ENHANCED SHADE AND REAL-WORLD OPTIMIZATION PROBLEMS

Petr Bujok, Josef Tvrdik
Department of Computer Science, University of Ostrava, Czech Republic
petr.bujok@osu.cz, josef.tvrdik@osu.cz

Abstract An enhanced adaptive differential evolution algorithm is described and applied to the CEC 2011 test suite of real-world optimization problems. The new algorithm combines success-history based adaptation known from SHADE and the competition of the various strategies in differential evolution. The comparison of the newly proposed algorithm (called SHADE4) with the algorithm winning in CEC 2011 competition shows that the new algorithm performs significantly better in 11 out of 22 test problems and worse only in four problems. The performance of SHADE4 was also significantly better than the original SHADE. The new algorithm is almost control-parameter free, which is helpful to its usage in the solution of the real-world problems.

Keywords: CEC 2011 real-world test problems, Competing strategies, Differential evolution, Experimental comparison, Success-history adaptation.

1. Introduction

A new adaptive variant of differential evolution (DE) (called SHADE4 hereafter) is applied to the real-world optimization problems collected in the CEC 2011 test suite [5]. Our experimental results are compared with the results of the standard DE/randrl/1/bin algorithm, the winner of CEC 2011 competition of algorithm [7] and the original SHADE algorithm. SHADE4 algorithm combines two adaptive mechanisms, namely the success-history based parameter adaptation used in SHADE [17] and the competition of strategies proposed in [18]. The aim of the paper is to present the performance of SHADE4 algorithm on the real-world optimization test suite and to demonstrate that the research of new adaptive variants of evolutionary algorithms is beneficial for real-world applications.

We consider the single-objective global optimization problem with the bound constraints defined as follows. The cost function to be minimized is $f(\mathbf{x}),\ \mathbf{x} = (x_1,\ x_2,\ \ldots, x_D) \in \mathbb{R}^D$. The domain of a feasible solutions Ω is constrained by bounds, a lower limit (a_j) and an upper limit (b_j), $\Omega = \prod_{j=1}^{D}[a_j,\ b_j]$, $a_j < b_j$, $j = 1,\ 2,\ \ldots, D$. The global minimum point $\mathbf{x}^*$ satisfying condition $f(\mathbf{x}^*) \leq f(\mathbf{x}),\ \forall \mathbf{x} \in \Omega$, is the solution of the problem.

The rest of the paper is organized in the following manner. The basic scheme of DE algorithm is shown in Section 2. An adaptive SHADE, from which the new algorithm was inspired, is presented in Section 3. A newly proposed SHADE4 algorithm is described in Section 4. Settings of experiments and the results are given in Section 5 and 6. Finally, the conclusions are made in Section 7.

2. Differential Evolution

Differential evolution introduced by Storn and Price in [15] is a population-based evolutionary algorithm for problems with a real-valued cost function. The population P of the size N is developed step-by-step from a generation to a generation. DE uses evolutionary operators, i.e., mutation, crossover, and selection that are applied in the development of new generation of P.

The new trial point $\mathbf{y}$ is created from a mutant point $\mathbf{u}$ generated by using a kind of the mutation and from the current point of the population $\mathbf{x}_i$ by the application of the crossover. A combination of the mutation and the crossover variant is usually called DE strategy. A better point from the pair of $\mathbf{x}_i,\ \mathbf{y}$, based on the value of the cost function, is selected for the new generation.

The DE algorithm has only a few parameters whose settings significantly influence the ability to solve various optimization problems. These control parameters and their settings have been studied intensively in recent years. A comprehensive summary of advanced results in DE research is available in [6, 11], where several kinds of the mutation and the crossover were listed and some adaptive or self-adaptive DE variants are described. Adaptive variants of DE, e.g., [1, 2, 8, 13, 23] enable the change of DE control-parameters during the run of the algorithm to the current problem without trial-and-error tuning of the control parameters. Some other adaptive DE variants like [10, 14, 21] use several DE strategies or control-parameter settings and select among them adaptively with respect to the previous progress of the search.

