2016 IEEE Congress on Evolutionary Computation (CEC)
10.1109/CEC.2016.7744164

Testing United Multi-operator Evolutionary Algorithms-II on Single Objective Optimization Problems

Saber Elsayed, Noha Hamza and Ruhul Sarker
School of Engineering and Information Technology
University of New South Wales at Canberra
Canberra 2600, Australia
Emails: s.elsayed@adfa.edu.au; noha.hamza@student.adfa.edu.au; r.sarker@adfa.edu.au

***Abstract*—Over the past few years, the success of multi-operator and multi-method algorithms encouraged researchers to combine them within a single framework. Although these algorithms have shown promising results, there are still rooms for further improvements. In this paper, we propose a new way of combining multiple evolutionary algorithms, each of which may run with multiple search operators. In its process, the algorithm gradually places emphasis on the better-performing multi-operator algorithm, as well as its own search operators. Such a process is designed based on the quality of solutions produced and diversity of the population. The proposed algorithm is assessed on the CEC2016 competition problems on single objective real-parameter optimization, with the results demonstrating its ability to attain better results than those of state-of-the-art algorithms.**

***Keywords*—Multi-operator, multi-method, differential evolution, covariance matrix adaptation evolution strategy**

I. Introduction

Optimization problems involve finding the values of decision variables by optimizing (either maximizing or minimizing) one or more objective functions [1]. Such problems can be found in many fields including, but not limited to, science, engineering and business [2]. Mathematically, it can be formulated as

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

$$\text{subject to: } \underline{x}_j \leq x_j \leq \overline{x}_j,\ j = 1, 2, \ldots, D \qquad (1)$$

where$f(\overrightarrow{x})$ is the objective function, $\overrightarrow{x} = [x_0, x_2, ..., x_D]$ a vector with D decision variables with each x_j has lower and upper limits $\underline{x}_j$ and $\overline{x}_j$, respectively.

Over the years, evolutionary algorithms (EAs) have shown their ability to successfully solve such problems. The EAs family contains different algorithms, such as genetic algorithms (GA) [3], differential evolution (DE) [4] and evolution strategy (ES) [5]. However, no single EA performs consistently well for all types of problems. For instance, GA was found to be able to solve noisy problems, but its convergence rate was slow in comparison with that of DE [6]. Also, DE was suitable when the feasible patches are parallel to the axes, but in multi-modal functions it could become stuck in local solutions, and the same is for many other EAs [6]. As a consequence, the multi-method EAs and multi-operator EAs concepts have emerged, in which the first utilizes the strengths of different EAs within a single framework, while multiple mutation and/or crossover operators are used in the later.

For instance, Vrugt et al. [7] introduced an algorithm, known as a multi-algorithm genetically adaptive multi-objective (AMALGAM) that has been proven to be a powerful approach for solving multi-objective problems. Later, it was extended for single objective problems (AMALGAM-SO) [8]. The algorithm obtained similar efficiencies as existing algorithms on relatively simple problems, but it was increasingly superior for more complex and higher-dimensional multi-modal optimization problems. Peng et al. [9] proposed a population-based algorithm portfolio (PAP) framework, which used multiple EAs as its constituent algorithms, then ran each algorithm with a part of the given fitness evaluations (time budget). In addition, a migration scheme was used among the constituent algorithms. The algorithm showed its superiority to other algorithms on a set of unconstrained problems. Considering multi-operator-based algorithms, Qin et al. [10] proposed a self-adaptive DE algorithm (SaDE) that used four mutation strategies and each individual in the population assigned to one of them based on a given probability. Then, the selection probability of each operator was updated based on its success and failure rates during previous generations (a learning period). Zamuda and Brest [11] introduced an algorithm that employed two mutation strategies in jDE [12], with the population size adaptively reduced during the evolutionary process based on their earlier technique proposed in [13]. Its performance on some real-world applications was better than that of two other algorithms. The algorithm was then extended by embedding a self-adaptation mechanism for parameter control [14], in it, whereby the three DE strategies were used, each of which was applied for a specific group of individuals based on predefined parameters. In addition, if an individual was not improved for a predefined number generations, then it would be reinitialized with a pre-defined probability. Recently, an adaptive DE algorithm was introduced [15], which utilized four mutation strategies in a sequential manner, i.e., one mutation at every predefined number of generations. In addition, a mechanism was used to reduce the population size.

Generally speaking, although such concepts improved the performance of optimization algorithms, there are still scopes for further improvement through better algorithm configuration. Hence, further investigations are needed to analyze the existing algorithm design and their performances. One possible way to design an improved algorithm is to combine multiple algorithms and/or operators under a single algorithmic framework, which was the goal of Elsayed et al. [16], as they proposed a united multi-operator EAs (UMOEAs), where multi-operator DE, multi-operator GA and covariance matrix adaptation ES (CMA-ES) were integrated within a single algorithm framework. The algorithm was applied to solve different optimization problems, where the results showed significant improvement over existing algorithms. This is an interesting design process where the algorithm is configured while solving the problem. However, a further research would be useful for finding better ways of configuring such algorithms.

In this paper, we propose an improvement of UMOEAs (introduced as UMOEAs-II). Although, the framework can consider any number of EAs, in this paper, it utilizes the search ability of efficient multi-operator DE and CMA-ES algorithms. Both are used to evolve two distinct sub-populations, for a predefined number of generations (cycle). After that the probability of applying each algorithm in the subsequent cycle is updated based on two criteria: (1) the quality of solutions; and (2) the diversity of the sub-population. After the second cycle, a simple information sharing scheme is used, and then both algorithms run again for another cycle, and the same steps are carried out again. Furthermore, to enhance the exploitation capability of the proposed algorithm, a local search is used at later generations based on the probability that dynamically changes. To clarify, the differences between UMOEAs-II and UMOEAs are: (1) as the multi-operator GA used in UMOEAs was not complementary to multi-operator DE and CMA-ES, it is removed in UMOEAs-II, (2) the interior-point method is used in UMOEAs-II as a local search, (3) a more powerful multi-operator DE is used in UMOEAs-II with different DE operators and parameters, (4) the selection of the best-performing EA and/or operator is different from that used in UMOEAs, and (5) other components, such as the information sharing scheme, and keeping all EA active to use during the optimization process, are different from those in UMOEAs.

