World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:16, No:8, 2022

A Multi-Population Differential Evolution with Adaptive Mutation and Local Search for Global Optimization

Zhoucheng Bao, Haiyan Zhu, Tingting Pang, Zuling Wang

Abstract—This paper presents a multi population Differential Evolution (DE) with adaptive mutation and local search for global optimization, named AMMADE in order to better coordinate the cooperation between the populations and the rational use of resources. In AMMADE, the population is divided based on the Euclidean distance sorting method at each generation to appropriately coordinate the cooperation between subpopulations and the usage of resources, such that the best-performed subpopulation will get more computing resources in the next generation. Further, an adaptive local search strategy is employed on the best-performed subpopulation to achieve a balanced search. The proposed algorithm has been tested by solving optimization problems taken from CEC2014 benchmark problems. Experimental results show that our algorithm can achieve a competitive or better result than related methods. The results also confirm the significance of devised strategies in the proposed algorithm.

Keywords—Differential evolution, multi-mutation strategies, memetic algorithm, adaptive local search.

I. Introduction

DE is a population-based stochastic search technique. It uses mutation, crossover, and selection operators at each generation to move its population toward the global optimum [1]. Due to its simplicity and efficiency, DE has been successfully applied to many fields. Generally, the performance of DE relies on recombination strategies, parameters as well as the population structure [2]. Typically, different problems require different mutation strategies and parameter settings [3]. To appropriately use mutation strategies, a common approach is to adaptively control the mutation strategies and parameters [4].

Many DE variants with adaptive mutation strategy and/or parameter control have been proposed in literature. For example, JADE [5] employed a parameter adaptation strategy along with DE/current-to-pbest mutation strategy to generate new solutions. In EPSDE [4], a pool of distinct mutation and crossover strategies along with a pool of values for each control parameter coexists throughout the evolution process and competes to produce offspring. CoDE [6] is a DE variant with composite trial vector generation strategies and control parameters, in which three trial vector generation strategies namely DE/rand/1, DE/rand/2 and DE/current-to-rand/1 are employed. SaDE [7] used trial vector generation strategy and parameter adaptation strategy to match different phases of search process. The mutation strategies in the above algorithms are employed on a single population and each mutation strategy is assigned with the same computational resources, which may lead to a waste of resources. Individual mutation strategies at each stage may have a different performance and a good strategy should be assigned with more computing resources [8].

In recent years, the integration of multiple mutation strategies to multiple subpopulations has also attracted attention. For example, MPEDE [9], which is a multi-population based DE, realized a dynamic ensemble of multiple mutation strategies. In MPEDE, the authors tried to divide the population into a larger reward sub-population and three smaller equal indicator subpopulations. At the beginning of evolutionary process, three indicator subpopulations are assigned to three mutation strategies, and the reward subpopulation is randomly assigned to one of the three mutation strategies. After a certain number of generations, the reward subpopulation is assigned to the best mutation strategy [8]. Three mutation strategies, including DE/current-to-pbest/1 with an archive, DE/current-to-rand/1, and DE/rand/1 are employed in MPEDE. DE/current-to-pbest/1 with an archive is very competitive in solving complex optimization problems, especially those with uni-modal and multimodal landscapes, DE/current-to-rand/1 without crossover operation is a rotation-invariant and useful in solving rotated problem, while the DE/rand/1 strategy has a strong exploration capability and can effectively maintain the diversity of populations [5], [8], [9]. The control parameters of MPEDE are based on the scheme proposed in JADE. However, in MPEDE, most computing resources are allocated to the best strategy, while the mutation strategy employed by the individual is random, which may not be effective.

To improve the efficiency of DE, memetic DE [10] has been proposed. For example, Rogalsky and Derksen [11] combined downhill simplex (DS) with DE in order to accelerate convergence. Molina et al. [12] proposed a memetic DE called SHADE-ILS for large-scale global optimization, in which SHADE [13] and ILS [14] are combined. These methods are generally based on single local search on a single population, or use a single local search in multi population, which may have difficulty to balance exploration and exploitation.

In this paper, two strategies have been proposed based on

Zhoucheng Bao is with the Department of Computer Science, Hangzhou Normal University, China (corresponding author, e-mail: baozc1031@163.com).

Haiyan Zhu is with the Department of Business, Jiaxing University, China (e-mail: zhuhaiyan1022@163.com).

Tingting Pang and Zuling Wang are with the Department of Computer Science, Hangzhou Normal University, China (e-mail: 2020112011025@stu.hznu.edu.cn, wangzuling@stu.hznu.edu.cn).

MPEDE to address global optimization problems. Different from the grouping of MPEDE, the reward population is removed, and the entire population is divided into three subpopulations of the same size. According to the performance of the three subpopulations at each generation, a main subpopulation is obtained. The main population will be assigned with a more computing resources at each generation. At the same time, a sorting method based on Euclidean distance is adopted to partition the population. At each generation, individuals will adopt a suitable mutation strategy to ensure a reasonable utilization of resources. Finally, in order to balance exploration and exploitation, adaptive local search is proposed and applied on the main subpopulation. Taking into account the limitation of computing resources, when the diversity of the population is small, the Cauchy local search is used for the outstanding individuals in the main subpopulation, otherwise the Gaussian local search is used to improve the best individual of the main subpopulation. Experiments show that the performance of AMMADE is competitive or better than JADE [5], SaDE [7], EPSDE [4], CoDE [6] and MPEDE [9].