3. SHADE Variant of Differential Evolution

Success-History Based Parameter Adaptation for Differential Evolution (SHADE) algorithm was proposed by Tanabe and Fukunaga in 2013 [17] and proved to be the best performing DE variant in CEC 2013 competition. SHADE was derived from the original algorithm Adaptive Differential Evolution With Optional External Archive (JADE) proposed by Zhang and Sanderson in 2009 [25]. The main extension of SHADE compared to original JADE is in a history-based adaptation of the control parameters F and CR. Both JADE and SHADE variants use an efficient *current-to-pbest* mutation variant, where the new mutant vector $\mathbf{u}_i$ is generated from four mutually different individuals of P – the current individual $\mathbf{x}_i$, randomly chosen individual from the p best points $\mathbf{x}_{pbest}$, randomly selected point $\mathbf{x}_{r1}$ from P and randomly selected point $\mathbf{x}_{r2}$ from $P \bigcup A$, as it is formed in the equation (1). A unification of the current population P and archive population A of old outperformed promising solutions aims at preventing DE algorithm to stuck in the local minimum.

$$\mathbf{u}_i = \mathbf{x}_i + F_i\ (\mathbf{x}_{pbest} - \mathbf{x}_i) + F_i\ (\mathbf{x}_{r1} - \mathbf{x}_{r2}) \tag{1}$$

where the point $\mathbf{x}_{pbest}$ is randomly selected from the p best points of P and F_i is a scale factor ($0 \leq F \leq 1$) which value is adapted during the search as is described in Algorithm 1.

Differently to JADE, the parameter of p for selecting the $\mathbf{x}_{pbest}$ in SHADE is generated randomly for each point of the population from uniform distribution according to the equation (2):

$$p_i = rand[2/N,\ 0.2], \tag{2}$$

and $\mathbf{x}_{pbest}$ is a point selected from the $p_i \times 100\,\%$ best points of P. After the mutation, a binomial crossover operation with CR parameter, ($0 \leq CR \leq 1$) which value is adapted during the search as is described in Algorithm 1, is used for generating the trial point $\mathbf{y}_i$ (3).

$$\mathbf{y}_{i,j} = \begin{cases} \mathbf{u}_{i,j} & \textit{if } rand_j(0,1) \leq CR \textit{ or } j = rand(1,D) \\ \mathbf{x}_{i,j} & \textit{otherwise.} \end{cases} \tag{3}$$

The changed coordinates are dispersed uniformly over the dimensions $1\ , 2, \ldots,\ D$. The crossover operator combines selected elements of mutant vector $\mathbf{u}_i$ and the current individual $\mathbf{x}_i$ in order to get a trial vector $\mathbf{y}_i$. SHADE algorithm uses only *current-to-pbest/bin* DE strategy.

If the function value of the newly composed trial point $f(\mathbf{y}_i)$ is better than the $f(\mathbf{x}_i)$, the new point $\mathbf{y}_i$ replaces the old one and the old solution

$\mathbf{x}_i$ is inserted into archive A. The values of the successfully used control parameters F and CR are also stored into auxiliary memories S_F and S_{CR}.

When the size of the archive exceeds the size of the population P, excessive randomly chosen points are deleted from the archive A. The purpose of the archive is to store old solutions from the previous generations and use them for the mutation according to (1).

The circle memories M_F and M_{CR} for generating new values of the control parameters F and CR are updated on the current position t, based on the stored successful values from auxiliary memories S_F and S_{CR}. In particular, the new value of M_{CR} is computed as a weighted arithmetic mean of the current values in S_{CR} (4) with the weights w_k (6) and the new value of M_F is computed as weighted Lehmer mean of the current values from S_F (5).

$$M_{F,t} = \frac{\sum_{k=1}^{|S_F|} w_k \; S_{F,k}^2}{\sum_{k=1}^{|S_F|} w_k \; S_{F,k}}, \tag{4}$$

$$M_{CR,t} = \sum_{k=1}^{|S_{CR}|} w_k \; S_{CR,k}, \tag{5}$$

$$w_k = \frac{\Delta f_k}{\sum_{j=1}^{|S_{CR}|} \Delta f_j}. \tag{6}$$

The values of Δf_k are computed as a difference between the cost function of the current point $\mathbf{x}_k$ and the new trial point $\mathbf{y}_k$. The values of M_F and M_{CR} remain the same if no successful point was created in the last generation. Update of values in M_F and M_{CR} is carried out at the position t, $t = 1$ at the beginning of the search and t is incremented by one after each updating of an element of M_F and M_{CR}. If $t > H$ then t is reset to $t = 1$, where H is the size of M_F and M_{CR}.