The performance of the proposed algorithm is evaluated on the CEC2016's competition on unconstrained problems, which are taken from [17], with 10, 30, 50 and 100 dimensions. The results demonstrate its ability to obtain good quality solutions which are better than those of other state-of-the-art algorithms.

The rest of this paper is organized as follows: an overview of DE and CMA-ES is provided in Section II; the proposed algorithm is elaborated in Section III; finally the experimental results and conclusions are discussed in Sections IV and VI, respectively.

II. Basic algorithms and operators

In this section, DE and CMA-ES, considered in this study, are briefly described.

A. *Differential evolution*

DE is a population-based stochastic algorithm [18]. Its algorithmic steps starts with a random vectors $(X = \{\overrightarrow{x}_1, \overrightarrow{x}_2, ... \overrightarrow{x}_{PS}\})$, where PS is the population size, each of which should be within the search domain. Then, a mutant population $(V = \{\overrightarrow{v}_1, \overrightarrow{v}_2, ... \overrightarrow{v}_{PS}\})$ is generated. Subsequently, every $\overrightarrow{x}_z$ is recombined with its corresponding $\overrightarrow{v}_z$ to generate a trial vector $\overrightarrow{u}_z$, where $z = \{1, 2, ..., PS\}$. A pair-wise comparison between $\overrightarrow{x}_z$ and $\overrightarrow{u}_z$, with the winning vectors, based on the fitness values and/or constraints violation, becoming the new population at the next generation X_{t+1}, where t is the generation number [1]. Below is a brief description of each step mentioned above.

- **Initialization:** each individual in the population is represented as a D-dimensional vector in which each variable is generated within its boundaries:

$$x_{z,j} = \underline{x}_j + rand_j(0,1) \times (\overline{x}_j - \underline{x}_j) \, \forall j = 1, 2, ...D \tag{2}$$

  where $rand_j(0,1)$ is a uniform random number within $[0,1]$ [1].

- **Mutation:** in this step new solutions that are perturbations of the current ones in which, in its simplest form (DE/rand/1), $\overrightarrow{v}_z$ is generated by adding a scaled difference between two random vectors to a third one (equation 3).

$$\overrightarrow{v}_z = \overrightarrow{x}_{r_1} + F \times (\overrightarrow{x}_{r_2} - \overrightarrow{x}_{r_3}) \tag{3}$$

  where $\overrightarrow{x}_{r_1}$, $\overrightarrow{x}_{r_2}$ and $\overrightarrow{x}_{r_3}$ are distinct solution vectors in the current population and none similar to $\overrightarrow{x}_z$, F a positive real number that controls the rate at which the population evolves[1]. Also, $\overrightarrow{x}_{r_1}$ is called the *base vector*.
  Over the last two decades many variations of this operator have been introduced, and for more details, readers are referred to [19].

- **Crossover:** two well-known crossover schemes, binomial and exponential, exist in the literature. The former, which sometimes is called a uniform or discrete [1], is conducted on every $j \in [1, D]$ with a predefined crossover probability. In particular, for each j, a uniform random number $(rand_j(0,1))$ is generated. If its value is less than Cr, the value of $\overrightarrow{u}_{z,j}$ will be copied from the corresponding value from $\overrightarrow{v}_{z,j}$, otherwise it will be equal to $\overrightarrow{x}_{z,j}$:

$$u_{z,j} = \begin{cases} v_{z,j} & if\ (rand_j(0,1) \leq cr \text{ or } j = j_{rand}) \\ x_{z,j} & \text{otherwise} \end{cases} \tag{4}$$

  where $j_{rand} \in 1, 2, ..., D$ is a randomly integer index which ensures that $\overrightarrow{u_z}$ obtains at least one component from $\overrightarrow{v_z}$.
  On the other hand, an exponential crossover is similar to a two-point crossover in which the first cut point (l) is randomly selected from a range $[1, D]$ and the second is determined such that L components are copied from $\overrightarrow{v}_z$ [20] as:

$$u_{z,j} = \begin{cases} v_{z,j} & \forall j = \langle l \rangle_D, \langle l+1 \rangle_D, ..., \langle l+L-1 \rangle_D \\ x_{z,j} & \forall j \in [1, D] \end{cases} \tag{5}$$

where $\langle l \rangle_D$ denotes a modulo function with a modulus of D and $L \in [1, D]$.

- **Selection:** DE uses a simple one-to-one survivor selection in which, at generation (t), $\overrightarrow{u}_{z,t}$ competes against $\overrightarrow{x}_{z,t}$, and the better, in terms of the fitness value and/or constraint violation, is considered a vector in the new population in the next generation $(t+1)$.

B. CMA-ES

Over the last two decades, CMA-ES has shown its ability to efficiently solve diverse types of optimization problems [21]. CMA-ES was derived from the concept of self-adaptation in ES, which adapts the covariance matrix of a multivariate normal search distribution. In it, the new individuals are generated by sampling from a Gaussian distribution, and instead of using a single mutation step, it considers the path that the population takes over generations [22, 5]. The main steps in CMA-ES are as follows:

1) Generate an initial solution[1] $(\overrightarrow{x}_m)$ and evaluate the fitness function.
2) Sample new individuals, such that

$$\overrightarrow{x}_{z,t+1} = N\left(\overrightarrow{x}_{m,t}, \sigma_t^2 C_t\right) = \overrightarrow{x}_m + \sigma_t N\left(0, C_t\right), \quad \forall z = 1 : PS \tag{6}$$

3) Evaluate the new offspring and sort them based on the fitness values.
4) The best μ individuals are selected as a parental vector, and their center is calculated according to

$$\overrightarrow{x}_{m,t+1} = \sum_{k=1}^{\mu} w_k \overrightarrow{x}_{k,t} \tag{7}$$

where $\sum_{k=1}^{\mu} w_k = 1$ and $w_1 \geq w_2 \geq ... \geq w_\mu$.
5) Update the evolution paths p_{t+1}^s and p_{t+1}^σ .
6) Adapt the covariance matrix (C_{t+1}).
7) Update global step size (σ_{t+1}).
8) Repeat steps 2 to 7 until a stopping criterion is met.