The rest of this paper is arranged as follows. In Section II, the related work is briefly reviewed. AMMADE algorithm is described in detail in Section III. Section IV gives the comparison results with other algorithms. Finally, Section V concludes this paper.

II. Related Work

The DE algorithm initializes the population of NP candidate solutions randomly, X_i={$x_i^1, \ldots, x_i^D$} $i = 1, ..., NP$ and explores the search space by sampling on the D-dimensional space. After population initialization, the DE algorithm will undergo a generational evolution, which consists of three operations: mutation, crossover and selection.

The mutation strategy has a great impact on the performance of the algorithm, and the DE algorithm generates a new solution through the corresponding mutation strategy in each generation to find the best fitness value. The commonly used mutation strategies in the past few years are as follows:

1) DE/best/1

$$V_i = X_{best} + F_i \cdot (X_{r2} - X_{r1}) \tag{1}$$

2) DE/rand/1

$$V_i = X_{r3} + F_i \cdot (X_{r2} - X_{r1}) \tag{2}$$

3) DE/current-to-pbest/1

$$V_i = X_i + F_i \cdot (X^p{}_{best} - X_i) + F_i \cdot (X_{r2} - X_{r1}) \tag{3}$$

4) DE/current-to-rand/1

$$V_i = X_i + F_i \cdot (X_{r3} - X_i) + F_i \cdot (X_{r2} - X_{r1}) \tag{4}$$

where X_{best} denotes the best parent vector in the current population, $X^p{}_{best}$ is randomly chosen from the top $100 * p\%$ individuals in the current population, F_i commonly known as the scaling factor to control the rate of evolution of the population. The indices r_1,r_2,r_3 are randomly generated anew for each mutant vector and are mutually exclusive $(r_1 \neq r_2 \neq r_3)$.

After the mutation, DE then undergoes a crossover operation to generate a trial vector U_i. Commonly used crossover strategies are binomial crossover and exponential crossover. The binomial crossover is defined as follows.

$$U_i{}^j = \begin{cases} V_i{}^j, & if\ rand[0,1) \le C_r\ or\ j = j_{rand}, \\ X_i{}^j, & otherwise. \end{cases} \tag{5}$$

where $C_r \in (0, 1]$ is crossover probability, which controls the number of decision variable values. j_{rand} is a random number to ensure that at least one number is always selected from the mutant vector V_i.

After the mutation and crossover operation, election operation will be employed. The selection operation is a process of elimination and screening of old and new individuals (V_i and U_i) based on fitness values. The selection operation is defined as:

$$X_i = \begin{cases} U_i, & if\ f(U_i) \le f(X_i) \\ X_i, & otherwise. \end{cases} \tag{6}$$

where $f(U_i)$ and $f(X_i)$ are the fitness values of U_i and X_i, respectively.

III. Proposed Algorithm

In this section, we present the detailed process and strategy of our algorithm. First is initializing a population randomly, calculating the fitness value of each individual and the Euclidean distance to the best individual. Unlike MPEDE [9], we changed the way of randomly allocating resources, sorted according to the Euclidean distance to the best individual, and then divided the population into three subpopulations of the same size. Then is the employing different mutation strategies for the three subpopulations, that is, using DE/rand/1 for the top ranking subpopulation, DE/current-to-rand/1 for the medium ranking subpopulation, and DE/current-to-pbest/1 for the worst ranking population. The three subpopulations share the best individual information according to the fitness value. The three sub-populations will run in parallel, and the most potential subpopulation P_{main} is obtained according to the indicators. An adaptive local search operation is performed on P_{main}, and the main population P_{main} will get more computing resources through the population migration strategy in the next generation. Finally, when the computing resource budget is finished, output is the final result. The diagram of MAs and the overview of the proposed algorithm are shown in Fig. 1 and Algorithm 1, respectively.

The left part of Fig. 1 represents three subpopulations with different mutation/crossover operators. The three subpopulations start with the same population size and then dynamically change according to the performance of each subpopulation. The middle part represents the main population, and the right part represents two different local search methods.

In the following subsections, we will describe the adaptive mutation operator strategy based on multi-populations in Section III *A*, and the adaptive local search strategy in Section III *B*.

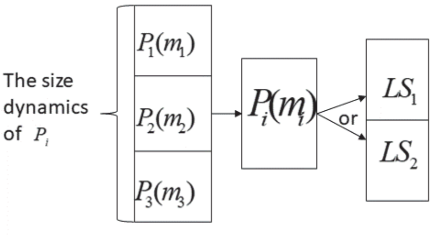

Fig. 1 Algorithm Structure Diagram

A. Adaptive Mutation Operator Strategy Based on Multi-Populations

The individual uses different mutation strategies at each stage according to the ranking of the Euclidean distance, and adopt a new resource allocation method, a better mutation strategy will get more computing resources. The specific operation is as follows.

Firstly, the entire population is divided into three equal-sized sub-populations according to the individual's Euclidean distance ranking in AMMADE. The top subpopulations have less diversity, and mutation strategy DE/rand/1 can be used to increase the diversity. The worst-ranked subpopulation has enough diversity, but the convergence speed is slow, and mutation strategy DE/current-to-pbest/1 will be used to improve the convergence speed. The top $1/3$ uses strategy DE/rand/1, the middle one uses strategy DE/current-to-rand/1, and the bottom $1/3$ one uses strategy of DE/current-to-pbest/1. During the run of three sub-populations, the most potential subpopulation is formed according to index SQF_p. We reorder the three subpopulations according to Euclidean distance, and migrate the top p_m individuals of the two populations with poor potential to the main subpopulation P_{main}. The index SQF_p is used to judge the potential of each subgroup and is defined as follows.

