MULTI-POPULATION ADAPTIVE INFLATIONARY DIFFERENTIAL EVOLUTION

Marilena Di Carlo
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, United Kingdom
marilena.di-carlo@strath.ac.uk

Massimiliano Vasile
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, United Kingdom
massimilano.vasile@strath.ac.uk

Edmondo Minisci
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, United Kingdom
edmondo.minisci@strath.ac.uk

Abstract In this paper, a multi-population version of Adaptive Inflationary Differential Evolution, which automatically adapts the crossover probability and the differential weight of the Differential Evolution, is presented. The multi-population algorithm exploits the use of different populations, and the local minima found by each population, to assess the distance between minima; a probabilistic kernel based approach is then used to automatically adapt the dimension of a bubble in which the population is re-initialized after converging to a local minimum. The algorithm is tested on two real case functions and on two difficult academic functions.

Keywords: Adaptive algorithms, Differential evolution, Global optimization.

1. Introduction

Differential Evolution (DE), [13], is a population-based stochastic algorithm for solving optimization problems. Although it has proved to be a very efficient global optimizer, work has been done to enhance its performance by combining it with deterministic or stochastic optimizers [4, 5, 15]. In [19], Inflationary Differential Evolution Algorithm (IDEA) was introduced. IDEA is based on the hybridization of Differential Evolution (DE) with the restarting procedure of Monotonic Basin Hopping (MBH) algorithm [20]. The performance of IDEA was found to be dependent upon the parameters controlling both the DE and MBH heuristics [19]. In particular, the DE performance is strongly influenced by the crossover probability, CR, and the differential weight, F, whose best settings are heavily problem dependent [8].

The need to have an algorithm capable of self-adapting these two parameters has resulted in many works [1, 3, 10, 12, 14]. The next step in the development of IDEA has therefore been the adaptation of CR and F, leading to Adaptive Inflationary Differential Evolution Algorithm (AIDEA) [11]. This algorithm uses a probabilistic kernel based approach to automatically adapt the values of both CR and F.

Starting from the successful results of AIDEA, this paper introduces a multi-population version of AIDEA (MP-AIDEA), using different strategies to create the mutant vector of the DE, different strategies to adapt CR and F and a new mechanism to adapt the dimension of the search space in which the population is re-initialized. Other multi-populations DE algorithms have been presented in [16, 21, 22].

In the first part of this paper MP-AIDEA is described. Then the results of four test cases are presented.

2. Multi-Population Adaptive Inflationary Differential Evolution Algorithm

The algorithm presented in this paper is a further development of AIDEA [11]. In the following, a summary of AIDEA and a detailed description of MP-AIDEA are given.

AIDEA. The first step in the run of AIDEA is a DE process in which each element of the population is associated to a different value of CR and F. During the advancement of the population from parents to children the values of CR and F are adapted using a kernel based approach. The DE is run until the population contracts below a threshold, identified by the parameter ρ. When this contraction condition is satisfied a local search is performed from the best individual in the population.

The found local minimum is archived in a matrix of minima and the population is restarted in a bubble of dimension δ_{local} around the local minimum (local restart). Local restart is iterated up to a predefined maximum value, identified by the value iun. When this value is reached the population is restarted at a distance δ_{global} from the cluster of local minima found thus far (global restart). The algorithm stops when the maximum number of function evaluation is reached.

MP-AIDEA. In MP-AIDEA the single population of AIDEA is replaced by many populations. The common archive of local minima of all the populations can be used to create the mutant vector of the DE. Three strategies have been considered for the generation of the mutant vector: 1) *DE/best/1-DE/rand/1*: the mutant vector is created randomly using the best element or a random element of the population; 2) *DE/arch/1-DE/rand/1*: the mutant vector is created randomly using an element from the archive of local minima or a random element of the population: 3) *DE/arch/1-DE/best/1*: the mutant vector is created randomly using an element from the archive of local minima or the best element of the population.