III. United Multi-operator EAs-II

In this section, the proposed framework and its components are described.

A. Framework

The main steps of the proposed algorithm are described in Algorithm 1. Firstly, an initial population of size PS is randomly generated. Then, PS is divided into two sub-populations of sizes, PS_1 and PS_2, respectively. The solutions in the first population are evolved by a multi-operator DE algorithm (MODE), while those in the second are evolved using CMA-ES, where the probabilities ($prob_1$ and $prob_2$) of applying each algorithm in each generation, respectively, are dynamically changing over the optimization process. In the beginning, both $prob_1$ and $prob_2$ are equal to 1. Hence, for a per-defined number of generations, also called a cycle (CS), both algorithms are running in parallel. Once a cycle is finished, $prob_1$ and $prob_2$ are updated based on two factors (1) the quality of the best solution in each sub-population; and (2) the diversity rate of each sub-population.

[1]It is also possible to start with a set of solutions and then consider taking the weighted mean vector of the μ best solutions.

Algorithm 1 General framework of UMOEAs-II

```
1:  Define $PS \leftarrow PS_1 + PS_2$, $cy \leftarrow 0$, $prob_1 \leftarrow 1$, $prob_2 \leftarrow 1$, $t \leftarrow 1$, $FFE_{max} \leftarrow 10,000 \times D$, $prob_{ls} \leftarrow 0.1$ and all other parameters required (Section IV).
2:  At $t = 1$, generate initial $PS$ random individuals $(X)$. The variables of each individual $(\overrightarrow{x_z})$ must be within their boundaries;
3:  Randomly assign $PS_1$ and $PS_2$ individuals from $X$ to $X_i, \forall i = 1, 2$, respectively;
4:  while $cfe < FFE_{max}$ do
5:      $cy \leftarrow cy + 1$
6:      if $cy = CS$ then
7:          Update $prob_1$ and $prob_2$ (Section III-D);
8:      end if
9:      if $cy = 2 \times CS$ then
10:         Share information;
11:         $prob_1 \leftarrow 1$, $prob_2 \leftarrow 1$;
12:         $cy \leftarrow 0$;
13:     end if
14:     if $rand \in [0, 1] \leq prob_1$ then
15:         Apply MODE, update $cfe$ and sort $X_1$;
16:     end if
17:     if $rand \in [0, 1] \leq prob_2$ then
18:         Apply CMA-ES, update $cfe$ and sort $X_2$;
19:     end if
20:     if $cfe \geq 0.75 \times FFE_{max}$ then
21:         if $rand \in [0, 1] \leq prob_{ls}$ then
22:             Apply interior-point method to the best solution found for up to $cfe_{LS}$ fitness evaluations;
23:             if the solution is improved then
24:                 $prob_{ls} \leftarrow 0.1$;
25:                 Update the best solution in $X_1$, $\overrightarrow{x}_{m,t}$ in $X_2$ and $\sigma$;
26:             else
27:                 $prob_{ls} \leftarrow 0.0001$;
28:             end if
29:         end if
30:     end if
        $t \leftarrow t + 1$, and go to step 4;
31: end while
```

In each generation in the subsequent cycle, two random numbers are generated, if the first is less than $prob_1$, then MODE will run to evolve its own sub-population. Similarly, if the second random number is less than $prob_2$, then CMA-ES will be used to evolve its individuals. Once the cycle is finished, an information sharing scheme is done and both $prob_1$ and $prob_2$ are reset to 1. Therefore, both algorithms will run for another cycle, and the same process re-carried out. To increase the exploitation capability of UMOEAs-II, the interior-point method is applied at the later stages with a dynamic probability. The algorithm continues until a stopping criterion is met.

In the following subsections, the algorithm's components are discussed.

B. MODE

As mentioned above, MODE starts with PS_1 individuals which are randomly taken from the entire PS individuals. In this paper, MODE uses three DE variants (DE$_1$, DE$_2$ and DE$_3$), with the first uses DE/current-to-$p$best with archive as a mutation operator, while the second employs DE/current-to-$p$best without archive, and third uses a wighted-rand-to-ϕbest, with the binomial crossover was used in all variants, as

$$u_{z,j} = \begin{cases} x_{z,j} + F_z\left(x_{p,j} - x_{z,j} + x_{r_1,j} - \widetilde{x}_{r_2,j}\right) \\ \qquad\qquad if\left(rand \le cr_z \text{ or } j = j_{rand}\right) \\ x_{z,j} \qquad\qquad\qquad \text{otherwise} \end{cases} \tag{8}$$

$$u_{z,j} = \begin{cases} x_{z,j} + F_z\left(x_{p,j} - x_{z,j} + x_{r_1,j} - x_{r_3,j}\right) \\ \qquad\qquad if\left(rand \le cr_z \text{ or } j = j_{rand}\right) \\ x_{z,j} \qquad\qquad\qquad \text{otherwise} \end{cases} \tag{9}$$

$$u_{z,j} = \begin{cases} F_z x_{r_1,j} + \lambda\left(x_{\phi,j} - x_{r_3,j}\right), \\ if\left(rand \le cr_z \text{ or } j = j_{rand}\right) \\ \\ x_{z,j} \qquad\qquad\qquad \text{otherwise} \end{cases} \tag{10}$$

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_1, $\overrightarrow{x}_p$ is selected from the best 10% individuals in X_1 [23], while $\widetilde{x}_{r_2,j}$ is chosen from the union of the entire X_1 and archive AR. Similar to JADE, 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, $2.6PS$ [24], randomly selected elements were deleted to make space for the newly inserted ones [23], $\overrightarrow{x}_\phi$ is selected from the best 50% individuals in X_1 [25] and λ is equal to 1.

