Improvement of Differential Evolution Multi-objective Optimization Algorithm Based on Decomposition

Jiaxin Han1,a  Manman He2,b  Xiaoxiao Wang3,c

1School of Computer Science, Xi’an Shiyou University,710065
2School of Computer Science, Xi’an Shiyou University,710065
3School of Computer Science, Xi’an Shiyou University,710065
ajxhan@xsyu.edu.cn  bmanmanhe@yeah.net  cxxwang@xsyu.edu.cn

ABSTRACT: Multi-objective optimization problem (MOP) is a challenging field of scientific research in real-life. The effective way to solve multi-objective optimization problems is Multi-objective Evolutionary Algorithm (MOEA). In this paper, enhancements to a Multi-objective Evolutionary algorithm MOEA/D-DE are proposed. The proposed improvement points help to improve both population distribution and algorithmic local search capabilities. In an existing study, in order to better distribute the population, the Monte Carlo method was used for population initialization. Adaptive differential evolution operators are used to improve the local search ability of the algorithm. The algorithm was tested on widely used ZDT and DTLZ family test problems. The experimental results show that the proposed algorithm is better than MOEA/D-DE and has better performance than other excellent multi-objective algorithms.

1. INTRODUCTION
A multi-objective optimization problem (MOP) can be described as follows:

\[
\begin{align*}
\text{min} & \quad y = F(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T \\
\text{s.t.} & \quad g_i(x) \leq 0, i = 1, 2, \ldots, q \\
& \quad h_j(x) = 0, j = 1, 2, \ldots, p
\end{align*}
\]

where \( x = (x_1, x_2, \ldots, x_n)^T \in \mathbb{X} \subseteq \mathbb{R}^n \) is n-dimensional decision vector \( \mathbb{X} \) is n-dimensional decision space. \( y = (y_1, y_2, \ldots, y_m)^T \in \mathbb{Y} \subseteq \mathbb{R}^m \) is m-dimensional objective vectors, \( \mathbb{Y} \) is m-dimensional objective space. Objective function \( F(x) \) defines a mapping function from the decision space to the objective space. \( g_i(x) \leq 0 (i = 1, 2, \ldots, q) \) and \( h_j(x) = 0 (j = 1, 2, \ldots, p) \) are inequality constrains and equality constrains, respectively.

The problems of the real world involve multi-objective problems, and most of the problems defined in equation (1) are conflicting. A single solution is not available by minimizing all goals. Therefore, MOP’s solution is not to find a single optimal solution, but to find a set of trade-off solutions. The traditional approach to solving multi-objective problems is to translate multi-objective problems into single-objective problems, including weighted sum and constrained methods. The shortcoming of these methods is that some parameters need to be determined empirically.
As a heuristic search algorithm, the evolutionary algorithm optimizes a group of individuals at the same time, and selects the best individual as the next generation population through crossover, mutation, and can generate multiple approximate solutions in the optimization process. The contemporary evolutionary multi-objective optimization algorithm [1] introduces new evolution paradigms, such as multi-objective optimization based on particle swarm optimization, multi-objective optimization based on artificial immune system, multi-objective optimization based on distributed estimation algorithm [1] and based on decomposition evolutionary multi-objective optimization algorithm. In 2007, Zhang et al. [2] proposed a decomposition-based multi-objective evolutionary algorithm (MOEA/D). In 2009, Li [3] proposed MOEA/D based on MOEA/D-DE to solve more complex multi-objective optimization problems. The MOEA/D-DE algorithm is a typical multi-objective optimization algorithm based on decomposition. The essence of the algorithm lies in the unique decomposition of the object space. The entire object space is decomposed into multiple sub-spaces through a series of uniformly distributed weight vectors. Optimizing the sub-target space greatly reduces the complexity of the algorithm. The MOEA/D-DE algorithm is presented as shown in Algorithm 1.

Compared with traditional evolutionary algorithms, the computational complexity is lower and can converge quickly. However, there are problems such as uneven distribution of populations and weak local search ability of algorithms.