As regards the adaptation, the presence of many populations can be exploited to adapt CR and F in a different way with respect to AIDEA. Two strategies for the adaptation of CR and F are proposed: 1) MP-AIDEA-CRF1 (CR and F adaptivity realized using CR and F values equal for every element of each population and comparing the populations to each other) and 2) MP-AIDEA-CRF2 (CR and F adaptivity realized using different CR and F values for each element of each population and comparing elements of each single population to each other, as in AIDEA [11]).

Finally, a strategy is proposed to adapt also the dimension of the bubble for the local restart of the population, using a kernel based approach similar to the one used for the adaptation of CR and F. Considering all these possibilities, twelve different versions of the algorithm have been developed and tested:

- MP-AIDEA 1: MP-AIDEA-CRF1-*DE/best/1-DE/rand/1*
- MP-AIDEA 2: MP-AIDEA-CRF1-*DE/arch/1-DE/rand/1*
- MP-AIDEA 3: MP-AIDEA-CRF1-*DE/arch/1-DE/best/1*
- MP-AIDEA 4 to 6: MP-AIDEA 1 to 3 with δ_{local} adaptation
- MP-AIDEA 7 to 12: MP-AIDEA 1 to 6 based on CRF2 strategy

A detailed description of the common structure of the algorithms is given in Algorithm 1. The procedure starts by setting values for n_{pop}

(number of elements in each population), N_{pop} (number of populations), iun (maximum number of local restart), $\bar{\rho}$ (size of the convergence box), δ_{global} (distance from the cluster centres for the global restart) and δ_{local} (dimension of the bubble for the local restart, if not adapted) as in line 1 and initializing the populations (line 3). The joint PDF for CR and F is then initialised to be a uniform distribution (lines 4 and 5). For MP-AIDEA-CRF1, DE is run (line 11) drawing probabilistically a value for F and CR from **CRF** for each population (line 9) and **CRF** is updated on the basis of the improvement of the populations (step 15). For MP-AIDEA-CRF2, lines 9, 15 and 16 are to be considered inside the **for** cycle over the elements of the population (different values of CR and F for each element of the populations). If the populations contracts below a predefined threshold (step 18), a local optimizer is run from the current minimum (line 19) and the found local minimum is saved in an archive of local minima of all the populations (line 32). iun_m is updated based on the improvement of the value of $f_{min,m}$ (lines 23 to 28). If the adaptation of δ_{local} is performed, when all the population have performed the local search, a matrix **B** for the adaptation of the dimension of the bubble can be created (step 34, Algorithm 3) using the local minima found thus far. At this point the populations go through local or global restart according to lines 39 to 45. In particular, if the local optimizer failed to improve the value of f_{min} more than iun_{max} times, the population is restarted globally and iun is set to 0, otherwise the population is restarted within a local bubble and $iun = iun+1$. The dimension of the bubble for the local restart is sampled from matrix **B** (line 40) or is the one defined at line 1 if δ_{local} is not adapted. The adaptation of **B** (line 36) is done only when the local optimizer has been applied to all the population for the second time (for each population, the adaptation can be performed only if two local minima are known for that population). At this point, the loop restart from the initialization of **CRF**. As a terminal criterion the algorithms stops if the maximum number of function evalutation $n_{feval,max}$ has been reached.

CR and F adaptation. The updating procedure for **CRF** is detailed in Algorithm 2 for MP-AIDEA-CRF1. For each population, the maximum objective function difference between parents and children, dd_{max}, is computed (line 1). Then the element of **CRF** are sequentially evaluated and the first time that the dd value associated to the considered row is lower than dd_{max} (line 4) the value of F used for the considered population substitutes the corresponding elements $\mathbf{CRF}_{j,2}$ (line 5). For CR, the value associated to the considered population sub-

stitutes $\mathbf{CRF}_{j,1}$ (line 7) only if dd_{max} is greater than a given value CRC (line 6), [11].

For MP-AIDEA-CRF2 the **for** cycle in line 2 is substituted by a **for** cycle over the elements of the single population and dd_{max} is replaced by $f(\mathbf{x}_{i,k}) - f(\mathbf{x}_{i,k+2})$ for each element i of the population, as in [11].