Considering the diversity of each sub population, the proportion of each sub population to the diversity of all subpopulations $DS_p \in [0, 1]$ is calculated according to (7).

$$DS_p = \frac{\frac{1}{NP_p}(\sum_{i=1}^{NP_p} dis(\overrightarrow{P_{\text{p},i}} - \overrightarrow{P_{p,best}}))}{\sum_{\text{p}}^{3} \frac{1}{NP_p}(\sum_{i=1}^{NP_p} dis(\overrightarrow{P_{\text{p},i}} - \overrightarrow{P_{p,best}}))}, \forall p = 1, 2, 3 \quad (7)$$

where $dis(\overrightarrow{P_{\text{p},i}} - \overrightarrow{P_{p,best}})$ is the Euclidean distance from each individual to the best individual of P_{main}. Considering the potential of each subpopulation from the best fitness value:

$$QF_p = \frac{fitnessP_{p,best}}{\sum_{p=1}^{3} fitnessP_{p,best}}, \forall p = 1, 2, 3 \quad (8)$$

Considering the quality and diversity of solutions, the larger the value of IFB_p, the better the potential to find the optimal value representing the sub population.

$$IFB_p = (1 - QF_p) + DS_p * (nfes/\max\_nfes), \forall p = 1, 2, 3 \quad (9)$$

Finally, considering the dynamic situation of optimization, the index SQF_p is finally based on the performance of the last three generations, which is shown in (10):

$$SQF_p = \begin{cases} IFB_{p,g}, \text{g} = 1 \\ \frac{1}{2} * \sum_{g=gen-1}^{gen} IFB_{p,g}, g \geq 2 \end{cases}, \forall p = 1, 2, 3 \quad (10)$$

where gen is the number of current running generation.

B. Adaptive Local Search Strategy

It has been well established that, in order to maintain the balance between exploration and exploitation, we should pay more attention to exploration at the early stage of evolution. While at the later stage of evolution, the algorithm should pay more attention to exploitation, thus accurately identifying the optimum [15]. We perform a local search operation on the top 2% individuals in the most potential subpopulation, when the index of population diversity LST is 1, the Cauchy local search method is used to enhance the population diversity, otherwise, the Gaussian local search is performed to further optimize the solution. The Gaussian local search and Cauchy local search are shown in (11) and (12), respectively.

$$x_{new} = Normrnd(x_{old}, e^{-(nfes/\max\_nfes)^2}) \quad (11)$$

As the iterative process $e^{-(nfes/\max\_nfes)^2}$ becomes smaller, the variation range of x_{new} changes is reduced, and the exploitation is more focused.

$$X_{i,new} = Cauchy(X_{i,old}, e^{-1+(nfes/\max\_nfes)^2}) \quad (12)$$

As the iterative process $e^{-1+(nfes/\max\_nfes)^2}$ becomes larger, the variation range of $X_{i,new}$ expands, and the diversity of the main population can increase, which is conducive to exploration.

The change rate of the average value of the solution of the most potential subpopulation P_{main} in the last five generations $DivR_{gen}$, which is in (13), will gradually become smaller.

$$DivR_{gen} = \frac{abs(Div_{gen}\text{-Div}_{gen-5})}{\text{Div}_{gen-5}} \quad (13)$$

where Div_{gen} is the diversity of the sub population P_{main}, which is shown in (14):

$$Div_{gen} = \frac{1}{NP_{gen}}(\sum_{i=1}^{NP_{gen}} dis(\overrightarrow{P_{gen,i}} - \overrightarrow{P_{\text{gen},best}})) \quad (14)$$

The index LST is taking into account the limitation of computing resources and the reduction of population diversity in the later period. When the LST is 1, it means that the diversity of the population is low, and the Cauchy local search needs to be added.

$$LST = \begin{cases} 1, \text{e}^{-DivR} > e^{-(nfes/\max\_nfes)^3} \\ 0, otherwise \end{cases} \quad (15)$$

Algorithm 1 AMMADE algorithm.

1) Set the initial global parameters gen=0, population size $popsize$ and population mobility p_m.
2) Calculate the Euclidean distance from each individual to the best individual and sort it order.
3) Set the initial parameters of the subpopulation mF_i, mCR_i, NP_i=$popsize/3$, $i = 1, 2, 3$
4) According to the sorting, divide three sub-populations P_i and use different mutation strategies,P_1 ("DE/rand /1" for the top ranking), P_2 ("DE/current-to-rand/1" for the general ranking) and P_3 ("DE/current-to-pbest/1" for the worst ranking)
5) Repeat the following process until a predefined termination condition is met.
   a) The three subpopulations P_i evolve in parallel with different mutationm_i operators and return to P_i,$fitnessP_i$,NP_i.
   b) Get the $score_i$ according to index SQF_p, and the population with the highest score is the most potential population P_{main}.
   c) Adaptive Local search for P_{main},which is shown in Algorithm 2.
   d) Implement population migration strategy,which is shown in Algorithm 2.
6) Output the solution with the best fitness in the terminal population.

Algorithm 2 Adaptive Local search for P_{main}.