Initially, the probability of evolving any individual using DE$_1$ or DE$_2$ or DE$_3$ is set to $prob_{DE_1} = prob_{DE_2} = prob_{DE_3} = \frac{1}{3}$, i.e., if $rand_z \in [0,1] \le 0.33, \forall z = 1,2,...,PS_1$, then evolve $\overrightarrow{x}_z$ using DE$_1$, if $0.33 \le rand_z \in [0,1] \le 0.667$, use DE$_2$, otherwise DE$_3$ is used. As using the raw values of the fitness improvements as a measure of success may affect the robustness of the algorithm's performance, in this paper, the improvement rates in fitness values are considered to update the probability of each variant (I_{DE_i}). So, at the end of each generation, the rates are calculated such that

$$I_{DE_i} = \frac{\sum_{z=1}^{PS_1} max\left(0, f_{new_z} - f_{old_z}\right)}{\sum_{z=1,DE=1}^{PS_1} f_{old_z}}, \quad \forall \overrightarrow{x}_z \text{ updated by } DE_i \text{ and } i = 1,2,3 \tag{11}$$

where f_{new} and f_{old} are the new and old fitness values, respectively.

Then, each probability is updated as

$$prob_{DE_i} = \max\left(0.1, \min\left(0.9, \frac{I_{DE_i}}{I_{DE_1} + I_{DE_2} + I_{DE_3}}\right)\right), \quad \forall i = 1,2,3 \tag{12}$$

As one operator may perform good at different stages of the evolutionary process, and perform badly in others, a minimum value of $prob_{DE_1}$is used. So, there will be a chance of any variant to get improved. Also, there is a possiblity to reset each probability to its initial value, i.e., $\frac{1}{3}$.

Also, to maintain diversity within the early evolutionary process, while enhancing the exploitation ability in later ones [24], a linear reduction of PS_1 is carried out at the end of each generation by removing the worst individual, such that

$$PS_{1,t+1} = \text{round}\left(\left(\left(\frac{PS_{1,min} - PS_{1,max}}{FFE_{max}}\right) \times cfe\right) + PS_{1,max}\right) \tag{13}$$

where $PS_{1,max}$ and $PS_{1,min}$ are the maximum and minimum values of PS_1, respectively, and FFE_{max} the maximum number of fitness evaluations.

1) Adaptation of F and Cr : In this paper, the mechanism proposed in [24], which is considered as an improvement of JADE [23], is adopted. It works as follows:

- A historical memory with H entries for both parameters (M_{Cr}, M_F) is initialized, where all values are initially set to a value of 0.5.
- Each individual $\overrightarrow{x}_z$ is associated with its own Cr_z and F_z, such that

$$Cr_z = \text{randni}(M_{Cr,r_z}, 0.1) \tag{14}$$

$$F_z = \text{randci}(M_{F,r_z}, 0.1) \tag{15}$$

  where r_z is randomly selected from $[1,H]$, randni and randci are values randomly selected from normal and Cauchy distributions with mean M_{Cr,r_z} and M_{F,r_z}, respectively, and variance 0.1.
- At the end of each generation, Cr_z and F_z used by the successful individuals are recorded in S_{Cr} and S_F, and then the contents of memory are updated as follows

$$M_{Cr,d} = mean_{WA}\left(S_{Cr}\right) \; if \; S_{Cr} \neq \text{null} \tag{16}$$

$$M_{F,d} = mean_{WL}\left(S_F\right) \; if \; S_F \neq \text{null} \tag{17}$$

  where $1 \le d \le H$ is the the position in the memory to be updated. It is initialized to 1, and then incremented whenever a new element is inserted into the history, and if it is greater than H, it is set to 1. mean$_{WA}(S_{Cr})$ and mean$_{WL}(S_F)$ are computed as follows [2]

$$\text{mean}_{WA}(S_{Cr}) = \sum_{\gamma=1}^{|S_{Cr}|} w_\gamma S_{cr,\gamma} \tag{18}$$

$$\text{mean}_{WL}(S_F) = \frac{\sum_{\gamma=1}^{|S_F|} w_\gamma S_{F,\gamma}^2}{\sum_{\gamma=1}^{|S_F|} w_\gamma S_{F,\gamma}} \tag{19}$$

  where

[2]$|S_{Cr}|$ is the number of successful Cr recorded in S_{Cr}, with $|S_{Cr}|$=$|S_F|$

$$w_\gamma = \frac{\Delta f_\gamma}{\sum_{\gamma=1}^{|S_{cr}|} \Delta f_\gamma} \tag{20}$$

and $\Delta f_\gamma = |f_{\gamma,old} - f_{\gamma,new}|$.

C. CMA-ES

UMOEAs-II uses CMA-ES described in Section II-B, with the difference that it starts with a random initial population $(X_2) = \{\overrightarrow{x}_{2,1}, \overrightarrow{x}_{2,2}, ..., \overrightarrow{x}_{2,z}, ..., \overrightarrow{x}_{2,PS_2}\}$ of size PS_2, with each individual is uniformly initialized within the whole search space, and then the initial $\overrightarrow{x}_m$ is set to the arithmetic mean of X_2, i.e., $\overrightarrow{x}_m = \sum_{z=1}^{PS_2} w_z \overrightarrow{x}_{2,z}$, where $\sum_{z=1}^{PS_2} w_z = 1$ and $w_1 = w_2 = ... = w_{PS_2} = \frac{1}{PS_2}$.

D. Improvements measure

As previously mentioned, to update $prob_1$ and $prob_2$, two factors are considered: (1) the quality of solutions obtained; and (2) the diversity of each sub-population.

Considering the first factor, the best solution obtained in each sub-population at the end of the cycle is considered. Then, the normalized quality values (NQ) are calculated, as

$$NQ_i = \frac{f_{CS,i}^{best}}{f_{CS,1}^{best} + f_{CS,2}^{best}}, \forall i = 1, 2 \tag{21}$$

Simultaneously, the diversity rate is calculated, as

$$div_i = \sum_{z=2}^{PS_i} dis\left(\overrightarrow{x}_{i,z}, \overrightarrow{x}_{best}\right), \forall\, i = 1, 2 \tag{22}$$

where $dis\left(\overrightarrow{x}_z, \overrightarrow{x}_{best}\right)$ is the Euclidean distance between the z^{th} individual and best individual in PS_i. Note that the best solution is $\overrightarrow{x}_1$, as each sub-population is sorted based on the fitness values in every generation.

