leftarrow \overrightarrow{S}_\beta$;
        $x_{i,j} \leftarrow S_{\alpha,j}$; // change the j^{th} variable in the i^{th} point
      else if $(\beta == \alpha)$ **and** $(\Omega(\beta) == 1)$ **then**
        $\alpha \leftarrow \alpha + 1$; // skip generating a similar point
      else if $\beta \sim= \alpha$ **then**
        $i \leftarrow i + 1$;
        $\overrightarrow{x}_i \leftarrow \overrightarrow{S}_\beta$;
        $x_{i,j} \leftarrow S_{\alpha,j}$; // change the j^{th} variable in the i^{th} point
      end if
    end for
  end for
end for=0

where $D = 5$ and $-100 \leq x_j \leq 100$. The optimal solution for this problem is $\overrightarrow{x}^* = \{-20.4706137885503, -20.4706134690268, -20.4706136869296, -20.4706137241736, 97.6422527509791\}$ with $f(\overrightarrow{x}^*) = -179.4480$. Assuming $\gamma = 50$, 12801 points are generated (remember this number can be reduced by setting γ at a smaller value), with their $f(\overrightarrow{x})$, $c_1(\overrightarrow{x})$ and $c_2(\overrightarrow{x})$ calculated and plotted, as depicted in Figure 1. From this plot, important information can be obtained, such as: the objective function and the first constraint are both multi-modal. Such information may help when solving a black-box optimization problem. Also, we can put more emphasis on areas of interest (a_1 and a_2), which have feasible solutions. Also, if we analyze the corresponding x values generated, we can find that when all x values increase (the second half of points plotted), the solution becomes infeasible, i.e., c_2 cannot be satisfied. This piece of information may help in reducing the search space that is explored. In contrast, when we generate the same number of solutions using a uniform distribution, and plot them (see Figure 1(b)), such information cannot be easily obtained.

B. Constraint-handling

The selection of better solutions is carried out based on three cases: (1) for two feasible solutions, the best one (based on the quality of fitness value) is chosen; (2) a feasible solution is always better than an infeasible one; and (3) between two infeasible individuals, the one with a smaller sum of constraint violations (Θ) is selected, where Θ of an individual ($\overrightarrow{x_z}$) is

$$\Theta_z = \sum_{k=1}^{K} max(0, g_k(\overrightarrow{x_z})) + \sum_{e=1}^{E} max(0, |h_e(\overrightarrow{x_z})| - \epsilon_e) \quad (4)$$

where $g_k(\overrightarrow{x_z})$ is the k^{th} inequality constraint and $h_e(\overrightarrow{x_z})$ the e^{th} equality constraint. Note that, as some equality constraints may be difficult, instead of setting the right-hand side of any h_e to 0.0001 a large value is used and then reduced to 0.0001. The initial value of ϵ_e is set at the average constraint value of the best individuals in the initial population, i.e., $\epsilon_e = min\left(1, \frac{\sum_{z=1}^{PS}|h_e(\overrightarrow{x_z})|}{PS}\right)$.

C. EAs with proposed initialization method

The initialization method is carried out only once before starting the first run of the optimization process. Then, the corresponding fitness and constraints values are calculated. As this set of solutions is generated ***only once***, and to attain a fair comparison with existing optimization algorithms, this number of fitness evaluations is deducted from the overall number of fitness evaluations (FE$_{max}$) and distributed over all runs (25 runs are used here), so that we perform FE$_{max} - \left(\frac{PS_{total}}{Runs_{max}}\right)$ fitness evaluations, where $Runs_{max}$ is the maximum number the algorithm will run. Based on the shape of the curve of the fitness values and constraints violation of a problem at hand, PS individuals are selected as follows:

1) All the PS_{total} solutions are divided into different groups (n_g), each of which is of size PS_g.
2) Solutions with a total constraints violation less than a threshold (τ), i.e., $\Theta_z < \tau, \forall z = \{1, 2, ..., PS_{total}\}$, where $\tau = \frac{max(\Theta)}{2}$, are accepted, while the other ones are removed and PS_g is updated. The aim of this step is to get information from those infeasible solutions with good fitness values which may help.
3) For each group, based on the average fitness values and constraints violation of solutions that satisfy the above mentioned condition, n_g solutions are selected, such that
   - The quality index of each group is firstly calculated

$$\bar{f}_g = \frac{\sum_{i=1}^{PS_g} f(\overrightarrow{x})}{PS_g} \; \forall g = 1, 2, ..n_g \quad (5)$$

$$\lambda_f = \sum_{g=1}^{n_g} \frac{\sum_{gf=1}^{n_g} \bar{f}_{gf}}{\bar{f}_g} \quad (6)$$

$$Index_{f,g} = \left(\frac{\sum_{gf=1}^{n_g} \bar{f}_{gf}}{f_g}\right) \times \lambda_f^{-1}, \; \forall g = 1, 2, ..n_g \quad (7)$$