1) **if** LST == 1
   a) Sort P_{main} according to fitness value order.
   b) Perform a local Cauchy search on the top 2% to increase the diversity of P_{main}.
   c) Replace the old individuals with new outstanding individuals.
2) **else if**
   a) Perform a Gaussian local search on the best individual of P_{main} .
   b) Replace the old individuals with new outstanding individuals.
3) **end if**

IV. Experiments

In this section, extensive experiments have been carried out to evaluate the AMMADE's performance. First, in Section IV *A*, the parameter setting of algorithm and function is mentioned. In Section IV *B*, we explore the effectiveness of the proposed strategy. Finally, we compare the performance of our method with related methods.

All algorithms are run with a computer of Intel Core i78700 3.20 GHz CPU, 16 GB RAM. We run each algorithm 51 times on each test problem and record the means (mean) and standard deviations (std) of function values among these runs. On each problem, the best mean fitness values among the algorithms to be compared are marked with boldface in the results.

TABLE I
Summary of the Parameter Values Used in the Proposed Algorithm

Parameters	$popsize$	mF_i	mCR_i	p	p_m
Value	210	0.5	0.5	0.05	0.05

TABLE II
Comparison of Results of MPEDE, AMMADE_1 and AMMADE in Terms of Mean (std) on the CEC2014 Test Functions with D = 30

	MPEDE	AMMADE_1	AMMADE
Functions	Mean (Std)	Mean (Std)	Mean (Std)
F1	**1.08E-03**(7.70E-03)	6.96E-03(4.88E-02)	2.48E-06(1.10E-05)
F2	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F3	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F4	8.30E-04(5.93E-03)	**4.29E-07**(3.06E-06)	**1.56E-11**(7.04E-11)
F5	2.04E+01(4.22E-02)	**2.03E+01**(4.75E-02)	**2.03E+01**(6.46E-02)
F6	**9.00E-01**(1.09E+00)	2.62E+00(1.64E+00)	1.77E+00(1.53E+00)
F7	**3.38E-04**(1.71E-03)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F8	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F9	2.82E+01(7.30E+00)	**2.44E+01**(6.20E+00)	**2.37E+01**(5.07E+00)
F10	1.30E+00(8.26E-01)	**5.28E-02**(5.07E-02)	6.20E-02(3.72E-02)
F11	2.39E+03(4.62E+02)	**1.97E+03**(4.07E+02)	**1.93E+03**(4.07E+02)
F12	5.22E-01(9.50E-02)	**4.05E-01**(8.49E-02)	**4.24E-01**(8.00E-02)
F13	**2.10E-01**(7.42E-02)	**2.19E-01**(3.59E-02)	**2.08E-01**(3.22E-02)
F14	2.37E-01(3.20E-02)	**2.25E-01**(2.59E-02)	**2.22E-01**(2.66E-02)
F15	4.03E+00(8.44E-01)	**3.17E+00**(7.13E-01)	**3.36E+00**(7.79E-01)
F16	9.97E+00(4.38E-01)	**9.76E+00**(4.19E-01)	**9.66E+00**(4.54E-01)
F17	**2.17E+02**(1.55E+02)	3.10E+02(1.68E+02)	**2.80E+02**(1.71E+02)
F18	1.44E+01(5.26E+00)	**1.27E+01**(5.13E+00)	**1.24E+01**(4.04E+00)
F19	**3.81E+00**(5.32E-01)	**3.84E+00**(5.82E-01)	**3.86E+00**(5.47E-01)
F20	**8.66E+00**(2.77E+00)	1.07E+01(4.22E+00)	1.03E+01(3.25E+00)
F21	**1.02E+02**(1.06E+02)	1.44E+02(1.43E+02)	1.48E+02(9.46E+01)
F22	**8.93E+01**(6.37E+01)	9.93E+01(7.16E+01)	**8.12E+01**(6.01E+01)
F23	**3.15E+02**(4.59E-13)	**3.15E+02**(3.21E-12)	**3.15E+02**(3.21E-12)
F24	2.25E+02(3.37E+00)	**2.24E+02**(6.65E-01)	**2.24E+02**(5.87E-01)
F25	**2.00E+02**(2.30E-03)	2.03E+02(2.99E-01)	2.03E+02(4.52E-01)
F26	**1.00E+02**(2.77E-02)	**1.00E+02**(2.98E-02)	**1.00E+02**(3.25E-02)
F27	**3.55E+02**(4.89E+01)	**3.76E+02**(4.30E+01)	**3.66E+02**(4.55E+01)
F28	8.35E+02(3.77E+01)	**7.92E+02**(2.84E+01)	**7.95E+02**(3.00E+01)
F29	**6.84E+02**(1.33E+02)	**6.97E+02**(9.94E+01)	**6.73E+02**(1.51E+02)
F30	7.62E+02(3.87E+02)	6.92E+02(2.16E+02)	**6.75E+02**(3.58E+02)
	13/11/6	4/25/1	+/-/=

A. Experimental Settings

To verify the performance of proposed algorithm, we conduct numerical experiments on the CEC2014 test suites. The data sets of CEC2014 can be categorized into four groups, F1-F3 are uni-modal functions, F4-F16 are the simple multi-modal functions, F17-F22 are the hybrid functions and F23-F30 are composition functions. For all problems, the search space is $[-100,100]^D$. The dimensions of benchmark functions are D = 30 and 100. The values of the optimal