Hence, the normalized diversity $(Ndiv)$ is as follows

$$Ndiv_i = \frac{div_i}{div_1 + div_2}, \forall i = 1, 2 \tag{23}$$

Then, the improvement index is updated, such that

$$I_i = (1 - NQ_i) + Ndiv, \forall i = 1, 2 \tag{24}$$

The reason for subtracting NQ_i from 1 is to satisfy the maximization target of I_i.

Finally,

$$prob_i = \max\left(0.1, \min\left(0.9, \frac{I_i}{I_1 + I_2}\right)\right), \forall i = 1, 2 \tag{25}$$

It is worthy to mention that in case that the total sum of I is zero, both $prob_1$ and $prob_2$ are set to 1.0.

E. Information sharing

The information sharing scheme in UMOEAs-II is simple. In it, as described in Algorithm 1, at the end of the second cycle $(cy = 2 \times CS)$, if $prob_1 > prob_1$, i.e., MODE is considered the best, based on III-D, then X_2 is replaced with PS_2 random individuals from X_1.Then, the CMA-ES's parameters are reset to their initial values, except σ that is updated as $\sigma = \sigma_{initial} \times \left(1 - \frac{cfe}{FFE_{max}}\right)$. This step can be seen as a restart mechanism of CMA-ES, which may enhance its performance.

On the other hand, if CMA-ES is the best-performing algorithm, the worst individual in X_1 (i.e., $\overrightarrow{x}_{ps_1,1}$) is substituted by the best individual in X_2.

Note that after the information sharing scheme is carried out, the algorithms goes back to the first cycle (step 12 in Algorithm 1), which means that this process is periodic.

F. Local search

To increase the exploitation ability of UMOEAs-II, at each generation of the last 25% of the evolutionary process, the interior-point method [26] is applied to the best individual found so far, with a probability of $prob_{ls} = 0.1$, and for up to cfe_{LS} fitness evaluations. It is worthy to mention that $prob_{ls}$ is dynamic, in which if the local search is not successful in finding a better solution, $prob_{ls}$ is set to a small value, i.e., 0001, otherwise it is set to 0.1.

In case that the new solution $(\overrightarrow{x}_{ls})$ is better than the best solution found so far, it replaces the worst individual in X_1. Also, the mean vector, $\overrightarrow{x}_m$, in CMA-ES is set to $\overrightarrow{x}_{ls}$ and σ is set to a small value.

IV. Experimental Results

In this section, the computational results obtained by UMOEAs-II for the set of CEC2016 unconstrained problems (which are taken from [17]) are presented and analyzed.

The algorithm was run 51 times for each test problem, with the stopping criterion run for up to $FFE_{max} = 10,000 \times$ D, with D = 10, 30, 50 and 100. For MODE, $PS_{1,max}$ was 18D individuals and $PS_{1,min}$ 4, H = 6 [24]. For CMA-ES, $PS_2 = 4 + \lfloor (3log\,(D)) \rfloor$ [21], $\mu = \frac{PS}{2}$ and $\sigma = 0.3$. $CS = 50$ and 100 generations for the 10D and30D, respectively, and 150 for 50D and 100D problems. For the local search, cfe_{LS} was set to $0.2 \times FFE_{max}$ fitness evaluations.

A. Results

The computational results, i.e., $\left|f\left(\overrightarrow{x}_{best}\right) - f\left(\overrightarrow{x^*}\right)\right|$, where $\overrightarrow{x^*}$ is the best known solutions, of UMOEAs-II are shown in Table II. Note that if $\left|f\left(\overrightarrow{x}_{best}\right) - f\left(\overrightarrow{x^*}\right)\right| \leq 1e{-}08$ the result is considered zero.

For unimodal problems$(F_1 : F_3)$, the algorithm showed its robustness in obtaining the optimal solutions for F_2 and F_3 with all dimensions. For F_1, UMOEAs-II attained the optimality for 10D and 30D, while the results were very close to the optimal for the 50D and 100D problems.

Among the multi-modal problems $(F_{04} : F_{16})$, UMOEAs-II was successfully able to the obtain the optimal solution in