   - Similarly, the index of constraints violation is measured

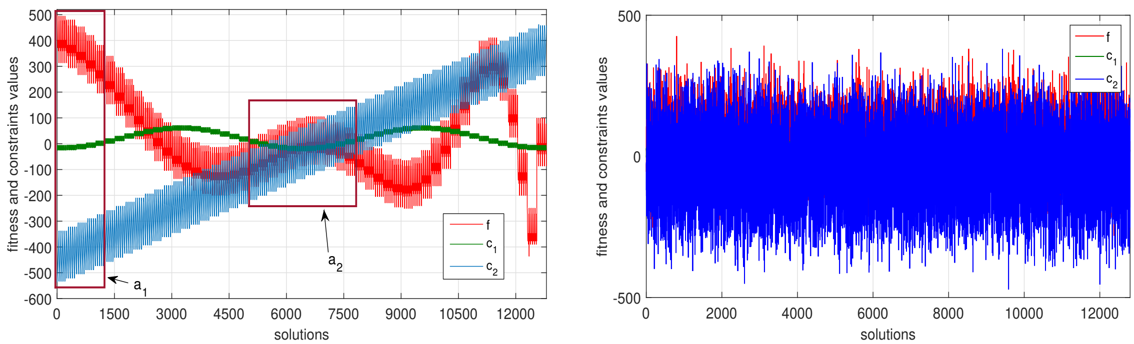
$$\bar{\Theta}_g = \frac{\sum_{i=1}^{PS_g} \Theta(\overrightarrow{x})}{PS_g} \; \forall g = 1, 2, ..n_g \quad (8)$$

$$\lambda_{vio} = \sum_{g=1}^{n_g} \frac{\sum_{gv=1}^{n_g} \bar{\Theta}_{gv}}{\bar{\Theta}_g} \quad (9)$$

$$Index_{\Theta,g} = \left(\frac{\sum_{gv=1}^{n_g} \bar{\Theta}_{gv}}{\Theta_g}\right) \times \lambda_{vio}^{-1}, \; \forall g = 1, 2, ..n_g \quad (10)$$

Θ is calculated as discussed in Section II-B.

   - Then, the average index ($AvgIndex$) of both rates is computed



(a) Proposed method

(b) Uniform distribution

Fig. 1. Fitness and constraints values of solutions generated by our proposed approach and by the use of a uniform distribution

$$AvgIndex_g = \frac{Index_{f,g} + Index_{\Theta,g}}{2}, \forall g = 1,2,..n_g \quad (11)$$

- Finally, the number of individuals selected from each group is as follows

$$PS_g = \frac{AvgIndex_g}{\sum_{l=1}^{n_g} AvgIndex_g} \times PS_{target}, \forall g = 1,2,..n_g \quad (12)$$

where PS_{target} is the maximum population size to be used.

4) The best individual in each group is selected, while the remaining $PS_g - 1$, $\forall\, g = \{1,2,...,n_g\}$ are randomly selected at the beginning of each run. This means $PS = \sum_{g=1}^{n_g} PS_g$.
5) For each individual $\overrightarrow{x}_i$, $\forall i = \{1,2,...,PS\}$, a new vector is generated, such that $\overrightarrow{x}'_i = \overrightarrow{x}_i + \overrightarrow{\triangle}_i$, where $\overrightarrow{\triangle}_i = \{\triangle_{i,1}, \triangle_{i,2}, ..., \triangle_{i,D}\}$ and $\triangle_{i,j}$ is randomly generated based on a uniform distribution within $\left[-\frac{(\overline{x}_j - \underline{x}_j)}{\gamma}, \frac{(\overline{x}_j - \underline{x}_j)}{\gamma}\right]$. Then, $\overrightarrow{x}'_i$ replaces $\overrightarrow{x}_i$ if it is better (based on both, its fitness value and the total sum of constraints violation) and we simultaneously update the number of fitness evaluations.