TABLE III
COMPARISON OF RESULTS IN TERMS OF MEAN (STD) ON THE CEC2014 TEST FUNCTIONS WITH D = 30

	JADE	CoDE	SaDE	EPSDE	MPEDE	AMMADE
Functions	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
F1	2.81E+03(3.07E+03)	2.23E+04(1.75E+04)	3.66E+05(2.34E+05)	8.53E+04(5.60E+05)	**1.08E-03**(7.70E-03)	2.48E-06(1.10E-05)
F2	**0.00E+00**(0.00E+00)	5.76E+00(2.44E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F3	1.13E-06(7.51E-06)	1.49E-04(6.86E-05)	2.70E+01(6.12E+01)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F4	**0.00E+00**(0.00E+00)	2.72E+01(2.67E+01)	4.04E+01(3.78E+01)	3.73E+00(2.24E+00)	8.30E-04(5.93E-03)	1.56E-11(7.04E-11)
F5	**2.03E+01**(3.53E-02)	2.06E+01(4.33E-02)	2.05E+01(5.08E-02)	2.04E+01(4.04E-02)	2.04E+01(5.41E-02)	**2.03E+01**(6.46E-02)
F6	1.00E+01(2.12E+00)	2.17E+01(1.84E+00)	5.11E+00(2.01E+00)	1.88E+01(1.58E+00)	**9.00E-01**(1.09E+00)	1.77E+00(1.53E+00)
F7	**1.45E-04**(1.04E-03)	5.65E-04(3.08E-03)	7.13E-03(1.18E-02)	1.45E-03(4.88E-03)	**3.38E-04**(1.71E-03)	**0.00E+00**(0.00E+00)
F8	**0.00E+00**(0.00E+00)	1.86E+01(1.38E+00)	1.95E-02(1.39E-01)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F9	2.69E+01(3.59E+00)	1.38E+02(9.84E+00)	4.20E+01(1.03E+01)	4.36E+01(5.80E+00)	2.82E+01(7.30E+00)	**2.14E+01**(4.81E+00)
F10	**5.72E-03**(1.18E-02)	7.73E+02(8.33E+01)	2.67E-01(4.23E-01)	2.49E-01(2.33E-01)	1.30E+00(8.26E-01)	6.20E-02(3.72E-02)
F11	**1.65E+03**(2.49E+02)	4.82E+03(2.57E+02)	3.22E+03(5.85E+02)	3.53E+03(3.62E+02)	2.39E+03(4.62E+02)	1.93E+03(4.07E+02)
F12	**2.58E-01**(3.33E-02)	1.01E+00(1.56E-01)	7.68E-01(9.97E-02)	5.01E-01(5.48E-02)	5.22E-01(9.50E-02)	4.24E-01(8.00E-02)
F13	**2.06E-01**(2.96E-02)	4.49E-01(5.15E-02)	2.57E-01(4.09E-02)	2.44E-01(3.62E-02)	2.10E-01(7.42E-02)	**2.08E-01**(3.22E-02)
F14	**2.19E-01**(3.42E-02)	2.87E-01(3.77E-02)	**2.29E-01**(3.62E-02)	2.92E-01(7.42E-02)	2.37E-01(3.20E-02)	**2.22E-01**(2.66E-02)
F15	**3.24E+00**(3.44E-01)	1.36E+01(9.60E-01)	4.83E+00(1.79E+00)	5.39E+00(7.94E-01)	4.03E+00(8.44E-01)	**3.36E+00**(7.79E-01)
F16	**9.38E+00**(4.33E-01)	1.16E+01(2.47E-01)	1.10E+01(3.11E-01)	1.11E+01(3.45E-01)	9.97E+00(4.38E-01)	9.66E+00(4.54E-01)
F17	1.62E+04(1.08E+05)	1.47E+03(2.25E+02)	1.27E+04(1.10E+04)	4.27E+04(4.08E+04)	**2.17E+02**(1.55E+02)	**2.80E+02**(1.71E+02)
F18	6.96E+01(3.18E+01)	4.96E+01(5.63E+00)	4.38E+02(6.73E+02)	2.24E+02(4.14E+02)	1.44E+01(5.26E+00)	**1.24E+01**(4.04E+00)
F19	4.34E+00(6.88E-01)	7.12E+00(8.33E-01)	5.33E+00(8.29E+00)	1.33E+01(1.16E+00)	**3.81E+00**(5.32E-01)	**3.86E+00**(5.47E-01)
F20	2.39E+03(2.61E+03)	3.06E+01(3.88E+00)	1.37E+02(1.90E+02)	5.61E+01(7.03E+01)	**8.66E+00**(2.77E+00)	1.03E+01(3.25E+00)
F21	2.92E+03(1.89E+04)	7.19E+02(1.28E+02)	3.73E+03(5.32E+03)	8.04E+03(9.02E+03)	**1.02E+02**(1.06E+02)	1.48E+02(9.46E+01)
F22	1.59E+02(7.08E+01)	1.20E+02(5.20E+01)	1.37E+02(5.70E+01)	2.25E+02(9.13E+01)	**8.93E+01**(6.37E+01)	**8.12E+01**(6.01E+01)
F23	3.15E+02(3.21E-12)	3.15E+02(6.02E-07)	3.15E+02(2.76E-12)	**3.14E+02**(1.38E-12)	3.15E+02(4.59E-13)	3.15E+02(3.21E-12)
F24	2.25E+02(1.14E+00)	2.26E+02(8.06E-01)	2.26E+02(2.51E+00)	2.29E+02(6.06E+00)	2.25E+02(3.37E+00)	**2.24E+02**(5.87E-01)
F25	2.05E+02(1.89E+00)	2.00E+02(7.28E-02)	2.08E+02(3.17E+00)	**2.00E+02**(1.73E-01)	2.00E+02(2.30E-03)	2.03E+02(4.52E-01)
F26	**1.00E+02**(3.50E-02)	1.00E+02(4.68E-02)	1.02E+02(1.40E+01)	1.00E+02(4.27E-02)	**1.00E+02**(2.77E-02)	**1.00E+02**(3.25E-02)
F27	**3.39E+02**(4.75E+01)	4.01E+02(2.22E-01)	4.04E+02(3.19E+01)	8.44E+02(9.13E+01)	3.55E+02(4.89E+01)	3.66E+02(4.55E+01)
F28	7.94E+02(3.56E+01)	9.39E+02(2.39E+01)	8.74E+02(2.62E+01)	**3.96E+02**(1.29E+01)	8.35E+02(3.77E+01)	7.95E+02(3.00E+01)
F29	7.66E+02(2.14E+02)	5.83E+02(2.03E+02)	1.08E+03(2.20E+02)	**2.14E+02**(1.31E+00)	6.84E+02(1.33E+02)	6.73E+02(1.51E+02)
F30	1.29E+03(3.69E+02)	1.15E+03(1.31E+02)	1.51E+03(5.32E+02)	**5.36E+02**(1.38E+02)	7.62E+02(3.87E+02)	6.75E+02(3.58E+02)
	14/9/7	29/0/1	25/4/1	21/4/5	13/11/6	+/-/=