SHADE algorithm was the best performing DE variant in CEC 2013 competition [16]. However, even SHADE fails in some hard problems defined by composition of several multi-modal functions, especially in rotated functions of high level of dimension.

4. Competing Strategies in SHADE Algorithm

We studied different types of mutation and crossover. In SHADE, only one DE strategy, namely the *current-to-pbest/bin* DE strategy with the binomial crossover (3) is used. Some results [19, 24] show that the exponential type of crossover (9) or a different variant of mutation can be efficient in problems where the standard variant with binomial crossover

performs poorly. It was shown that even only replacement of the binomial crossover with the exponential one can increase performance of the SHADE algorithm [4]. Based on these results, we propose an enhanced SHADE algorithm with the competition of DE strategies. We suppose that the competition of different DE strategies can increase the efficiency of the search in situations, where variant with only one strategy fails.

The scheme for selecting variant of the DE strategies is taken from the competitive DE [18, 20]. Let us have K strategies and each DE strategy has its value of probability q_k to be used and the values of q_k are updated according to the success of the corresponding strategy in previous generations. Thereafter, a more successful strategy is used more likely than a strategy which is rarely able to generate successful individuals.

$$q_k = \frac{n_k + n_0}{\sum_{j=1}^{K} (n_j + n_0)}, \tag{7}$$

where q_k is probability of use of the kth DE strategy, n_k is the current count of the kth DE strategy successes, n_0 is an input parameter to prevent from a dramatic change in q_k and all probabilities are reset to starting uniformly distributed values if any q_k decreases below a given δ, $\delta > 0$.

Beside the *current-to-pbest/bin* strategy used in SHADE the strategies using different mutation and different crossover are selected for competition. The mutation (called *randrl/1*) which was proposed by Kaelo and Ali [9] has appeared efficient [3, 12, 19]. This mutation scheme (8) is based on the popular *rand/1* mutation, where three mutually different points $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$ that are randomly selected from P.

$$\mathbf{u} = \mathbf{r}_1^x + F\,(\mathbf{r}_2^x - \mathbf{r}_3^x), \tag{8}$$

where in contrast to *rand/1* the point $\mathbf{r}_1^x$ is tournament best among $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$, i.e., $f(\mathbf{r}_1^x) \leq f(\mathbf{r}_j^x),\ j = 2, 3$.

The exponential crossover in DE is defined by the following rule:

$$\mathbf{y}_{i,j} = \begin{cases} \mathbf{u}_{i,j} & \textit{for } j = \langle n \rangle_D,\ \langle n+1 \rangle_D, \ldots, \langle n+L-1 \rangle_D \\ \mathbf{x}_{i,j} & \textit{otherwise}, \end{cases} \tag{9}$$

where the brackets $\langle\rangle_D$ represent the modulo function with modulus D. The starting position of crossover (n) is chosen randomly from $\{1, \ldots, D\}$, and L consecutive elements (counted in a circular manner) are taken from the mutant vector $\mathbf{u}_i$. Thus, L adjacent elements are changed in exponential crossover and the probability of replacing the kth element in the sequence $1, 2, \ldots, L$, $L \leq D$, decreases exponentially

with increasing k in dependence on CR, next element is replaced if $rand_j(0,1) < CR$. The relationship between CR and the mean value of L was derived by Zaharie [24].

When we combine the aforementioned types of mutation and crossover, we obtain four different DE strategies to compete. Other components of SHADE algorithm remain unchanged, i.e., archive A of outperformed good solutions and way of adaptation of the control parameters F and CR. All the strategies in the competition use the common M_F and M_{CR} memories.

The advantage of this scheme is ability to adapt the search to the currently solved problem. A pseudo-code of SHADE variant with competition of four DE strategies is illustrated in Algorithm 1. Operators randn and randc generate random numbers from normal and Cauchy distribution, respectively.

At the beginning, a population of N potential solutions is generated distributed uniformly in the search area. Then all H values of both historical memories $M_{CR,t}$ and $M_{F,t}$, $t = 1, 2, \ldots, H$ for generating the control parameters CR and F are initialized to 0.5, where the learning rate H is an input parameter. The archive A for storing the old good solutions is set as empty.

Temporary memories S_F and S_{CR} for saving the successful values of the control parameters F and CR are set empty in each generation. The values of distribution parameters needed for random updating of F and CR are chosen randomly from M_F and M_{CR} for each point of the population. F is generated from Cauchy distribution with the parameters $(M_{F,r_i}, 0.1)$ and CR is generated from the normal distribution with the mean value M_{CR,r_i} and standard deviation 0.1, where r_i is randomly selected index from $\{1, 2, \ldots, H\}$. The unfeasible values of F and CR are regenerated according to the rules of JADE algorithm [25].