Note that steps 4 and 5 are considered to add randomness in each run to allow a fair comparison, i.e., we avoid starting with the same initial points in each run.

III. Experimental Results

In this section[1], the computational results obtained by five optimization algorithms (each of which has two variants, with the first adopting a uniform distribution to generate the initial solutions, while the proposed method was used in the second) for a set of 24 constrained problems [17] are presented and analyzed. The algorithms are (1) enhanced multi-operator DE (E-MODE) [18]; (2)DE$_1$: rand-to-$p$best with archive/1/bin (equation 13); (3) DE$_2$: current-to-$p$best with archive/1/bin DE (equation 14); (4) covariance matrix adaption-ES (CMA-ES) [19]. Note that as CMA was originally developed for unconstrained problems, the constraint-handling technique discussed in section II-B was adopted to make it work for COPs; and (5) DE with adaptation of operators and continuous control parameters (DE-AOPS) [20].

$$u^1_{z,j} = \begin{cases} x^1_{r_1,j} + F_z.(x^i_{\phi,j} - x^1_{r_1,j} + x^1_{r_2,j} - \widetilde{x}_{r_3,j}) \\ \qquad if(rand \le cr_z \text{ or } j = j_{rand}) \\ x^2_{z,j} \qquad otherwise \end{cases} \quad (13)$$

$$u^2_{z,j} = \begin{cases} x^2_{z,j} + F_z.(x^i_{\phi,j} - x^2_{z,j} + x^2_{r_1,j} - \widetilde{x}_{r_3,j}) \\ \qquad if(rand \le cr_z \text{ or } j = j_{rand}) \\ x^2_{z,j} \qquad \text{otherwise} \end{cases} \quad (14)$$

where $r_1 \neq r_2 \neq r_3 \neq z$ are random integer numbers, with $\overrightarrow{x}_{r_1}$ and $\overrightarrow{x}_{r_2}$ randomly selected from x^i, $x^i_{\phi,j}$ was selected from the best 10% individuals in x^i [21], while $\widetilde{x}_{r_3,j}$ was chosen from the union of the entire X and archive AR. Initially, the archive was empty. Then, parent vectors which failed in the selection process were added to it and, once its size exceeded a threshold, $1.4PS$, randomly selected elements were deleted to make space for the newly inserted ones [21].

As previously mentioned, based on the proposed method mentioned in Section II-C, PS can be changed from one problem to another. In other words, by setting $PS_{target} = 100$, PS is 82, 85, 87, 100, 100, 98, 93, 81, 81, 99, 97, 96, 83, 55, 93, 100, 86, 100, 94, 52, 100, 100, 99, 100 for all the 24 problems, respectively. Therefore, both versions of each algorithm started with the same PS. F and Cr of the first three algorithms were adapted as discussed in [18] with $H = 6$. For CMA-ES, $\mu = \frac{PS}{2}$ and $\sigma = 0.3$. $Runs_{max}$ was set to 25 times for each test problem for up to FE$_{max}$ $= 200,000$. Note that DE-AOPS used a different constraint-handling technique and adaptation mechanism of its parameters, as described in the corresponding paper. Also, as DE-AOPS uses an initial PS_{set} that has four values, in this paper, to start with the same number of solutions, we set it to $PS_{set} = \{PS_{id} - 15,\, PS_{id} - 10,\, ,PS_{id} - 5,\, PS_{id}\}$, where PS_{id} is the population size of the id^{th} problem previously mentioned.

[1]Due to pages limitation, some results were moved to a supplementary document which can be found HERE.

1) Quality of solutions: Considering the quality of the solutions obtained (see Tables 1 and 2 in the supplementary material file), it was found that all the algorithms obtained the optimal solutions for all problems, except for g_{20} and g_{22}, and CMA-ES for g_2. To our knowledge, no feasible solution has been found for g_{20} in the literature. Considering the average fitness values obtained, a summarized comparisons is given in Table I, which reveals that the proposed method adds benefits to the optimization algorithms considered.