δ adaptation. The generation of the matrix $\mathbf{B}$ for the adaptation of the dimension of the bubble for the local restart is described in Algorithm 3; it reflects the procedure for the creation of **CRF** and is based on the computation of the distance between the local minima found by the different populations and stored in the common archive of local minima. The updating procedure for $\mathbf{B}$, detailed in Algorithm 4, follows the same approach used for the adaptation of **CRF**; the distance between local minimum found at subsequent local restart is used to assess the validity of the used dimension of the bubble for the local restart (a local restart is effective if the algorithm move from a local minimum to another).

The process of adaptation of the dimension of the bubble for the local restart will be presented in greater details in the first two test cases of the Test Results section.

3. Test Results

The test cases are taken from the technical report of the CEC 2005 and CEC 2011 competitions [6,17]. The considered problems are: Spread Spectrum Radar Polyphase Code Design and Tersoff Radar Function Minimization Problem from CEC 2011; Schwefel's Problem, Function 12, and Rotated Version of Hybrid Composition Function, Function 16, from CEC 2005. The statistics reported are computed on the results obtained from 100 independent runs in which new populations are generated at each run. The success rate reported in the next tables and figures is computed as number of times (over the 100 runs) in which the minimum found by the algorithm is lower than $f_{min}+\epsilon$ where f_{min} is the minimum value of the function and ϵ is a given threshold [18]. $\epsilon = 0.001$ for the test cases from CEC 2011 and $\epsilon = 0.01$ for the CEC 2005 problems [17].

3.1 Spread Spectrum Radar Polyphase Code Design

This problem has dimension $n_D = 20$ and the best solution found is $f_{min} = 0.5$. The maximum number of function evaluation is 1.5e5 [6]. The same parameters setting of [11] was used, that is $\delta_{local} = \delta_{global} = 0.1$, $\rho = 0.2$ and $iun = 10$. The results obtained using AIDEA in [11] (where the DE strategy was *DE/best/1*) and new results obtained using AIDEA with the DE strategy *DE/best/1-DE/rand/1* are reported in

Algorithm 1 Multi-Population Adaptive Inflationary Differental Evolution Algorithm

Set values for n_{pop}, N_{pop}, iun, $\bar{\rho}$, δ_{global}
Set $n_{feval,m} = 0$ and $k_m = 1$ for all populations $m \in [1, \ldots, N_{pop}]$
Initialize population $\mathbf{x}_{m,i,k}$ for all $m \in [1, \ldots, N_{pop}]$ and for all $i \in [1, \ldots, n_{pop}]$
A regular mesh with $(n_D + 1)^2$ points (where n_D is the dimensionality of the problem) in the space $CR \in [0.1, 0.99] xF \in [-0.5, 1]$ is created
Initialize **CRF** with points of the mesh: $\mathbf{CRF}_{j,1} \leftarrow \mathbf{CR}_j$ and $\mathbf{CRF}_{j,2} \leftarrow \mathbf{F}_j$ for all $j \in [1, \ldots, (n_D + 1)^2]$
Associate to each row of **CRF** an element $dd_j = 0$ for all $j \in [1, \ldots, (n_D + 1)^2]$
Row sort **CRF** in terms of dd values
for $m \in [1, \ldots, N_{pop}]$ **do**
    Sample $\mathbf{CR}_{m,k}$ and $\mathbf{F}_{m,k}$ from **CRF**
    for $i \in [1, \ldots, n_{pop}]$ **do**
        $\mathbf{x}_{m,i,k+1} \leftarrow \mathrm{DE}(\mathbf{x}_{m,i,k}, \mathbf{CR}_{m,k}, \mathbf{F}_{m,k})$
        $n_{feval,m} = n_{feval,m} + 1$
    end for
    $k_m = k_m + 1$
    Update **CRF** (see Algorithm 2)
    Row sort **CRF** in terms of dd values
    $\rho_m = \max\left(||\mathbf{x}_{m,i,k} - \mathbf{x}_{m,j,k}||\right) \quad \forall \mathbf{x}_{m,i,k}, \mathbf{x}_{m,j,k} \in P_{m,k}$
    if $\rho_m < \bar{\rho} \cdot \rho_{max,m}$ **then**
        Run a local optimizer from $\mathbf{x}_{best,m}$ and let $\mathbf{x}_{l,m}$ be the local minimum found by the local optimizer
        if $f(\mathbf{x}_{l,m}) < f(\mathbf{x}_{best,m})$ **then**
            $f(\mathbf{x}_{best,m}) \leftarrow f(\mathbf{x}_{l,m})$
        end if
        if $f(\mathbf{x}_{best,m}) < f_{min,m}$ **then**
            $f_{min,m} \leftarrow f(\mathbf{x}_{best,m})$
            $iun_m = 0$
        else
            $iun_m = iun_m + 1$
        end if
    else
        Termination Unless $n_{feval,m} \geq n_{feval,max}$ goto (10)
    end if
    Add $\mathbf{x}_{best,m}$ to the archive of minima of population: $A_{g,m} = A_{g,m} + \{\mathbf{x}_{best,m}\}$
end for
Create matrix **B** for adaptation of the dimension of the bubble (see Algorithm 3)
if (All population went for the 2nd time through the local minimizer) **then**
    Update **B** (see Algorithm 4)
end if
for $m \in [1, \ldots, N_{pop}]$ **do**
    if $iun \leq iun_{max}$ **then**
        Sample $\delta_{local,m}$ from **B** to define the bubble D_m
        Initialize population $\mathbf{x}_{m,i,k}$ for all $i \in [1, \ldots, n_{pop}]$ in the bubble D_m
    else
        Define clusters in the archive and compute baricentre $\mathbf{x}_{c,m}$ of each cluster
        Initialize population $\mathbf{x}_{m,i,k}$ for all $i \in [1, \ldots, n_{pop}]$ such that $\forall i, j ||\mathbf{x}_{m,i,k} - \mathbf{x}_{m,j,k}|| > \delta_{global}$
    end if
end for
Termination Unless $n_{feval,m} \geq n_{feval,max}$ goto (4)