solutions are known in advance for all benchmark functions. The maximum number of objective function evaluations is D×10,000. The parameter settings of AMMADE are listed in Table I, and the control parameters of other algorithms are set as suggested in the corresponding papers.

B. Exploring the Proposed Strategies

In this section, we explore the proposed strategies by comparing AMMADE with MPEDE and its variant: MPEDE, in which random sorting mutation strategy and reward population strategy are used. AMMADE_1 uses adaptive mutation operator strategy based on multi-population and AMMADE uses adaptive mutation operator strategy based on multi-population and adaptive local search strategy.

The Nonparametric Wilcoxon rank-sum test at a 0.05 significance level has been performed between AMMADE and each algorithm to be compared on each benchmark function. The sign '+' in the results indicates that the performance of AMMADE is significantly better than the corresponding algorithm, '-' vice versa and '=' denotes there is no significant difference between the performance.

The results (error values $f - f*$) on 30 dimensions are shown in Table II, in which f* is the best fitness value. From the results, we can see that among the 30 benchmark functions, AMMADE achieves the best mean values on 23 functions. The AMMADE results are significantly better than the other two algorithms. It can be seen from the results that the strategies proposed are effective.

C. Comparisons with Related Algorithms

In this section, we compared AMMADE to the following five DE variants:

TABLE IV
COMPARISON OF RESULTS IN TERMS OF MEAN (STD) ON THE CEC2014 TEST FUNCTIONS WITH D = 100

	JADE	CoDE	SaDE	EPSDE	MPEDE	AMMADE
Functions	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
F1	**1.82E+05**(5.94E+04)	7.04E+06(2.01E+06)	6.84E+06(1.91E+06)	3.19E+05(1.21E+05)	2.67E+05(1.05E+05)	2.92E+05(1.14E+05)
F2	3.80E-03(1.05E-02)	1.68E+03(9.97E+02)	2.99E+04(6.90E+03)	4.36E+03(2.09E+04)	**0.00E+00**(0.00E+00)	**0.00E+00**(0.00E+00)
F3	5.08E+03(4.66E+03)	6.33E+01(6.83E+01)	8.94E+01(9.74E+01)	**1.40E-02**(4.55E-02)	7.23E+02(8.01E+02)	9.97E+02(1.03E+03)
F4	1.38E+02(4.75E+01)	1.68E+02(2.77E+01)	3.80E+02(3.66E+01)	1.41E+02(4.35E+01)	9.46E+01(5.73E+01)	**8.25E+01**(5.79E+01)
F5	**2.04E+01**(1.40E-01)	2.12E+01(2.78E-02)	2.10E+01(2.79E-02)	2.11E+01(4.77E-02)	2.08E+01(8.48E-02)	2.07E+01(9.86E-02)
F6	4.65E+01(1.46E+01)	1.72E+01(4.35E+00)	8.92E+01(2.73E+01)	1.31E+02(4.40E+00)	**4.32E+01**(4.81E+00)	**4.34E+01**(7.02E+00)
F7	**4.29E-03**(9.45E-03)	3.31E-07(2.31E-07)	6.39E-02(2.53E-02)	4.78E-03(8.88E-03)	**1.74E-03**(4.73E-03)	**1.59E-03**(4.48E-03)
F8	**0.00E+00**(0.00E+00)	3.41E+02(1.64E+01)	2.93E+02(1.05E+01)	9.12E+01(1.39E-01)	2.15E-01(5.38E-01)	5.07E-01(1.17E+00)
F9	**1.56E+02**(1.88E+01)	4.81E+02(1.83E+02)	7.43E+02(2.06E+01)	6.43E+02(4.47E+01)	**1.56E+02**(2.15E+01)	**1.59E+02**(2.52E+01)
F10	**1.04E-02**(7.14E-03)	1.22E+04(6.22E+02)	1.06E+04(4.33E+02)	1.03E+04(1.94E+03)	7.04E-01(5.02E-01)	3.37E+00(1.87E+00)
F11	**1.07E+04**(5.05E+02)	2.65E+04(7.20E+02)	2.37E+04(4.71E+02)	2.67E+04(1.27E+03)	1.13E+04(1.11E+03)	1.12E+04(9.75E+02)
F12	**3.35E-01**(2.53E-02)	2.68E+00(1.46E-01)	1.91E+00(1.24E-01)	1.96E+00(1.67E-01)	7.72E-01(1.64E-01)	4.51E-01(1.54E-01)
F13	4.07E-01(4.56E-02)	6.00E-01(4.58E-02)	4.52E-01(2.48E-02)	4.80E-01(5.83E-02)	**3.67E-01**(3.84E-02)	**3.66E-01**(3.92E-02)
F14	**3.08E-01**(2.25E-02)	3.55E-01(2.41E-02)	**3.09E-01**(1.26E-02)	3.34E-01(3.16E-02)	**3.02E-01**(2.16E-02)	**3.01E-01**(2.47E-02)
F15	3.57E+01(5.73E+00)	6.67E+01(3.42E+00)	7.14E+01(5.23E+00)	9.15E+01(1.44E+01)	**1.76E+01**(2.78E+00)	**1.81E+01**(3.59E+00)
F16	3.99E+01(5.92E-01)	4.57E+01(2.87E-01)	4.44E+01(3.03E-01)	4.59E+01(5.46E-01)	4.04E+01(6.95E-01)	**3.94E+01**(1.09E+00)
F17	2.50E+04(8.01E+03)	2.97E+05(1.40E+05)	**1.40E+04**(5.01E+03)	4.48E+06(8.83E+06)	2.15E+04(1.04E+04)	2.66E+04(9.90E+03)
F18	1.02E+03(9.35E+02)	5.11E+02(5.34E+02)	4.24E+02(2.87E+02)	3.93E+03(5.28E+03)	**2.84E+02**(4.22E+01)	**2.81E+02**(8.76E+01)
F19	9.44E+01(2.11E+01)	9.47E+01(1.30E+00)	7.48E+01(2.36E+01)	**5.56E+01**(2.53E+01)	9.33E+01(2.15E+01)	9.63E+01(8.44E+00)
F20	5.85E+03(1.27E+04)	2.69E+02(1.12E+02)	**2.62E+02**(5.56E+01)	1.43E+03(4.50E+03)	4.91E+02(1.52E+02)	5.97E+02(3.05E+02)
F21	7.87E+03(3.70E+03)	7.53E+04(4.61E+04)	**2.88E+03**(1.23E+03)	3.62E+05(2.29E+05)	4.84E+03(2.43E+03)	6.91E+03(4.09E+03)
F22	1.53E+03(2.28E+02)	2.01E+03(5.13E+02)	2.63E+03(1.91E+02)	2.16E+03(3.67E+02)	1.70E+03(4.20E+02)	**1.64E+03**(4.32E+02)
F23	3.48E+02(1.96E-11)	3.48E+02(4.14E-07)	3.48E+02(1.03E-01)	**3.45E+02**(9.19E-13)	3.48E+02(0.00E+00)	3.48E+02(9.19E-13)
F24	3.97E+02(4.86E+00)	3.68E+02(3.26E+00)	3.71E+02(3.38E+00)	4.07E+02(8.30E+00)	3.96E+02(5.43E+00)	**3.91E+02**(4.40E+00)
F25	2.69E+02(7.65E+00)	2.03E+02(7.96E-01)	2.00E+02(6.74E-04)	2.59E+02(3.31E+01)	2.15E+02(2.12E+01)	**2.02E+02**(8.49E+00)
F26	2.00E+02(1.89E-02)	1.92E+02(2.70E+01)	2.00E+02(1.31E-02)	**1.40E+02**(5.04E+01)	1.98E+02(1.40E+01)	1.98E+02(1.40E+01)
F27	1.12E+03(1.29E+02)	4.79E+02(5.65E+01)	7.89E+02(5.68E+01)	3.76E+03(6.07E+01)	1.14E+03(1.12E+02)	**9.54E+02**(1.17E+02)
F28	2.39E+03(2.73E+02)	2.69E+03(2.58E+02)	8.80E+03(9.07E+02)	**8.68E+02**(2.79E+02)	2.31E+03(2.60E+02)	2.43E+03(4.32E+02)
F29	1.37E+03(8.84E+01)	1.97E+03(1.45E+02)	2.06E+03(2.22E+02)	**2.64E+02**(2.56E+01)	1.05E+03(2.22E+02)	1.22E+03(2.18E+02)
F30	8.42E+03(1.19E+03)	5.49E+03(1.16E+03)	6.79E+03(1.62E+03)	**2.60E+03**(3.81E+02)	7.08E+03(1.46E+03)	7.13E+03(1.10E+03)
	14/9/7	22/1/7	21/2/7	21/2/7	8/17/5	+/-/=

1) JADE [5]: DE with adaptive control parameters and optional external archive;
2) CoDE [6]: DE with composite trial vector generation strategies and control parameters;
3) SaDE [7]: DE with strategy adaptation;
4) EPSDE [4]: DE with ensemble of parameters and mutation strategies;
5) MPEDE [9]: DE with multi-population based ensemble of mutation strategies.