TABLE I
SUMMARIZED COMPARISON BASED ON THE QUALITY OF THE SOLUTIONS

Variants	Average fitness values		
	Better	Similar	Worse
E-MODE(proposed) vs.E-MODE(random)	7	15	0
DE_1(proposed) vs.DE_1(random)	7	15	0
DE_2(proposed) vs.DE_2(random)	7	14	1
CMAES(proposed) vs. CMAES (random)	5	13	4
DE-AOPS(proposed) vs.DE-AOPS(random)	4	17	1

2) Statistical validation: Statistically speaking, the Wilcoxon signed rank test [22] was carried out with a significance level of 5%. Three symbols (+, −, and ≈) were used, with the first, second and third symbol meaning that the optimization algorithm with the proposed method was statistically superior, inferior and similar to the same algorithm with a uniform initialization, respectively. The results in Table II show E-MODE, DE_1 and DE_2 were superior to those adopting a uniform initialization. However, there was no significant difference between the CMA-ES variants. The same also occurred for the DE-AOPS variants. However, if we considered a significance level of 10%, DE-AOPS with the proposed method would statistically outperform the other variants.

TABLE II
WILCOXON TEST RESULTS BETWEEN EACH ALGORITHM WITH PROPOSED AND UNIFORM INITIALIZATION METHODS BASED ON AVERAGE FITNESS VALUES OBTAINED

Algorithms	p
E-MODE(proposed) vs. E-MODE(uniform)	0.012(+)
DE_1(proposed) vs. DE_1(uniform)	0.012(+)
DE_2(proposed) vs. DE_2(uniform)	0.021(+)
CMA-ES(proposed) vs. CMA-ES(uniform)	0.671(≈)
DE-AOPS(proposed) vs. DE-AOPS(uniform)	0.08(≈)

3) Computational time: To further show the benefit of the proposed method, the average savings in both (1) average fitness evaluations, and (2) computational time to reach the optimal solution with a threshold of $1e-04$ in the objective value ($\left|f\left(\overrightarrow{x}_{best,t}\right) - f\left(\overrightarrow{x^*}\right)\right| \leq 1e-04$, where $f\left(\overrightarrow{x}_{best,t}\right)$ and $f\left(\overrightarrow{x^*}\right)$ are the fitness values of the best feasible vector at generation t and the optimal solution, respectively), were computed, as shown in Table III. Note that all experiments were run on a PC with a Core(TM) i7-3770 CPU @ 3.40GHz, 16 GB RAM and Windows 7 using MATLAB 8.5.0.197613 (R2015a). The results show that up to 9.00% and 12.64% savings in the average fitness evaluations and computational time were achieved, respectively.

TABLE III
AVERAGE SAVINGS IN AVERAGE FITNESS EVALUATIONS AND TIME

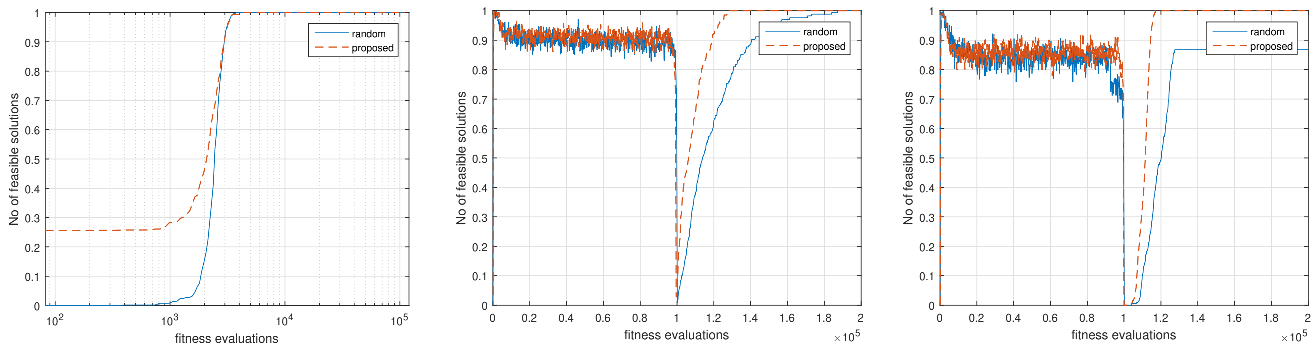
	Avg. FFEs (saving%)		Average computational time (saving%)	
	random	proposed	random	proposed
E-MODE	78102.2	**72131.82** (7.64%)	11.15	**9.74** (12.64%)
DE_1	79994.77	**72797.27** (9.00%)	9.86	**9.05** (8.13%)
DE_2	82852.91	**76492.65** (7.68%)	9.723	**8.97** (7.77%)
CMA-ES	79859.49	**75464.49** (5.5%)	9.73	**9.39** (3.5%)
DE-AOPS	59855.6	**58514.07** (2.24%)	9.813	**8.98** (8.46%)