Table 1, along with the results of two of the best performing algorithms of the CEC 2011 competition, the Genetic Algorithm with Multi Parent Crossover (GA-MPC) [7] and the Weed Inspired Differential Evolution (WI-DE) [9].

AIDEA gives better results than GA-MPC or WI-DE. In Figure 1 the success rates obtained using different population number N_{pop} composed

Algorithm 2 Updating procedure for **CRF**

For each population compute $dd_{max,m} = max||f(\mathbf{x}_{m,i,k+1}) - f(\mathbf{x}_{m,i,k})||$ for all $i \in [1, \ldots, n_{pop}]$
for $m \in [1, \ldots, N_{pop}]$ **do**
  for $j \in [1, \ldots, (n_D + 1)^2]$ **do**
    if $dd_j < dd_{max,m}$ **then**
      $\mathbf{CRF}_{j,2,k} \leftarrow \mathbf{F}_{m,k}$
      if $dd_{max,m} > CRC$ **then**
        $\mathbf{CRF}_{j,1,k} \leftarrow \mathbf{CR}_{m,k}$
      end if
    end if
  end for
end for

Algorithm 3 Generation of matrix **B** for the adaptation of the bubble

Compute mean and minimum distance between all local minima in A_{gm} for all $m \in [1, \ldots, N_{pop}]$: $\mathbf{d}_{minMIN}$ and $\mathbf{d}_{minMEAN}$
Create regular mesh with $(n_D + 1)^2$ points in the space $[\mathbf{d}_{minMIN}, \mathbf{d}_{minMEAN}]$
Initialize **B** with points of the mesh
Associate to each row of **B** an element $dd_{bj} = 0$ for all $j \in [1, \ldots, (n_D + 1)^2]$
Row sort **B** in terms of dd_b values