After setting the control parameters of mutation and crossover, a new trial vector $\mathbf{y}_i$ is constructed applying the selected DE strategy (one of the four in the competition) to the current parent vector $\mathbf{x}_i$. It is obvious that only *current-to-pbest* mutation variant can use archive A. On the other hand, any of applied the four DE strategies can insert outperformed old parent vector $\mathbf{x}_i$ into archive A.

At the beginning of SHADE4, the counters of successes of the DE strategies are set to zero, which results in equal values $q_k = 1/K$ for $k = 1, 2, \ldots, K$. $K = 4$ is the number of the DE strategies. After each successful generating a trial point $\mathbf{y}_i$, the counter of DE strategy currently used is increased by one. After each generation, the probabilities are updated based on the counters of the successes (7).

Algorithm 1 SHADE with competition of DE strategies (SHADE4)

initialize population $P = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$
evaluate $f(\mathbf{x}_i),\ i = 1, 2, \ldots, N$
set all M_{CR} and M_F to 0.5
initialize empty archive A
while $FES < MaxFES$ **do**
  initialize S_{CR}, S_F empty
  for $i = 1, 2, \ldots, N$ **do**
    select r_i randomly from $\{1, 2, \ldots, H\}$
    $CR_i = \text{randn}(M_{CR,r_i}, 0.1)$
    $F_i = \text{randc}(M_{F,r_i}, 0.1)$
    $p_i = \text{rand}[2/N,\ 0.2]$
    generate trial vector $\mathbf{y}_i$ by selected DE strategy (roulette wheel)
    if $f(\mathbf{y}_i) \leq f(\mathbf{x}_i)$ **then**
      insert $\mathbf{y}_i$ into Q
    else
      insert $\mathbf{x}_i$ into Q
    end if
    if $f(\mathbf{y}_i) < f(\mathbf{x}_i)$ **then**
      $\mathbf{x}_i \rightarrow A$
      $CR_i \rightarrow S_{CR}$, $F_i \rightarrow S_F$
      increase counter of the DE strategy success
    end if
  end for
  if size(A) > size(P) **then**
    delete randomly chosen individuals from A
  end if
  if size$(S_{CR}) > 0$ **and** size$(S_F) > 0$ **then**
    update M_{CR} and M_F based on S_{CR} and S_F
  end if
  $P \leftarrow Q$
end while

Finally, M_F and M_{CR} memories are updated using the stored successful values in S_F and S_{CR} as it is described in Section 3. The values of M_F and M_{CR} remain the same if no successful point was created in the last generation. The main cycle of the SHADE4 algorithm is repeated until the stopping condition is achieved. Matlab source code of SHADE4 algorithm is available online on web site `www1.osu.cz/~bujok/`.

Table 1: Function values obtained from 25 runs of standard DE/randrl/1/bin

Problem	*D*	*Best*	*Worst*	*Median*	*Mean*	*Std*
T01	6	0	1.04E-20	0	4.34E-22	2.09E-21
T02	30	-9.63396	-4.61191	-7.17815	-7.18629	0.99431
T03	1	1.15E-05	1.15E-05	1.15E-05	1.15E-05	5.19E-21
T04	1	0	0	0	0	0
T05	30	-22.9184	-17.9343	-19.277	-19.5956	1.30424
T06	30	-18.3201	-11.9093	-14.0873	-14.4051	1.61769
T07	20	1.49985	1.91389	1.72156	1.72729	0.10434
T08	7	220	220	220	220	0
T09	126	288886	525927	392200	403142	52960.3
T10	12	-21.537	-21.0853	-21.3571	-21.351	0.12976
T11.1	120	2.15E+07	7.48E+08	4.62E+07	4.56E+07	1.29E+07
T11.2	240	5.06E+06	6.66E+06	5.62E+06	5.74E+06	4.78E+05
T11.3	6	15444.2	15444.2	15444.2	15444.2	5.57E-12
T11.4	13	18982.1	19691	19211	19221.7	160.566
T11.5	15	32914.3	33108.8	32972.6	32975.4	44.5107
T11.6	40	135262	142690	138479	138718	2296.81
T11.7	140	3.56E+06	3.70E+07	1.15E+07	1.24E+07	7.69E+06
T11.8	96	4.00E+06	7.65E+06	5.84E+06	5.73E+06	959208
T11.9	96	3.13E+06	8.27E+06	5.53E+06	5.82E+06	1.30E+06
T11.10	96	2.68E+06	7.24E+06	4.78E+06	4.73E+06	1.17E+06
T12	26	24.7192	32.1438	28.0203	28.1726	1.80743
T13	22	11.3761	26.1714	13.3083	14.3622	3.12482