4) Feasibility rate: It was also important to calculate the feasibility rate (FR) of the initial population (the ratio of the feasible solution (those with $\Theta = 0$) to PS) generated by both methods. The results shown in Table IV demonstrate that the proposed initialization method was able to have more feasible solutions than those generated by a uniform initialization. It is important to mention that for g_2, the reason for getting a high FR by the uniform initialization method was the way in which the initial points were selected from PS_{total}. To clarify, as discussed in II-C, some infeasible solutions might be preferred rather than feasible ones, as $\tau > 0$, but such behavior may be convenient considering the usefulness of infeasible solutions reported in the literature [23]. In addition, the FR of the population during the evolutionary generations was higher in case of using the proposed initialization, as depicted in Figure 2. For g_{03} and g_{15}, there is a sharp drop in the number of feasible solutions at 100K FFs, due to the way equality constraints are handled, in which, each h_e was relaxed with a parameter ϵ_e which adaptively reaches 0.0001 at $100,000$ FFs.

TABLE IV
FEASIBLE RATE OF THE INITIAL SOLUTIONS GENERATED BY UNIFORM AND THE PROPOSED INITIALIZATION METHODS

Prob	FR in initial population		Prob	FR in initial population	
	random	proposed		random	proposed
g_1	0.00%	25.61%	g_{13}	0.00%	0.00%
g_2	100.00%	76.19%	g_{14}	0.00%	0.00%
g_3	0.00%	3.91%	g_{15}	0.00%	0.00%
g_4	27.56%	38.92%	g_{16}	0.00%	0.00%
g_5	0.00%	0.00%	g_{17}	0.00%	0.00%
g_6	0.00%	0.00%	g_{18}	0.00%	1.00%
g_7	0.00%	0.00%	g_{19}	32.64%	63.96%
g_8	0.54%	5.68%	g_{20}	0.00%	0.00%
g_9	0.49%	25.73%	g_{21}	0.00%	0.00%
g_{10}	0.00%	0.00%	g_{22}	0.00%	0.00%
g_{11}	0.08%	4.12%	g_{23}	0.00%	10.14%
g_{12}	4.96%	89.04%	g_{24}	44.84%	74.76%

5) Convergence pattern: About the convergence rate, both E-MODE variants were selected, due to their good



(a) g01 (b) g03 (c) g15

Fig. 2. Percentage of feasible solutions in each generation for E-MODE with both initialization methods

performance, to depict their convergence plots, as shown in Figure 3, with the proposed method adding benefits to E-MODE by making it faster to converge to high-quality solutions. It is worth mentioning that, for g_{12}, E-MODE with the proposed method was able to obtain the optimal solution, which was at the corner of search space, in the first generation. Note that in problems with equality constraints, i.e., g_3, g_{17} and g_{23}, the curves go below the optimal solution and then go up, due to the way the equality constraints were handled, in which, each h_e was relaxed with a parameter ϵ_e, as discussed in II-B.

6) Comparison with other initialization methods: To show the benefit of the proposed method, all the algorithms previously adopted were run with two other initialization methods (LHS and OBL-uniform) and their performances were compared to those of a variant with the proposed initialization method. Then, the average savings in the average fitness evaluations and time were recorded. As presented in Table V, all the algorithms with the proposed method showed similar or superior performance to those with LHS and OBL-uniform. It was also interesting to observe that there was no difference between CMA-ES with LHS and its version with our proposed method. However, CMA-ES with the proposed method was able to save the average time and fitness evaluations compared with the same algorithm with the OBL-uniform method.

Regarding the quality of solutions, a closer inspection of Table V shows that algorithms with the proposed method have the ability to obtain better average results than those with the other initialization methods. However, no significant differences were found among algorithms, except DE_1, in which the variant with the proposed method was statistically better than that with the OBL-uniform method.

7) Parameters analysis: The new parameter introduced by the proposed method is γ . Therefore, in this subsection, its effects are analyzed. To do this, we selected one algorithm from above mentioned ones: E-MODE. This algorithm was ran with different values of γ, namely, $\gamma = 10$, 15, 20, 30 and 40.