In this paper, MOEA/D-MC-SA was proposed in an attempt to eliminate the potential weaknesses of its predecessor and to combine the latest results to design a powerful and up-to-date Evolutionary Multi-Objective Optimization Algorithm (MOEA). The main differences between MOEA/D-DE and MOEA/D-MC-SA are as following:

- Use the Monte Carlo method to initialize the population to ensure a better population distribution
- Use differential evolution operators with adaptive parameters to improve the local search ability of the algorithm

The structure of this paper is as follows. Section 2 introduces the proposed MOEA/D-MC-SA algorithm. Section 3 presents experimental studies on MOEA/D-MC-SA. Section 4 concludes this paper.

**Algorithm 1: MOEA/D-MC-SA algorithm**

Input: \( N \) (population size), \( T \) (the number of the weight vectors in the neighborhood of each weight vector)

Output: \( p^1, \ldots, p^\text{max} \)

**Step 1:** Generate a randomly initialized population and a uniformly distributed weight vector;

**Step 2:** use the Tchebycheff method [4] to associate each of the weight vectors with a nearest individual and initialize its neighborhood;

**Step 3:** Select individuals in the neighborhood as the parent population to perform differential operations and polynomial mutations [5, 6] to generate progeny populations;

**Step 4:** Update the sub-questions under each weight and update its neighborhood information;

**Step 5:** If terminating conditions are satisfied, end; else return to 2.

2. **THE PROPOSED ALGORITHM**

Firstly, we point out the problems with the MOEA/D-DE algorithm, then, introduce some key techniques used in the proposed algorithm, and finally give the details of the proposed algorithm.

Compared with traditional evolutionary algorithms, MOEA/D-DE has lower computational complexity and can converge quickly. However, there are problems such as uneven population
distribution and weak local search ability of the population. In this paper, two improvements are proposed for the shortcomings of MOEA/D-DE.

The first one is using the Monte Carlo method to generate the initial population. Because it is very important in genetic algorithms that population initialization affects the global convergence speed of the algorithm and the quality of the solution. In the absence of any historical information available, the general algorithm uses a random initialization method. Because this method cannot guarantee that the initial population is widely distributed in the whole problem space, which affects the efficiency of algorithm execution in a way. The last one is employing an adaptive differential evolution operator. Since the setting of the scaling factor $F$ and the crossover probability $C_r$ in the differential evolution operator is problem-dependent, the local search ability of the population will decrease according to the test problem in the evolution process.

2.1. Key Techniques Used in the Proposed Algorithm

2.1.1. Monte Carlo Method

The Monte Carlo method, also known as stochastic simulation method or statistical test method, is a calculation method based on "random number". It simulates the process by continuously generating a sequence of random numbers. The calculation formula is as follows:

$$ a_x = a/2 \quad b_x = b/2 $$  \hspace{1cm} (2)

$$ mu = a_x + b_x \quad sig = b_x - a_x $$  \hspace{1cm} (3)

$$ r = mu + sig \times (rand(a,b) - 1) $$  \hspace{1cm} (4)

Where $a$ is the upper boundary, $b$ is the lower boundary, and $rand(a,b)$ produces the random number in the interval $(a,b)$.

2.1.2. Adaptive Difference Operator

Brest et al. [7] proposed an adaptive differential evolution algorithm. In DE, the scaling factor $F$ and the hybridization probability $C_r$ are encoded and evolved along with the population individual. The initial values of $F$ and $C_r$ are 0.5 and 0.9, respectively. The mutation operation randomly takes values of a certain interval. When the probability is satisfied, the hybridization operation takes the variant individual, otherwise the original value is taken.