Algorithm 4 Updating procedure for **B**

For each population compute $p_m = ||\mathbf{x}_{l,m,k+1} - \mathbf{x}_{l,m,k}||$
for $m \in [1, \ldots, N_{pop}]$ **do**
  for $j \in [1, \ldots, (n_D + 1)^2]$ **do**
    if $dd_{bj} < p_m$ **then**
      $\mathbf{B}_{j,1,k} \leftarrow \delta_{local,m}$
    end if
  end for
end for

by $n_{pop} = 10$ elements are shown for MP-AIDEA 1, 4, 7 and 10 and MP-AIDEA 2, 5, 8 and 11. Results from MP-AIDEA 3, 6, 9, 12 are very similar to MP-AIDEA 2, 5, 8, 11 and therefore are not shown. AIDEA was tested with a number of individuals in the single population equal to the total number of individuals of MP-AIDEA. The most successful versions of MP-AIDEA are 1, 7 and 10; their results are always better than AIDEA's one. MP-AIDEA versions 2, 5, 8 and 11 show a success

Table 1. Spread Spectrum Radar Polyphase Code Design – AIDEA, GA-MPC and WI-DE results.

Algorithm	n_{pop}	Min	Mean	Max	Str.Dev.	S.Rate
AIDEA *DE/best*	20	0.5000	0.5150	0.6509	0.0343	-
AIDEA DE/*best*-DE/*rand*	20	0.5000	0.5130	0.6422	0.0263	75
GA-MPC	-	0.5000	0.7484	0.9334	0.1249	-
WI-DE	-	0.5000	0.656	0.993	0.116	-

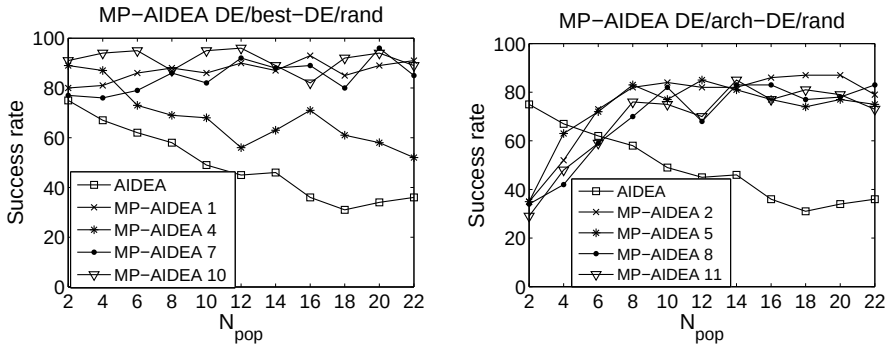

Figure 1. Spread Spectrum Radar Polyphase Code Design – MP-AIDEA success rate.

rate increasing with N_{pop} and greater than the success rate of AIDEA for N_{pop} sufficiently high.

In Figure 2 the process of adaptation of the dimension of the bubble for the local restart is shown for MP-AIDEA 4 and $N_{pop} = 3$ for a sequence of 19 subsequent local restarts before the global restart of the algorithm. The bold line represents the mean value of δ_{local} over all the populations. It is evident that $\delta_{local} = 0.1$ proves to be a good guess for the value of δ_{local}.

3.2 Tersoff Potential Function Minimization Problem

This problem has dimension $n_D = 30$ and the best solution is $f_{min} = -36.9286$. The maximum number of function evaluation is 1.5e5. AIDEA and MP-AIDEA were tested using two different sets of parameters settings: $\delta_{local} = \delta_{global} = 0.1$, $\rho = 0.2$, $iun = 10$ (Case 1) and $\delta_{local} = 0.3$, $\delta_{global} = 0.1$, $\rho = 0.2$, $iun = 10$ (Case 2). The results obtained using AIDEA in [11] and new results obtained using AIDEA with the DE strategy DE/*rand/1*-DE/*best/1* are reported in Table 2, along with the results obtained by GA-MPC and WI-DE.

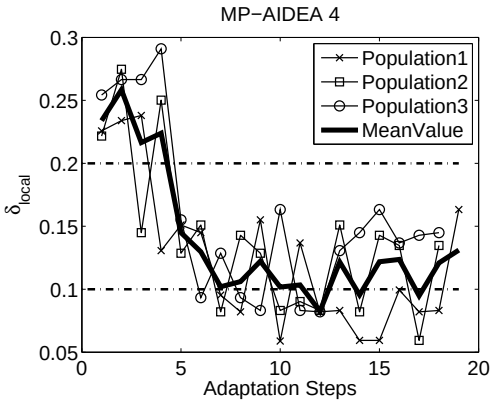

Figure 2. Spread Spectrum Radar Polyphase Code Design – adaptation of δ_{local}.