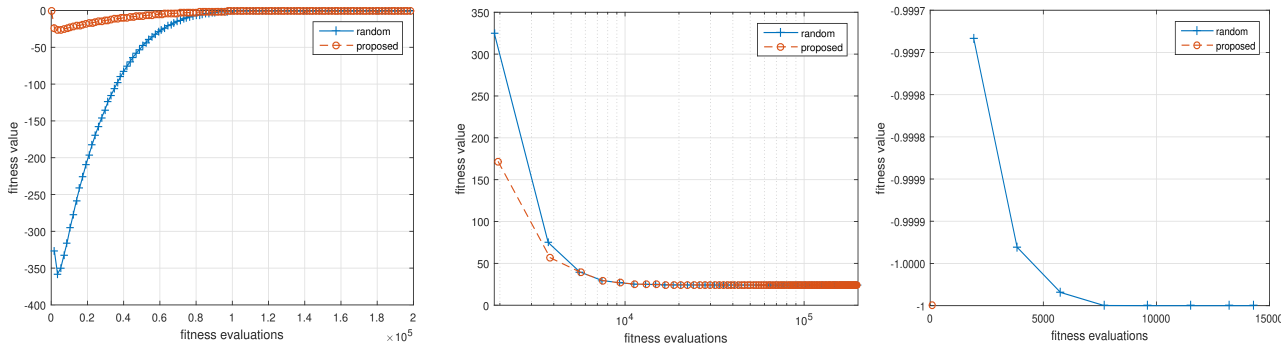
The first set of analyses examined the impact of γ on the quality of the solutions. The results show that all the variants were able to attain the optimal solution for the majority of test problems, with minor differences appearing in g_2, g_{13}, g_{17} and g_{21}. Nevertheless, it was hard to determine the best one. This was also the conclusion after carrying out the statistical test (Table VI), in which no significance difference was found among all the variants. To decide which one to use. The average ranks of all the variants based on the Friedman test were calculated (Table VII), and the results revealed that E-MODE with $\gamma = 10$ had the best performance, although no big differences were found.

The second set of analyses examined the impact of γ on the average fitness evaluations required to reach one of the stopping criteria mentioned in Section III-3. The results shown in Table VIII reveal that increasing the value of γ leads to an increase of the average fitness evaluations.

A. Solving the CEC2010 Benchmark

In this section, the performance of the proposed method is evaluated on the CEC2010 test suite with $30D$. Two algorithms (E-MODE and DE-AOPS) which showed good performance in solving the CEC2006 test suite were run 25 times for each test problem, using as a stopping criterion $600,000$ fitness evaluations. The average fitness errors from the best known solutions $\left(\left|f\left(\overrightarrow{x}_{best,t}\right) - f\left(\overrightarrow{x^*}\right)\right|\right)$ were considered successful when an algorithm achieved an average fitness error lower than $1.00\text{E}{-04}$ and the result was presented as zero.

The average fitness errors are shown in Tables 3 and 4 in the supplementary material document. In terms of solutions quality, it is clear from the comparison summary shown in Table IX that E-MODE with the proposed method was superior



(a) g03 (b) g07 (c) g12

Fig. 3. Convergence plots for E-MODE with random and the proposed initialization methods

TABLE V
COMPARISON SUMMARY OF DIFFERENT ALGORITHMS WITH BY PROPOSED INITIALIZATION METHOD WITH RESPECT TO THE LHS AND OBL-UNIFORM ONES

Algorithm	Variants	better	similar	worse	St. Test	Saving	
						Avg. FFE	avg. time
E-MODE	proposed vs. LHS	6	15	1	0.091 (≈)	10.53%	6.73%
	proposed vs. OBL-uniform	4	16	2	0.917 (≈)	5.62%	9.11%
DE_1	proposed vs. LHS	4	17	1	0.225 (≈)	6.16%	11.80%
	proposed vs. OBL-uniform	7	15	0	0.018 (+)	7.91%	14.09%
DE_2	proposed vs. LHS	4	16	2	0.753 (≈)	7.00%	9.45%
	proposed vs. OBL-uniform	4	15	3	0.612 (≈)	6.45%	10.71%
CMA-ES	proposed vs. LHS	6	11	5	0.790 (≈)	≈ 0%	≈ 0%
	proposed vs. OBL-uniform	7	10	5	0.875 (≈)	5.11%	3.94%
DE-AOPS	proposed vs. LHS	1	19	2	1.000 (≈)	≈ 1%	3.44%
	proposed vs. OBL-uniform	4	16	0	0.068 (≈)	2.13%	4.51%