5. Experiments

The test suite of 22 real-world problems proposed for CEC 2011 Special Session on Real-Parameter Numerical Optimization is used as a benchmark in the experimental comparison. The test functions are described in [5], including the experimental settings required for the competition and the source code of the functions is available on the web site given in this report.

Four algorithms are compared in this paper. One of them is GA-MPC [7], which is the winner of CEC 2011 competition, and its results are taken from the cited paper.

The standard DE (DE/randrl/1/bin), the original SHADE and SHADE4 algorithms are the other algorithms in the comparison. They are implemented in Matlab 2010a and this environment was used for experiments. All computations were carried out on a standard PC with Windows 7, Intel(R) Core(TM)2 CPU 6320, 1.86GHz 1.87GHz, 2GB RAM.

The control parameters of the standard DE are set to $F = 0.8$, $CR = 0.8$. The control parameters of the original SHADE and SHADE4 are set

Table 2: Function values obtained from 25 runs of GA-MPC [7]

Problem	*Best*	*Worst*	*Median*	*Mean*	*Std*
T01	0	**0**	0	**0**	0
T02	-28.4225	**-27.113**	**-27.4797**	**-27.7007**	0.46731
T03	1.15E-05	1.15E-05	1.15E-05	1.15E-05	0
T04	13.77076	14.32911	13.77076	13.8154	0.1546
T05	-36.8454	-34.1076	-35.0098	-35.0388	0.83293
T06	-29.1661	-21.2585	-27.4298	-27.4881	1.78214
T07	0.5	0.93343	0.75806	0.74841	0.12491
T08	220	220	220	220	0
T09	**466.763**	**1818.26**	**1171.82**	**1220.59**	361.119
T10	-21.8425	**-21.4757**	-21.6445	-21.7022	0.11635
T11.1	**50925.1**	**52745.0**	**52002.8**	**52054.6**	449.912
T11.2	**1.07E+06**	1.08E+06	**1.07E+06**	**1.07E+06**	1617.59
T11.3	15444.2	15444.2	15444.2	15444.2	1.75E-07
T11.4	18100.6	18375.8	18278.2	18261.0	69.8163
T11.5	32723.8	32823.0	32770.8	32769.8	26.8249
T11.6	129213	136059	133186	133230	1878.80
T11.7	1.92E+06	1.97E+06	1.96E+06	1.95E+06	14083.6
T11.8	949500	995043	969605	971289	10387.4
T11.9	972102	1.21E+06	1.05E+06	1.06E+06	57041.6
T11.10	946598	995008	975758	975109	11848.9
T12	**7.09556**	**16.9249**	**13.7260**	**12.8182**	3.24134
T13	**8.39869**	**10.8102**	**8.62031**	**9.35934**	0.94543

to the values given in Section 3 and 4 including the learning rate the same as the population size, $H = N$. The population size is set to $N = 90$ (SHADE4) and $N = 100$ (the original SHADE) for all the test problems without respect to the problem dimension. The parameters controlling the competition of DE strategies in SHADE4 are set as follows: $n_0 = 2$ and $\delta = 1/(4 * 5) = 0.05$.

The tests were carried out with 25 independent runs per each test function. The run of the algorithm stops if the prescribed amount of function evaluation $MaxFES = 150000$ is reached. The point in the terminal population with the smallest function value is the solution of the problem found in the run.