Table I. FITNESS VALUES OBTAINED BY UMOEAS-II FOR 10D AND 30D

Prob.	10D					30D				
	best	**worst**	**median**	**mean**	$std.$	**best**	**worst**	**median**	**mean**	$std.$
F_1	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_2	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_3	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_4	0.00000E+00	4.33541E+00	0.00000E+00	1.70016E-01	8.49910E-01	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_5	0.00000E+00	2.00000E+01	1.99891E+01	1.30965E+01	9.42357E+00	1.99967E+01	1.99998E+01	1.99988E+01	1.99987E+01	7.57474E-04
F_6	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	2.30978E-01	0.00000E+00	4.55869E-03	3.23392E-02
F_7	0.00000E+00	7.39604E-03	0.00000E+00	1.45020E-04	1.03565E-03	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_8	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_9	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	4.97480E+00	9.94959E-01	9.90721E-01	1.13478E+00
F_{10}	0.00000E+00	6.24544E-02	0.00000E+00	1.22460E-03	8.74537E-03	0.00000E+00	2.08192E-02	0.00000E+00	4.08221E-04	2.91528E-03
F_{11}	3.53987E+00	1.45715E+02	2.18246E+01	3.19259E+01	3.47228E+01	7.36666E+02	1.79305E+03	1.33217E+03	1.33109E+03	2.40293E+02
F_{12}	4.65576E-06	1.00148E-01	2.99739E-02	3.67203E-02	2.54924E-02	1.69259E-02	2.24366E-01	8.43935E-02	8.43281E-02	4.89799E-02
F_{13}	1.20732E-02	6.86941E-02	2.99086E-02	3.32557E-02	1.47985E-02	4.36819E-02	1.50057E-01	9.83499E-02	9.65645E-02	2.92333E-02
F_{14}	2.28377E-02	1.26281E-01	7.16255E-02	7.40414E-02	2.73709E-02	1.46760E-01	2.94806E-01	2.32791E-01	2.32496E-01	2.71667E-02
F_{15}	1.54398E-01	4.82588E-01	2.71272E-01	3.01865E-01	8.18759E-02	9.09751E-01	2.76953E+00	1.88781E+00	1.89331E+00	4.25870E-01
F_{16}	2.07192E-01	1.60920E+00	9.59188E-01	1.01464E+00	3.60657E-01	7.65912E+00	1.00590E+01	8.89732E+00	8.86487E+00	5.00857E-01
F_{17}	0.00000E+00	6.38604E+00	1.20310E+00	1.49207E+00	1.39362E+00	5.98374E+01	4.37953E+02	1.81004E+02	1.73590E+02	8.09263E+01
F_{18}	1.67124E-03	1.15995E+00	1.07153E-01	2.15008E-01	2.36169E-01	2.31506E+00	1.04155E+01	5.28570E+00	5.50465E+00	2.05382E+00
F_{19}	1.25611E-02	6.79869E-01	7.67777E-02	1.03739E-01	1.21823E-01	9.64560E-01	4.76157E+00	2.96737E+00	2.87044E+00	8.93773E-01
F_{20}	5.35679E-03	6.22503E-01	1.10292E-01	1.33174E-01	1.02789E-01	1.05488E+00	6.47270E+00	3.47235E+00	3.53112E+00	1.40652E+00
F_{21}	5.45384E-04	1.12719E+00	3.19717E-01	3.24303E-01	2.61552E-01	2.26343E+00	2.89144E+02	2.16344E+01	6.69862E+01	7.89957E+01
F_{22}	4.91850E-03	1.64855E-01	6.46664E-02	6.67014E-02	3.67610E-02	2.10229E+01	4.54041E+01	2.47109E+01	2.66688E+01	6.33044E+00
F_{23}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{24}	1.00000E+02	1.09508E+02	1.06701E+02	1.05204E+02	3.36758E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{25}	1.00000E+02	1.16649E+02	1.08441E+02	1.05792E+02	6.03468E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{26}	1.00011E+02	1.00067E+02	1.00034E+02	1.00036E+02	1.56246E-02	1.00035E+02	1.00177E+02	1.00104E+02	1.00099E+02	2.89612E-02
F_{27}	7.26250E-01	2.14662E+00	1.29537E+00	1.29906E+00	2.98026E-01	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{28}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{29}	1.31746E+02	2.23260E+02	2.21761E+02	2.20290E+02	1.26568E+01	7.13317E+02	7.25345E+02	7.15306E+02	7.16346E+02	2.69519E+00
F_{30}	4.54289E+02	4.65910E+02	4.62335E+02	4.62432E+02	1.32680E+00	4.16617E+02	1.98625E+03	9.01519E+02	9.18282E+02	3.46480E+02

all runs for F_8 with all dimensions. The algorithm was robust in solving F_7 with 30D, 50D and 100D. The same occurred for F_4 with 30 variables and F_6 with 10D. Regarding the remaining ones, the algorithm was able to obtain the optimal solutions in several occasions, with the average fitness values achieved were close to the optimal.

Considering the hybrid functions $(F_{17} : F_{22})$, the algorithm was able to obtain near-optimal average results for all problems except F_{11} and F_{17}. In F_{17}, although UMOEAs-II was able to obtain the optimal solution when D was 10, its performance was reduced when D increased. It was also noticed that F_{11} was difficult to solve, with a large average deviation from the optimal solution.

In regards to the composition problems $(F_{23} : F_{30})$, it was noted that all of these problems were difficult, as they contain a huge number of local optima, as a consequence UMOEAs-II became stuck. However, it was also noted that the algorithm converged to the same fitness value for several problems.

For a further illustration of the algorithm's performance, a few convergence plots are depicted in Fig. 1, which show the ability of UMOEAs-II to quickly converge to good solutions.

V. COMPUTATIONAL COMPLEXITY

The computational complexities of UMOEAs-II are calculated based on 10D, 30D, and 50D. Note that the algorithm was coded using Matlab R2015a, and was run on a PC with a 3.4 GHz Core I7 processor with 16 GB RAM, and Windows 7. A summary of the results obtained is shown in Table III. The results showed that the computational complexity of UMOEAs-II based on 10D was bigger than that on 30D and 50D. The reason for this was that the diversity measure was more frequently done for 10D problems than the other dimensions. It is worthy to mention that the computational complexity of UMOEAsII is better than that of the original variant.

Table III. COMPUTATIONAL COMPLEXITY OF UMOEAS-II

	T_0	T_1	$\hat{T}_2$	$\frac{\hat{T}_2 - T_1}{T_0}$
10D	0.108	0.113	4.3	38.769
30D		0.3149	3.153	26.279
50D		0.7009	3.8904	29.532