TABLE VI
COMPARISON SUMMARY OF E-MODE WITH DIFFERENT VALUE OF γ

E-MODE Variants	better	similar	worse	St. Test
$\gamma = 10$ vs. $\gamma = 15$	4	17	1	0.500 (≈)
$\gamma = 10$ vs. $\gamma = 20$	4	18	0	0.068 (≈)
$\gamma = 10$ vs. $\gamma = 30$	4	18	0	0.068 (≈)
$\gamma = 10$ vs. $\gamma = 40$	3	18	1	0.456 (≈)

TABLE VII
AVERAGE RANKS OF E-MODE WITH DIFFERENT VALUE OF γ (BASED ON THE FRIEDMAN TEST)

$\gamma = 10$	$\gamma = 15$	$\gamma = 20$	$\gamma = 30$	$\gamma = 40$
2.7	2.93	3.09	3.18	3.09

to the other variants. Moreover, there was a bias towards DE-AOPS with the proposed method in terms of the number of problems in which it was successful in obtaining better solutions.

Statistically, E-MODE with the proposed initialization method was superior to the random initialization technique, while no difference was observed with respect to the two other variants. However, if we set the confidence level to 90%, E-MODE with the proposed initialization technique would statistically outperform the variant using the LHS method. For the DE-AOPS variants, none of these differences were statistically significant.

Further analysis showed that the proposed method helped E-MODE to save the average fitness evaluations by up to 1.92%. This rate increased to 5.80% when the proposed method was incorporated in DE-AOPS.

Based on the Friedman test, what stands out in Table X is that the proposed method was consistently ranked first with both algorithms. Interestingly, it was found that the random initialization method was the worst-performing method with E-MODE, but it outperformed the LHS method when they incorporated it in DE-AOPS.

TABLE VIII
AVERAGE FFE FOR E-MODE WITH DIFFERENT VALUE OF γ

$\gamma = 10$	$\gamma = 15$	$\gamma = 20$	$\gamma = 30$	$\gamma = 40$
72131.82	73996.53	74881.74	76005.8	75107.37

TABLE IX
SUMMARY OF THE COMPARISON OF E-MODE WITH THE PROPOSED METHOD WITH RESPECT TO THE OTHER METHODS

Algorithm	Variants	better	similar	worse	St. Test	FEs saving
E-MODE	proposed vs. LHS	6	11	1	0.063 (≈)	1.37%
	proposed vs. OBL	7	10	1	0.123 (≈)	0.82%
	proposed vs. Random	7	11	0	0.018 (+)	1.92%
DE-AOPS	proposed vs. LHS	8	7	3	0.248 (≈)	4.14%
	proposed vs. OBL	4	11	3	1.000 (≈)	5.80%
	proposed vs. Random	7	6	5	0.530 (≈)	4.01%

TABLE X

AVERAGE RANKS, BASED ON THE FRIEDMAN TEST FOR DIFFERENT VARIANTS OF E-MODE AND DE-AOPS WITH DIFFERENT INITIALIZATION METHODS

	proposed	LHS	OBL-uniform	Random
E-MODE	**2.00**	2.64	2.69	2.67
DE-AOPS	**2.28**	2.78	2.42	2.53

IV. CONCLUSIONS AND FUTURE WORK

For any EA, starting the evolutionary process with a good set of solutions could affect its performance in solving COPs. As a consequence, researchers have proposed several initialization methods to tackle this aspect. However, such existing initialization approaches could not get any direct information about the objective function and constraints being optimized. Recently, a new deterministic space filling approach showed its advantages, but it was only adopted to solve unconstrained problems. Therefore, in this paper, the approach was extended to solve COPs. In it, a set of solutions was systematically generated, and based on both the objective and constraints values, a subset of solutions was selected using an automatic selection procedure.

Based on the CEC2006 benchmark problems, the proposed approach was incorporated into five algorithms. In comparison with a uniform initialization method, the results showed that the algorithms with the proposed method were able to (1) achieve better solutions, (2) attain an average saving in time of up to 12.64% , (3) save the average fitness evaluations by 9.00%, and (4) attain a higher feasibility rate in the initial solutions and throughout the execution of the algorithm. All the algorithms with the proposed method were then compared with the same ones with LHS and OBL-uniform initialization methods. The results confirmed the superiority of the proposed method in saving both computational time and fitness evaluations. Also, it is concluded that setting γ at a value of 10 could achieve good results.