The parameter crosser operator can be described by the following formula:

$$ \text{oldCr} = C_r \quad \text{oldF} = F $$  \hspace{1cm} (5)

$$ \text{newCr} = \begin{cases} 0.0 + 1.0 \times \text{rand}[0,1] & \text{if rand}[0,1] < r_1 \\ \text{oldCr} & \text{otherwise} \end{cases} $$  \hspace{1cm} (6)

$$ \text{newF} = \begin{cases} 0.1 + 0.9 \times \text{rand}[0,1] & \text{if rand}[0,1] < r_2 \\ \text{oldF} & \text{otherwise} \end{cases} $$  \hspace{1cm} (7)

2.2. The Framework of MOEA/D-MC-SA

The proposed method is based on MOEA/D-DE, initialized by Monte Carlo method, and mutated by adaptive differential evolution operator. The MOEA/D-MC-SA algorithm is presented as shown in Algorithm 2.

| Algorithm 2 : MOEA/D-MC-SA algorithm |
|--------------------------------------|
| **Input:** $N$ (population size), $T$ (the number of the weight vectors in the neighborhood of each weight vector) |
| **Output:** $\rho^1, \ldots, \rho^N$ |

**Step 1:** Initialize:

**Step 1.1:** For each weight vector $\chi$, calculate the Euclidean distance between it and other weight vectors, and find the $T$ weight vector nearest to it. The $T$ weight vectors are the neighborhood of
For any $i = 1, \ldots, N$, there are $B(i) = (i, \ldots, i_N)$, $x^i_1, \ldots, x^i_N$ are the nearest T weight vectors;

**Step 1.2:** Using the Monte Carlo method initial population $x', \ldots, x^N$ and calculating the objective function of the response $F(x')$;

**Step 1.3:** Initialize the reference point $z = (z_1, \ldots, z_N)$, where $z_j = \min_{x \in X} f_j(x)$ represents the minimum value of each dimension of the object space;

**Step 2:** Update: for $i = 1, \ldots, N$

**Step 2.1:** Select the range of matching or updating: selecting individuals from the neighborhood of the sub-question to make the parent’s choice;

**Step 2.2:** Genetic operation: randomly select three individuals with indices of $r_1, r_2, r_3$ from $P$, perform adaptive differential operations on individuals $x^{r_1}, x^{r_2}, x^{r_3}$ generate a new solution $y'$, and perform polynomial mutation operations on the new solution $y'$, generating a new solution $y$;

**Step 2.3:** Fix the new solution: If a dimension in $y$ exceeds the search boundary, correcting the new solution $y$ according to the specific problem, so that each dimension of the new solution is within the search scope;

**Step 2.4:** Update the reference point $z$: For any $j = 1, \ldots, m$, if $z_j > f_j(y)$, then set $z_j = f_j(y)$;

**Step 2.5:** Update the neighborhood solution: For any $j \in P$, if $g(y|\lambda', z) \leq g(x'|\lambda', z)$, then set $x' = y$;

**Step 3:** Determine the termination condition: If the termination condition is met, the algorithm stops and outputs $\{x', \ldots, x^N\}$ and $\{F(x'), \ldots, F(x^N)\}$, otherwise, go to Step 2.

### 3. Experimental Analysis

So as to evaluate the effectiveness of the proposed algorithm, we compare its performance with ZDT and DTLZ family test function. And a comparison is making among PESA-II algorithm [8], MISA algorithm [9], NSGS-II [10, 11] algorithm, and MOEA/D algorithm [2] in terms of IGD. For the special parameters of different algorithms are from Table 1. For each test function, the algorithm runs 10 times independently on identical computers (Intel (R) Core(TM) i7-6700 CPU (3.41 GHz) 8 GB RAM Windows 10 system).

#### 3.1. ZDT and DTLZ Family Test Function

For the performance test of the multi-objective optimization algorithm, the ZDT test function set [12] proposed by Zitzler et al. and the DTLZ test function set [13] proposed by Deb are usually used for testing. This section mainly tests the target problems of ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 in the ZDT test set and the DTLZ1, DTLZ2 functions in the DTLZ test set.