6. Results

The basic characteristics of the function values found by the compared algorithms are presented in Table 1, 2, 3 and 4. The dimensions of the test problems are shown in Table 1. The function values, where an algorithm achieved less value of the characteristic than remaining

Table 3: Function values obtained from 25 runs of SHADE4

Problem	*Best*	*Worst*	*Median*	*Mean*	*Std*	*Sign.*
T01	0	1.97E-15	0	7.98E-17	3.93E-16	≈
T02	-28.4225	-22.4173	-24.1182	-24.7689	1.52554	−
T03	1.15E-05	1.15E-05	1.15E-05	1.15E-05	5.19E-21	≈
T04	**0**	**0**	**0**	**0**	0	+
T05	**-36.8955**	**-35.5469**	**-36.5439**	**-36.5116**	0.37292	+
T06	-29.1661	**-29.1388**	**-29.1661**	**-29.165**	5.46E-03	+
T07	0.5	**0.85883**	**0.61107**	**0.62462**	0.10125	+
T08	220	220	220	220	0	≈
T09	1393.46	3207	1979.9	2104.18	571.735	−
T10	-21.8425	-21.4421	**-21.8424**	**-21.7321**	0.12753	≈
T11.1	51003.9	53613.4	52513.6	52374.8	687.639	≈
T11.2	1.07E+06	**1.08E+06**	1.07E+06	1.07E+06	2071.30	≈
T11.3	15444.2	15444.2	15444.2	15444.2	5.57E-12	≈
T11.4	18056.1	18260.4	**18114.9**	**18128.9**	46.6809	+
T11.5	**32694.6**	**32798.7**	**32746**	**32740.8**	27.8249	+
T11.6	127718	**131474**	129911	129722	1095.99	+
T11.7	1.89E+06	1.94E+06	1.90E+06	1.91E+06	12769.8	+
T11.8	**936218**	946720	939786	**940219**	2540.09	+
T11.9	941254	1.15E+06	977446	987332	47146.4	+
T11.10	**933823**	**944030**	**939680**	**939613**	2610.84	+
T12	13.185	22.8099	18.5719	18.2831	2.30866	−
T13	8.62086	21.6398	14.9047	15.6816	3.88541	−

two others, are emphasized by bold print. Notice that the standard DE/randrl/1/bin never outperforms the GA-MPC, the original SHADE and SHADE4 simultaneously. The differences in the mean performance of GA-MPC and SHADE4 were assessed statistically by two-sample t-test at the significance level of 0.01. The t-test was used due to the fact that only summary results from [7] were available for GA-MPC. The results are depicted in the last column of Table 3. The symbol of "+" denotes that SHADE4 outperforms GA-MPC significantly, the symbol of "−" marks better performance of GA-MPC, and "≈" is used when there is no significant difference between the two algorithms.

The performance of SHADE4 and the original SHADE were compared by Wilcoxon two-sample test. The results are shown in the last column of Table 4. The symbol of "+" denotes that SHADE outperforms SHADE4 significantly, the symbol of "−" marks better performance of SHADE4. SHADE4 outperformed significantly the original SHADE in 7 out of 22 test problems, while the original SHADE was better only in one problem. In the rest of the problems, the performance was not significantly different.

Table 4: Function values obtained from 25 runs of SHADE

Problem	*Best*	*Worst*	*Median*	*Mean*	*Std*	*Sign.*
T01	0.02737	11.5882	0.13659	0.75903	2.29845	−
T02	-24.4943	-21.6473	-22.9994	-23.0316	0.74605	−
T03	1.15E-05	1.15E-05	1.15E-05	1.15E-05	5.19E-21	≈
T04	0	0	0	0	0	≈
T05	-36.6566	-34.1495	-36.3187	-36.0052	0.63807	−
T06	-29.142	-28.609	-29.092	-29.0486	0.12389	−
T07	0.90641	1.23144	1.13808	1.12130	0.08344	−
T08	220	220	220	220	0	≈
T09	1141.14	4169.19	2132.08	2228.75	714.195	≈
T010	-21.8425	-21.216	-21.6444	-21.6289	0.14074	−
T11.1	51559.6	71320.6	52580.5	53225.2	3803.80	≈
T11.2	1.07E+06	1.30E+06	1.08E+06	1.10E+06	53420.6	≈
T11.3	15444.2	15444.2	15444.2	15444.2	5.57E-12	≈
T11.4	**18028.6**	**18224.9**	18139.9	18130.7	51.7471	≈
T11.5	32742.9	32867.6	32749.6	32760	28.0246	≈
T11.6	**126951**	132462	**129061**	**129231**	1585.63	≈
T11.7	**188E+06**	**1.94E+06**	**1.90E+06**	**1.91E+06**	14075.4	≈
T11.8	937267	**944437**	**939593**	940475	2161.43	≈
T11.9	**939771**	**987681**	**948911**	**952462**	12130.8	+
T11.10	934264	945774	941238	940667	2991.61	≈
T12	14.1104	20.9105	17.5435	17.5903	1.42842	≈
T13	13.1146	23.6202	20.1044	19.9439	2.87591	−

7. Conclusion

The newly proposed SHADE4 algorithm outperformed the winner of CEC 2011 competition in 11 out of 22 problems significantly and the performance was the same in 7 problems. SHADE4 was outperformed only in four test problems. Compared to the original SHADE, SHADE4 performed better in 7 problems and it was defeated only in one problem. The results demonstrate that the development of new adaptive algorithms is beneficial to the utilization in solving the real-world optimization problems.