Further analysis was conducted by solving the CEC2010 test suite. Two algorithms were run with four initialization methods, with the proposed one achieving better results and/or reducing the average fitness evaluations. Also, it was observed that the proposed method was consistently ranked first based on a statistical test, while other methods might perform well with one algorithm, and poorly with another.

In terms of directions for future research, further work could investigate the performance of the proposed method in solving large-scale problems. Another possible area of future research would be to investigate how to use the initialization method in determining the set of operators and/or parameters to use in an optimization algorithm.

ACKNOWLEDGEMENTS

This research is supported by the UNSW Rector's start up grant awarded to S. Elsayed. C.A. Coello Coello gratefully acknowledges support from CONACyT grant no. 1920 and from a SEP-Cinvestav project (proposal no. 4).

REFERENCES

[1] S. M. Elsayed, R. A. Sarker, and D. L. Essam, “Multi-operator based evolutionary algorithms for solving constrained optimization problems,” *Computers & operations research*, vol. 38, no. 12, pp. 1877–1896, 2011.

[2] L. Davis, “Handbook of genetic algorithms,” 1991.

[3] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[4] B. Kazimipour, X. Li, and A. Qin, “A review of population initialization techniques for evolutionary algorithms,” in *IEEE Congress on Evolutionary Computation*, July 2014, pp. 2585–2592.

[5] H. Maaranen, K. Miettinen, and M. M. Mäkelä, “Quasi-random initial population for genetic algorithms,” *Computers & Mathematics with Applications*, vol. 47, no. 12, pp. 1885–1895, 2004.

[6] S. Helwig and R. Wanka, “Theoretical analysis of initial particle swarm behavior,” in *Parallel Problem Solving from Nature*. Springer, 2008, pp. 889–898.

[7] H. G. Schuster and W. Just, *Deterministic chaos: an introduction*. John Wiley & Sons, 2006.

[8] A. B. Ozer, “Cide: chaotically initialized differential evolution,” *Expert Systems with Applications*, vol. 37, no. 6, pp. 4632–4641, 2010.

[9] Y. Wang and K. T. Fang, “A note on uniform distribution and experimental design,” *KeXue TongBao*, vol. 26, no. 485, p. e9, 1981.

[10] W. Gong, Z. Cai, and L. Jiang, “Enhancing the performance of differential evolution using orthogonal design method,” *Applied Mathematics and Computation*, vol. 206, no. 1, pp. 56–69, 2008.

[11] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, “Design and analysis of computer experiments,” *Statistical science*, pp. 409–423, 1989.

[12] S. Rahnamayan, H. R. Tizhoosh, and M. Salama, “Opposition-based differential evolution,” *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[13] S. Rahnamayan and G. G. Wang, “Center-based sampling for population-based algorithms,” in *IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 933–938.

[14] H. Wang, Z. Wu, J. Wang, X. Dong, S. Yu, and C. Chen, “A new population initialization method based on space transformation search,” in *Fifth International Conference on Natural Computation*, vol. 5. IEEE, 2009, pp. 332–336.

[15] Q. Xu, N. Wang, and R. Fei, “Influence of dimensionality and population size on opposition-based differential evolution using the current optimum,” *Information Technology Journal*, vol. 12, no. 1, p. 105, 2013.

[16] S. Elsayed, R. Sarker, and C. A. C. Coello, “Sequence-based deterministic initialization for evolutionary algorithms,” *IEEE transactions on cybernetics*, 2016.

[17] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. A. Coello Coello, and K. Deb, “Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization,” Tech. Rep., 2005.

[18] S. Elsayed, R. Sarker, and C. C. Coello, “Enhanced multi-operator differential evolution for constrained optimization,” in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 4191–4198.

[19] N. Hansen, S. Müller, and P. Koumoutsakos, “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es),” *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[20] S. Elsayed, R. Sarker, C. C. Coello, and T. Ray, “Adaptation of operators and continuous control parameters in differential evolution for constrained optimization,” *Soft Computing*, pp. 1–22, 2017.

[21] J. Zhang and A. C. Sanderson, “Jade: adaptive differential evolution with optional external archive,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[22] G. W. Corder and D. I. Foreman, *Nonparametric statistics for non-statisticians: a step-by-step approach*. John Wiley & Sons, 2009.

[23] E. Mezura-Montes and C. A. C. Coello, “A simple multimembered evolution strategy to solve constrained optimization problems,” *IEEE Transactions on Evolutionary computation*, vol. 9, no. 1, pp. 1–17, 2005.

APPENDIX

The supplementary material document can be found **HERE**.