Tables III and IV show the comparison results (error values $f - f*$) of the six algorithms. By analyzing the experimental results from 30D, conclusions are given as follows.

Firstly, for uni-modal functions F1–F3, MPEDE shows the best performance on F1, and AMMADE is also competitive compared to other algorithms on F1. For F2, several related algorithms except CoDE can find the optimal value. For F3, only EPSDE, MPEDE and AMMADE can find the optimal value. Secondly, for basic multi-modal benchmark functions F4-F16, JADE performed the best, achieving the best mean fitness on 11 functions (F4-F5, F7-F8, F10-F16). AMMADE's performance is also quite competitive, with excellent performance on 7 functions (F5, F7-F9, F13-F15). Thirdly, for the hybrid functions F17-F22, the best performer is MPEDE, which has the best mean fitness on 5 functions (F17, F19-F22). AMMADE performs well on 4 functions (F17-F19, F23). This can reflect that AMMADE is also competitive for hybrid functions. Finally, for the complex composition functions F23-F30, EPSDE achieves the best results, with excellent performance on 5 functions (F23, F25, F28-F30). AMMADE, MPEDE and JADE perform well on 2 functions (F24, F26), one function (F26) and two functions (F26-F27), respectively. Based on the experimental results and analysis on the CEC2014 test suit, we can see that AMMADE is competitive on various functions with related algorithms.

Algorithm 3 Implement population migration strategy.

1) Calculate and sort the Euclidean distance from each individual to the best individual.
2) Re-divide sub-populations according to sorting
3) **if** $P_1 == P_{main}$
    a) Migrate the top p_m individuals from P_3 to P_1.
    b) Migrate the top p_m individuals from P_2 to P_1.
4) **else if** $P_2 == P_{main}$
    a) Migrate the top p_m individuals from P_3 to P_2.
    b) Migrate the top p_m individuals from P_1 to P_2.
5) **else if** $P_3 == P_{main}$
    a) Migrate the top p_m individuals from P_2 to P_3.
    b) Migrate the top p_m individuals from P_1 to P_3.
6) **end if**
7) Sort according to fitness value, replace the best individuals of the three subpopulations with the best individuals of the entire population
8) update P_i,$fitnessP_i$,NP_i.

V. Conclusions

In this paper, we presented a multi population based DE with adaptive mutation and local search for global optimization. In the proposed method, a grouping and sorting method is used to partition the population and the individual in different subpopulation will use different mutation strategies at each stage according to the ranking of Euclidean distance. Secondly, a resource allocation method is proposed such that a better mutation strategy will get more computing resources. In addition, an adaptive local search strategy is proposed, in which different local search strategies will be dynamically employed to improve the solution at different stages. The experimental results clearly show the merits of the proposed strategies and the resulting method could outperform the related methods to be compared.

References

[1] K. M. Sallam, S. M. Elsayed, R. K. Chakrabortty, and M. Ryan, "Evolutionary framework with reinforcement learning-based mutation adaptation," *IEEE Access*, vol. 8, 2020.

[2] X. F. Liu, Z. H. Zhan, Y. Lin, W. N. Chen, Y. J. Gong, T. L. Gu, H. Q. Yuan, and J. Zhang, "Historical and heuristic-based adaptive differential evolution," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, pp. 1–13, 2018.

[3] S. Kitayama, M. Arakawa, and K. Yamazaki, "Differential evolution as the global optimization technique and its application to structural optimization," *Applied Soft Computing*, vol. 11, no. 4, pp. 3792–3803, 2011.

[4] R. Mallipeddi and P. N. Suganthan, "Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies," in *Swarm, Evolutionary, and Memetic Computing - First International Conference on Swarm, Evolutionary, and Memetic Computing, SEMCCO 2010, Chennai, India, December 16-18, 2010. Proceedings*, 2010.

[5] 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.

[6] W. Yong, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[7] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *IEEE Congress on Evolutionary Computation*, 2005.

[8] X. Li, L. Wang, Q. Jiang, and N. Li, "Differential evolution algorithm with multi-population cooperation and multi-strategy integration," *Neurocomputing*, vol. 421, no. 1, pp. 285–302, 2021.

[9] G. Wu, R. Mallipeddi, P. N. Suganthan, W. Rui, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Information Sciences An International Journal*, vol. 329, no. C, pp. 329–345, 2016.

[10] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, pp. 302–312, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523121730807X

[11] T. Rogalsky and R. W. Derksen, "Hybridization of differential evolution for aerodynamic design," 2000.

[12] D. Molina, A. Latorre, and F. Herrera, "Shade with iterative local search for large-scale global optimization," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, 2018.

[13] R. Tanabe and A. Fukunaga, "Evaluating the performance of shade on cec 2013 benchmark problems," in *2013 IEEE Congress on evolutionary computation*. IEEE, 2013, pp. 1952–1959.

[14] J. Brito, L. Ochi, F. Montenegro, and N. Maculan, "An iterative local search approach applied to the optimal stratification problem," *International Transactions in Operational Research*, vol. 17, no. 6, pp. 753–764, 2010.

[15] X. Wang, M. Sheng, K. Ye, J. Lin, J. Mao, S. Chen, and W. Sheng, "A multilevel sampling strategy based memetic differential evolution for multimodal optimization," *Neurocomputing*, vol. 334, no. MAR.21, pp. 79–88, 2019.