Table 2. Tersoff Potential Function Minimization Problem – AIDEA, GA-MPC and WI-DE results.

Algorithm	n_{pop}	Min	Mean	Max	Str.Dev.	S.Rate
Case 1						
AIDEA DE/*best*	20	-36.9286	-36.8527	-35.5171	0.2442	-
AIDEA DE/*best-rand*	20	-36.9286	-36.8046	-35.9700	0.2483	34
Case 2						
AIDEA DE/*best-rand*	20	-36.9286	-36.6219	-35.4467	0.4694	11
GA-MPC	-	-36.8457	-35.03883	-34.1076	0.8329	-
WI-DE	-	-36.8	-35.6	-34.2	0.904	-

In Figure 3 the results obtained from different combinations of $N_{pop} \times n_{pop}$ are shown for the best variants of MP-AIDEA and for both Case 1 and Case 2.

For Case 1 the best results are given by MP-AIDEA 1 and MP-AIDEA 7. Changing the values of δ_{local} from 0.1 to 0.3 (Case 2) results however in a successful performance of the algorithms with adaptation of δ_{local}, that is MP-AIDEA 4 and MP-AIDEA 10. This is due to the fact that for Case 1 the chosen value of δ_{local} was close to the optimal value for this problem. This is proved in Figure 4, where the process of adaptation of δ_{local} is shown for MP-AIDEA 4 using 3 populations. Arbitrary chosen values, such as $\delta_{local} = 0.3$ (Case 2) are very dissimilar from the one obtained through the adaptation process ($\delta_{local} = 0.1$).

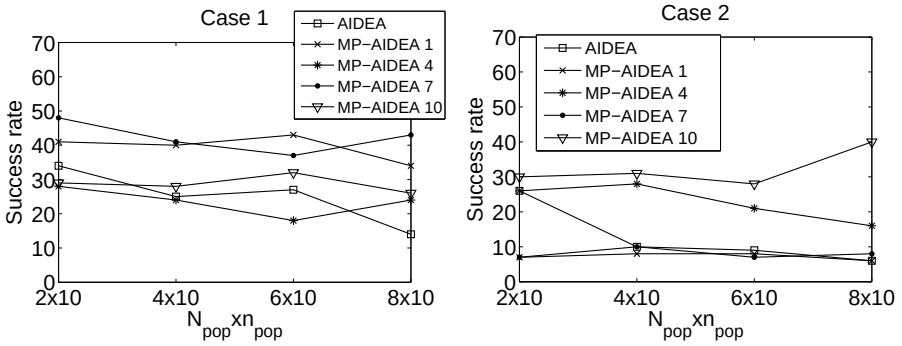

Figure 3. Tersoff Potential Function Minimization Problem – MP-AIDEA success rate.

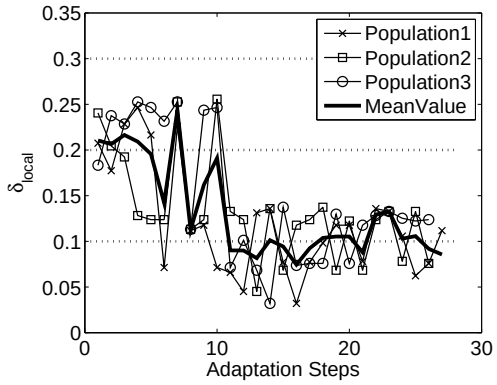

Figure 4. Tersoff Potential Function Minimization Problem – adaptation of δ_{local}.