#### 3.2. Parameter Setting

In all algorithms, for a two-dimensional problem, the population size is taken as 100, and the three-dimensional problem is 210. The population size setting is related to the design of the MOEA/D-DE weight. The algorithm runs 10 times independently for each test problem.
The parameters of each algorithm are set as follows. The general parameters of all algorithms: $m$ represents the dimension of the objective function is usually 2 or 3, the corresponding population size is 100 or 210, $n$ represents the dimension of the decision variable, and $\text{iter}$ represents the number of iterations is 250.

Table 1 is the unique parameters of different algorithms. Where $C_r$ and $F$ are the control parameters of the differential evolution operator, $C_r$ is the crossover probability, $F$ is the scaling factor, $\eta_m$ and $p_m$ are the distribution index and mutation probability of the polynomial variation, respectively. $\eta_c$ and $p_c$ are the distribution indices and crossover probabilities of the simulated binary crossover (SBX), respectively. $T$ represents the number of each weight neighborhood in MOEA/D-DE, $N_e$ is secondary population size, the object dimension is 100 in 2-dimensions and 210 in 3-dimensions, respectively, and $Ngs$ is number of grid subdivisions, $cl$ is coding length.

Table 1. Specific parameters of different algorithms.

| Para. | PESA-II | MISA | NSGA-II | MOEA/D | MOEA/D-MC-SA |
|-------|---------|------|---------|--------|--------------|
| $C_r$ | -       | -    | -       | -      | -            |
| $F$   | -       | -    | -       | -      | 0.9          |
| $\eta_m$ | 20    | 20   | 20      | 20     | 20           |
| $p_m$ | $1/n$  | -    | $1/n$   | $1/n$  | $1/n$        |
| $\eta_c$ | -     | -    | 20      | 20     | 20           |
| $p_c$ | -       | -    | 1       | 1      | 1            |
| $T$   | 20      | -    | -       | -      | -            |
| $N_e$ | 100/210 | 100/210 | -     | -      | -            |
| $Ngs$ | 32      | 25   | -       | -      | -            |
| $cl$  | -       | 30   | -       | -      | -            |

3.3. Performance Metrics

Here, the IGD [14, 15] indicator is used as a performance indicator for the evaluation algorithm. Next, we briefly introduce the IGD indicator. IGD, also known as the reverse generation distance, is able to evaluate the convergence and diversity of the optimal solutions found by the algorithm. We assume that a set of optimal solutions uniformly distributed over the PF is the optimal solution found by the algorithm. Then, the IGD indicators are defined as follows:

$$IGD(P^*, P^C) = \frac{\sum_{v \in P^C} d(v, P^*)}{|P^*|}$$  \hspace{1cm} (8)

Where $d(v, P^*) = \min_{u \in P} \|F(v) - F(u)\|$ is the distance between individuals $v$ and $P^*$, and $|P^*|$ is the cardinality of $P^*$, representing the number of individuals in $P^*$. The smaller the IGD value, the closer it is to the real PF.

3.4. Experimental Results and Analysis

In this paper, we compare the MOEA/D-MC-SA with the existing four evolutionary multi-objective optimization algorithms PESA-II, MISA, NSGA-II and MOEA/D. For each test function, the algorithm runs independently 10 times under each environment parameter setting. The mean and variance of the IGD values of the five algorithms are shown in Table 2.

It can be seen from Table 2 that the proposed algorithm outperforms the other four algorithms in the test problems ZDT2, ZDT4, ZDT6, DTLZ1 and DTLZ2. In a word, the proposed MOEA/D-MC-SA algorithm is superior to other comparison algorithms in most test functions.
### Table 2. Experimental results of different algorithm comparisons.