Moreover, SHADE4 has much less control parameters to set compared to GA-MPC when solving an optimization problem. The application of such almost control-parameter free algorithms is more convenient to the real-world users.

Acknowledgment: This work was supported by University of Ostrava from the project SGS08/UVAFM/2016.

References

[1] J. Brest, S. Greiner, B. Bosković, M. Mernik and V. Žumer. Self-adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. In *IEEE Transactions on Evolutionary Computation*, 10:646–657, 2006.

[2] J. Brest, A. Zamuda, B. Bosković, M. Maučec and V. Žumer. Dynamic optimization using self-adaptive differential evolution. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 415–422, 2009.

[3] P. Bujok and J. Tvrdík. A Comparison of Various Strategies in Differential Evolution. *Proceedings of the International Conference on Soft Computing MENDEL*, pages 48–55, 2011.

[4] P. Bujok and J. Tvrdík. Adaptive differential evolution: SHADE with competing crossover strategies. *Lecture Notes in Computer Science*, 9119:329–339, 2015.

[5] S. Das and P. N. Suganthan. Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems. Technical report, Jadavpur University, India and Nanyang Technological University, Singapore, 2010.

[6] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15:27–54, 2011.

[7] S. Elsayed, R. Sarker and D. Essam. GA with a New Multi-Parent Crossover for Solving IEEE-CEC2011 Competition Problems. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1034–1040, 2011.

[8] S. M. Islam, S. Das, S. Ghosh, S. Roy and P. N. Suganthan. An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: (Cybernetics)*, 42(2):482–500, 2012.

[9] P. Kaelo and M. M. Ali. A numerical study of some modified differential evolution algorithms. *European Journal of Operational Research*, 169:1176–1184, 2006.

[10] R. Mallipeddi, P. N. Suganthan, Q. K. Pan and M. F. Tasgetiren. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, 11:1679–1696, 2011.

[11] F. Neri and V. Tirronen. Recent advances in differential evolution: a survey and experimental analysis. *Artificial Intelligence Review*, 33:61–106, 2010.

[12] R. Poláková. A Comparison of Two Similar Mutation Operators in Differential Evolution. *Proceedings of the International Conference on Soft Computing MENDEL*, pages 1–6, 2015.

[13] W. Qian and A. Li. Adaptive differential evolution algorithm for multiobjective optimization problems. *Applied Mathematics and Computation*, 201(12):431–440, 2008.

[14] 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*, 13(2):398–417, 2009.

[15] R. Storn and K. V. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[16] R. Tanabe and A. Fukunaga. Evaluating the performance of SHADE on CEC 2013 benchmark problems. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1952–1959, 2013.

[17] R. Tanabe and A. Fukunaga. Success-history based parameter adaptation for differential evolution. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 71–78, 2013.

[18] J. Tvrdík. Competitive differential evolution. *Proceedings of the International Conference on Soft Computing MENDEL*, pages 7–12, 2006.

[19] J. Tvrdík. Exponential crossover in competitive differential evolution. *Proceedings of the International Conference on Soft Computing MENDEL*, pages 44–49, 2008.

[20] J. Tvrdík. Adaptation in differential evolution: A numerical comparison. *Applied Soft Computing*, 9(3):1149–1155, 2009.

[21] Y. Wang, Z. Cai and Q. Zhang. Differential Evolution with Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Transactions on Evolutionary Computation*, 15:55–66, 2011.

[22] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82, 1997.

[23] Z. Yang, K. Tang and X. Yao. Self-adaptive differential evolution with neighborhood search. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1110–1116, 2008.

[24] D. Zaharie. Influence of crossover on the behavior of differential evolution algorithms. *Applied Soft Computing*, 9:1126–1138, 2009.

[25] J. Zhang and A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13:945–958, 2009.