3.3 Schwefel's Problem

Schwefel's problem was tested with dimension $n_D = 30$ and $n_D = 50$. The best solution is $f_{min} = -460$; the parameters settings is $\delta_{local} = \delta_{global} = 0.1$, $\rho = 0.2$ and $iun = 5$. The maximum number of function evalutions is 3e5 for the 30D problem and 5e5 for the 50D problem, [17]. The results obtained using AIDEA with DE strategy DE/*rand/1*-DE/*best/1* are reported in Table 3 as statistics of the objective function error values with respect to f_{min}, as required by [17], along with the results obtained by one of the best performing algorithms of the CEC 2005 competition, the Restart Covariance Matrix Adaptation Evolution Strategy with Increasing Population Size (IPOP-CMA-ES) [2]. AIDEA gives better results for the 30D problem and for the 50D problem it locates the global minimum, while CMA-ES was not able to find it.

Table 3. Schwefel's Problem – AIDEA and IPOP-CMA-ES results.

Algorithm	n_D	n_{pop}	Min	Mean	Max	Str.Dev.	S.Rate
AIDEA	30	20	2.01e-9	1.03e+2	1.00e+3	1.97e+2	43
IPOP-CMA-ES	30	-	3.79e-9	4.43e+4	1.10e+6	2.19e+5	-
AIDEA	50	40	1.45e-8	2.63e+3	1.75e+4	2.74e+3	1
IPOP-CMA-ES	50	-	9.67e+0	2.27e+5	5.57e+6	1.11e+6	-

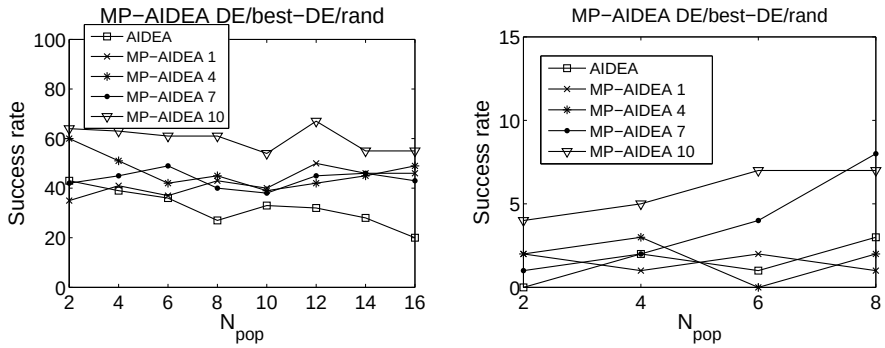

Figure 5. Schwefel's Problem – MP-AIDEA success rate.

In Figure 5 the success rates obtained for different values of N_{pop}, with $n_{pop} = 10$ for the 30D problem and $n_{pop} = 20$ for the 50D problem, are shown for the most successful versions of MP-AIDEA. For the 30D problem MP-AIDEA gives better results than AIDEA in most of the cases; for the 50D problem MP-AIDEA is able to find the global minimum of the function.

3.4 Rotated Version of Hybrid Composition Function

For this function the best solution is $f_{min} = 120$, $n_D = 10$, $\delta_{local} = \delta_{global} = 0.1$, $\rho = 0.2$ and $iun = 5$. The maximum number of function evaluations is 1e5 [17]. The results obtained using AIDEA, IPOP-CMA-ES and MP-AIDEA are shown in Table 4 and Table 5, where results from different combinations of $N_{pop} \times n_{pop}$ are presented.

The most successful variants of MP-AIDEA were able to locate the global minimum of this function, a minimum that neither IPOP-CMA-ES nor AIDEA were able to find.

Table 4. Rotated Version of Hybrid Composition Function – AIDEA and IPOP-CMA-ES results.

Algorithm	n_{pop}	Min	Mean	Max	Str.Dev.	S.Rate
AIDEA	40	5.38e+1	1.02e+2	1.14e+2	8.42e+0	0
IPOP-CMA-ES	-	7.92e+1	9.13e+1	9.68e+1	3.49e+0	-

Table 5. Rotated Version of Hybrid Composition Function – MP-AIDEA success rate.