A. *Comparison with state-of-the-art algorithms*

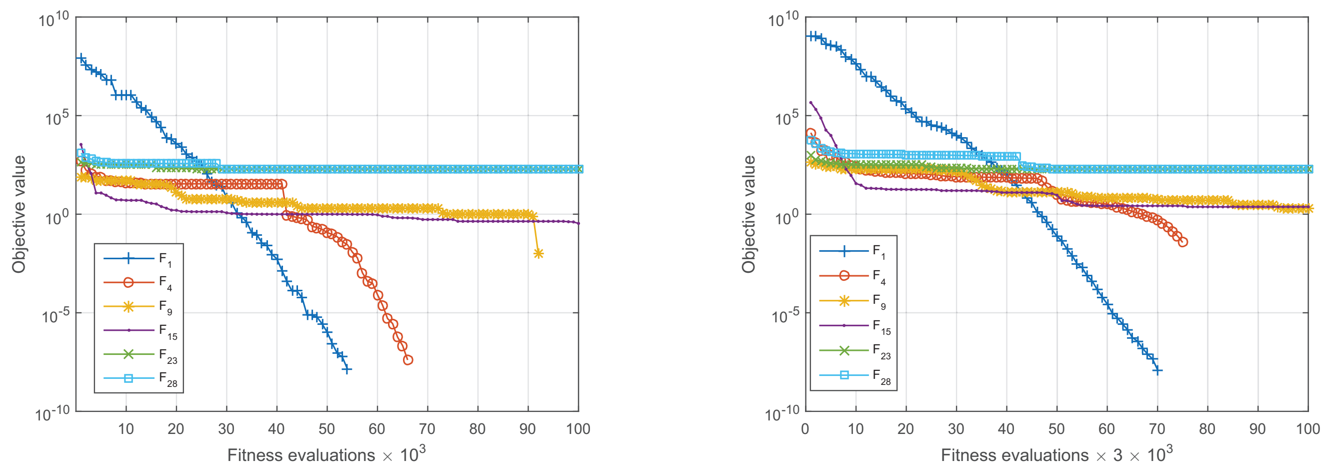
UMOEAs-II is compared with the best two algorithms solved the same problems, known as (1) UMOEAs [16]; and (2) LSHADE [24].

A comparison summary of the quality of solutions is provided in Table IV. Generally speaking, UMOEAs-II was able to obtain better average results than those of the other algorithms in the majority of test problems. Regarding the best fitness values of all algorithms, UMOEAs-II was consistently better than LSHADE for all dimensions. However, UMOEAs-II was competitive to UMOEAs.

Furthermore, the Wilcoxon test was carried out to find if there is a statistically significant difference between UMOEAs-II and the two other algorithms compared in this paper. Based on the results shown in IV, UMOEAs-II showed superior performance to that of LSHADE based on both the best and average fitness values obtained for all dimensions. In comparison with UMOEAs, UMOEAs-II was statistically better based on the average results attained. However, there was no significant difference between them considering the best fitness values obtained.

Table II. Fitness values obtained by UMOEAs-II for 50D and 100D, where * means that reported value $\left(\left|f\left(\overrightarrow{x}_{best}\right)-f\left(\overrightarrow{x^{*}}\right)\right|\right)$ is $\leq 1e-08$ and hence converted to zero

Prob.	50D					100D				
	best	**worst**	**median**	**mean**	*std.*	**best**	**worst**	**median**	**mean**	*std.*
F_1	2.10764E-04	6.70606E-03	1.82389E-03	1.83402E-03	1.14872E-03	2.32903E-03	5.34402E-03	4.02603E-03	3.93367E-03	6.26106E-04
F_2	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	1.48980E-07	0.00000E+00	0.00000E+00*	2.09743E-08
F_3	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	1.89377E-08	0.00000E+00	0.00000E+00*	4.42806E-09
F_4	0.00000E+00	9.83971E+01	0.00000E+00	3.84891E+01	4.83954E+01	0.00000E+00	2.12514E+02	1.43588E+02	1.62439E+02	4.43055E+01
F_5	1.99990E+01	1.99999E+01	1.99998E+01	1.99997E+01	1.98119E-04	1.99993E+01	1.99999E+01	1.99997E+01	1.99997E+01	1.23138E-04
F_6	9.60337E-05	2.20896E+00	6.96133E-02	2.03005E-01	4.50250E-01	1.72444E+00	1.89939E+01	9.39991E+00	9.79923E+00	3.25177E+00
F_7	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_8	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
F_9	9.94959E-01	8.95463E+00	3.97984E+00	4.10038E+00	1.94634E+00	1.79093E+01	3.48236E+01	2.48740E+01	2.52056E+01	4.17725E+00
F_{10}	9.99334E-02	2.69692E+00	2.33560E-01	5.43143E-01	6.34737E-01	6.57050E+00	3.75537E+02	1.62327E+01	7.94350E+01	9.77268E+01
F_{11}	2.61302E+03	4.54353E+03	3.48202E+03	3.47027E+03	4.82481E+02	8.83818E+03	1.34384E+04	1.13757E+04	1.12541E+04	9.20423E+02
F_{12}	1.67111E-02	2.79466E-01	1.23376E-01	1.30325E-01	7.14723E-02	5.38774E-02	5.57481E-01	2.77218E-01	2.63688E-01	1.18810E-01
F_{13}	4.17144E-02	2.11519E-01	1.57859E-01	1.46601E-01	3.84452E-02	8.53750E-02	2.87029E-01	2.17210E-01	1.93178E-01	5.78556E-02
F_{14}	2.67232E-01	3.44424E-01	3.05914E-01	3.04112E-01	2.28437E-02	1.92128E-01	2.45365E-01	2.19582E-01	2.19132E-01	1.31481E-02
F_{15}	3.53407E+00	7.36755E+00	5.31240E+00	5.28756E+00	8.16928E-01	8.96529E+00	1.67820E+01	1.23228E+01	1.25674E+01	1.80341E+00
F_{16}	1.70328E+01	2.05553E+01	1.81778E+01	1.82636E+01	7.39213E-01	3.80698E+01	4.37862E+01	4.21409E+01	4.17266E+01	1.45517E+00
F_{17}	3.44687E+02	1.79954E+03	1.09216E+03	1.08920E+03	3.44690E+02	3.07609E+03	5.67787E+03	4.31059E+03	4.29244E+03	6.35379E+02
F_{18}	3.80752E+01	9.78314E+01	7.39037E+01	7.27632E+01	1.73998E+01	1.92477E+02	2.49102E+02	2.15375E+02	2.17548E+02	1.41319E+01
F_{19}	5.37878E+00	1.20269E+01	9.29251E+00	8.78672E+00	1.97858E+00	9.05985E+01	1.02185E+02	9.58059E+01	9.55452E+01	2.30569E+00
F_{20}	6.95334E+00	2.33249E+01	1.28911E+01	1.36263E+01	3.95516E+00	7.04263E+01	2.16315E+02	1.30004E+02	1.35032E+02	3.41115E+01
F_{21}	2.51889E+02	7.91607E+02	4.89106E+02	4.52103E+02	1.32762E+02	1.08878E+03	2.88559E+03	1.99053E+03	1.97198E+03	4.46312E+02
F_{22}	2.68846E+01	3.67358E+02	1.85169E+02	1.98245E+02	9.63166E+01	3.77482E+02	1.76186E+03	1.11985E+03	1.11287E+03	3.20006E+02
F_{23}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{24}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	2.54815E-13	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	6.75095E-12
F_{25}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{26}	1.00056E+02	1.00193E+02	1.00140E+02	1.00139E+02	3.56089E-02	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{27}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{28}	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00	2.00000E+02	2.00000E+02	2.00000E+02	2.00000E+02	0.00000E+00
F_{29}	7.23436E+02	9.25158E+02	7.90307E+02	8.08451E+02	4.64272E+01	7.15684E+02	1.47041E+03	8.32054E+02	8.87935E+02	1.69576E+02
F_{30}	7.96751E+03	1.02237E+04	8.53975E+03	8.61878E+03	4.64757E+02	5.89928E+03	9.46481E+03	8.09283E+03	8.09357E+03	8.58189E+02



(a) 10D (b) 30D

Figure 1. Convergence plots of UMOEAs-II for 6 test problems with 10D and 30D

Table IV. Comparison summary of UMOEAs-II against UMOEAs and LSHADE (Dec. is the statistical decision taken based on Wilcoxon test)

Algorithms	Criteria	10D				30D				50D				100D			
		Better	**Similar**	**Worse**	**Dec.**	**Better**	**Similar**	**Worse**	**Dec.**	**Better**	**Similar**	**Worse**	**Dec.**	**Better**	**Similar**	**Worse**	**Dec.**
UMOEAs-II vs. UMOEAs	**Best fitness values**	6	13	11	≈	11	9	10	≈	15	6	9	≈	N/A	N/A	N/A	N/A
	Average fitness values	19	5	6	+	19	5	6	+	22	3	5	+	N/A	N/A	N/A	N/A
UMOEAs-II vs. LSHADE	**Best fitness values**	15	11	4	+	16	8	6	+	18	4	8	+	18	5	7	+
	Average fitness values	23	4	3	+	20	6	4	+	19	3	8	+	18	5	7	+

VI. Conclusions

Many EAs have been introduced for solving unconstrained optimization problems. Among them, the multi-operator and multi-method algorithms showed great success, which motivated researchers to combine them within a single framework. In this paper, we adopted the concept of multiple algorithms empowered by multiple operators, in which the initial population was divided into two sub-populations, each of which was independently evolved using either a multi-operator DE or CMA-ES. The probability of applying each algorithm was dynamically changing based on the quality of solutions as well as the diversity. In addition, an information sharing scheme was adopted, and to enhance the intensification capability of the proposed algorithm, a local search was used are the later stages of the optimization process.

The algorithm was tested on the CEC2016 real-parameter benchmark problems and showed its ability to obtain high-quality solutions. Furthermore, it was found superior to the best two algorithms in the literature.

References

[1] K. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media, 2006.

[2] 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.

[3] L. Davis *et al.*, *Handbook of genetic algorithms*. Van Nostrand Reinhold New York, 1991, vol. 115.

[4] 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.

[5] 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.

[6] S. M. E. Elsayed, "Evolutionary approach for constrained optimization," Ph.D. dissertation, University of New South Wales at the Australian Defence Force Academy at Canberra, 2012.

[7] J. A. Vrugt and B. A. Robinson, "Improved evolutionary optimization from genetically adaptive multimethod search," *Proceedings of the National Academy of Sciences*, vol. 104, no. 3, pp. 708–711, 2007.

[8] J. A. Vrugt, B. A. Robinson, and J. M. Hyman, "Self-adaptive multi-method search for global optimization in real-parameter spaces," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 243–259, 2009.

[9] F. Peng, K. Tang, G. Chen, and X. Yao, "Population-based algorithm portfolios for numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 5, pp. 782–800, 2010.

[10] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[11] A. Zamuda and J. Brest, "Population reduction differential evolution with multiple mutation strategies in real world industry challenges," in *Swarm and Evolutionary Computation*, L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. Zadeh, and J. Zurada, Eds. Springer, 2012, pp. 154–161.

[12] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[13] J. Brest and M. S. Maučec, "Population size reduction for the differential evolution algorithm," *Applied Intelligence*, vol. 29, no. 3, pp. 228–247, 2008.

[14] J. Brest, B. Boskovic, A. Zamuda, I. Fister, and E. Mezura-Montes, "Real parameter single objective optimization using self-adaptive differential evolution algorithm with more strategies," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 377–383.

[15] T. J. Choi and C. W. Ahn, "An adaptive cauchy differential evolution algorithm with population size reduction and modified multiple mutation strategies," in *Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems-Volume 2*. Springer, 2015, pp. 13–26.

[16] S. M. Elsayed, R. Sarker, D. L. Essam, N. M. Hamza *et al.*, "Testing united multi-operator evolutionary algorithms on the cec2014 real-parameter numerical optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2014, pp. 1650–1657.

[17] J. Liang, B. Qu, and P. Suganthan, "Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization," *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2013.

[18] R. Storn and K. Price, *Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces*. ICSI Berkeley, 1995, vol. 3.

[19] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[20] D. Zaharie, "A comparative analysis of crossover variants in differential evolution," *Proceedings of IMCSIT*, pp. 171–181, 2007.

[21] N. Hansen, "Benchmarking a bi-population cma-es on the bbob-2009 function testbed," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*. ACM, 2009, pp. 2389–2396.

[22] M. W. Iruthayarajan and S. Baskar, "Covariance matrix adaptation evolution strategy based design of centralized pid controller," *Expert systems with Applications*, vol. 37, no. 8, pp. 5775–5781, 2010.

[23] 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.

[24] R. Tanabe and A. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *IEEE Congress on Evolutionary Computation*, July 2014, pp. 1658–1665.

[25] R. Sarker, S. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 5, pp. 689–707, Oct 2014.

[26] Y. Nesterov, A. Nemirovskii, and Y. Ye, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994, vol. 13.