Algorithm	2x20	4x20	6x20	8x20
MP-AIDEA 4	**2**	**1**	**1**	0
MP-AIDEA 10	**1**	0	**1**	**3**

4. Conclusions

In this paper a multi-population version of AIDEA have been presented and tested. Results have shown that MP-AIDEA can give results which are better, or at least comparable, to the ones provided by AIDEA. The new strategies DE/*arch*/*1*-DE/*rand*/*1* and DE/*arch*/*1*-DE/*best*/*1* have shown to be effective when the number of populations is not too low. The adaptation of the bubble dimension has proven to give good results, having moreover the advantage of not requiring the setting of the parameter δ_{local} for the dimension of the bubble of the local restart. In addition, the most successful versions of MP-AIDEA were able to locate for the first time the global minima of two difficult academic functions.

References

[1] M. M. Ali and A. Torn. Population set based global optimization algorithms: Some modifications and numerical studies. *Comput. Oper. Res.*, 31(10):1703–1725, 2004.

[2] A. Auger and N. Hansen. A Restart CMA Evolution Strategy with Increasing Population Size. *Proc. IEEE Congress on Evolutionary Computation (CEC)*, pp. 1769–1776, 2005.

[3] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problem. *IEEE T. Evolut. Comput.*, 10:646–657, 2006.

[4] J. P. Chiou and F. S. Wang. A Hybrid Method of Differential Evolution with Application to Optimal Control Problem of a Bioprocess System. In *Proc. IEEE World Congress on Computational Intelligence (WCCI)*, pages 627–632, 1998.

[5] L. Coelho and V. C. Mariani. A Hybrid Method of Differential Evolution and SQP for Solving the Economic Dispatch Problem with Valve-Point Effect. *Applications of Soft Computing*, pages 311–320, 2006.

[6] 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, 2010.

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

[8] R. Gamperle, S.D. Muller, and P. Koumoutsakos. A Parameter Study for Differential Evolution. In *Proc. WSEAS International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, pages 293–298, 2002.

[9] U. Halder, S. Das, D. Maity, A. Abraham, and P. Dasgupta. Self Adaptive Cluster Based and Weed Inspired Differential Evolution Algorithm for Real World Optimization. *Proc. IEEE Congress on Evolutionary Computation (CEC)*, pp. 750–756, 2011.

[10] J. Liu and J. Lampinen. A fuzzy adaptive differential evolution algorithm. *J. Soft Computing*, 9:448–462, 2005.

[11] E. Minisci and M. Vasile. Adaptive Inflationary Differential Evolution. In *Proc. IEEE World Congress on Computational Intelligence (WCCI)*, 2014.

[12] M. G. H. Omran, A. Salman, and A. P. Engelbrecht. Self-adaptive Differential Evolution. *Lect. Notes Artif. Intell.*, 3801:192–199, 2005.

[13] K. V. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution. A Practical Approach to Global Optimization. Natural Computing Series*, Springer, 2005.

[14] A. K. Qin, V. L. Huang, and P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE T. Evolut. Comput.*, 13:398–417, 2009.

[15] A. Qinq. *Differential Evolution: Fundamentals and Applications in Electrical Engineering*. John Wiley & Sons, 2009.

[16] D. Shen, Y. Li, B. Wei, and X. Xia. Adaptive Forking Multipopulation Differential Evolution Algorithm for Multimodal Optimization. *J. Convergence Information Technology*, 7(5):218–227, 2011.

[17] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Technical Report, 2005.

[18] M. Vasile, E. Minisci, and M. Locatelli. Analysis of Some Global Optimization Algorithms for Space Trajectory Design. *J. Spacecraft Rockets*, 47:334–344, 2010.

[19] M. Vasile, E. Minisci, and M. Locatelli. An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization. *IEEE T. Evolut. Comput.*, 15:267–281, 2011.

[20] D.J. Wales, and J.P.K. Doye. Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms. *J. Phys. Chem.*, 10:5111–5116, 1997.

[21] W. Yu and J. Zhang. Multi-population Differential Evolution with Adaptive Parameter Control for Global Optimization. In *Proc. 13th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, pages 1093–1098, 2011.

[22] D. Zaharie. A Multipopulation Differential Evolution Algorithm for Multimodal Optimization. In *Proc. 10th International Conference on Soft Computing (MENDEL)*, pages17–22, 2004.