| Prob. | statistics | PESA-II | MISA | NSGA-II | MOEA/D | MOEA/D-MC-SA |
|-------|------------|---------|------|---------|--------|--------------|
| ZDT1  | mean       | 0.00577 | 0.08558 | **0.0043** | 0.0048 | 0.0045       |
|       | variance   | **1.01714E-07** | 4.21649E-05 | 3.7153e-04 | 2.6414e-04 | 1.9637e-04 |
| ZDT2  | mean       | 0.00575 | 0.12812 | 0.0043 | 0.0041 | **0.0040** |
|       | variance   | **5.73028E-08** | 0.000124648 | 1.7162e-04 | 8.3271e-05 | 4.7228e-05 |
| ZDT3  | mean       | 0.1231 | 0.08335 | **0.0046** | 0.0165 | 0.0135       |
|       | variance   | 0.046772382 | **3.09038E-05** | 2.1113e-04 | 0.0120 | 0.0089       |
| ZDT4  | mean       | 0.00917 | 1.01727 | 0.0098 | 0.0050 | **0.0048** |
|       | variance   | **2.07992E-06** | 0.160700464 | 0.0173 | 0.0010 | 5.5408e-04 |
| ZDT6  | mean       | 0.00366 | 0.82628 | 0.0023 | **0.0019** | **0.0019** |
|       | variance   | **1.95901E-07** | 0.001563093 | 0.3241 | 1.7428e-05 | 1.6813e-05 |
| DTLZ1 | mean       | 77.0498 | 81.2972 | 0.1216 | 0.0570 | **0.0214** |
|       | variance   | 26.01056791 | 7.197038871 | 0.139698244 | 0.0901 | **0.0025** |
| DTLZ2 | mean       | 0.08001 | 0.05402 | 0.0491 | 0.0454 | **0.0453** |
|       | variance   | **7.73508E-05** | 3.41675E-06 | 0.0031 | 3.3875e-04 | 1.6358e-04 |

### 4. CONCLUSION

Based on the MOEA/D-DE algorithm, this paper proposes a new MOEA/D-MC-SA algorithm. Compared with the results of the existing four evolutionary multi-objective optimization algorithms, the algorithm achieves better results on most test problems, and has a more uniform population distribution and local search ability has been further improved. Further research applied the algorithm to complex high-dimensional multi-objective optimization problems, and further improved the local search ability of the algorithm.

### ACKNOWLEDGMENT

Our thanks to the Shaanxi Provincial International Co-operation and Exchanges in Science and Technology Plan Project (2016kw-047) and Shaanxi Provincial Natural Science Foundation (2018JM6093).

### REFERENCES

[1] Gong MG, Jiao LC, Yang DD, Ma WP. Research on evolutionary multi-object optimization algorithm. Journal of software 2009, 20(2):271-289.

[2] H. Li and Q. Zhang. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II[J], IEEE Transactions on Evolutionary Computation, 2009, vol. 13, no. 2, pp. 284–302.

[3] Brest J, Greiner S, Boskovic B, et al. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems[J]. IEEE Transaction on Evolutionary Computation, 2006, 10(6):646-657.

[4] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on evolutionary computation, 2007, 11(6): 712-731.

[5] Fu Li et al, Improvement of Non-dominated Sorting Genetic Algorithm, Computer & Digital Engineering, 2011, 39(2):11-15
[6] Zitzler E., Deb K, et al. Comparison of multiobjective evolutionary algorithms: Empirical results, IEEE Transactions on Evolutionary Computation, 2000, vol. 8, no. 2, pp.173-195.

[7] Brest J, Greiner S, Boskovic B, et al. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems [J]. IEEE Transactions on Evolutionary Computation, 2006,10(6):646-657.

[8] Corne DW, Jerram NR, Knowles JD, Oates MJ. PESA-II: Region-Based selection in evolutionary multi-objective optimization. In: Spector L, Goodman ED, Wu A, Langdon WB, Voigt HM, Gen M, eds. Proc. of the Genetic and Evolutionary Computation Conf.,GECCO 2001. San Francisco: Morgan Kaufmann Publishers, 2001. 283–290.

[9] Coello C A C, Cortés N C. An approach to solve multiobjective optimization problems based on an artificial immune system [C]//first international Conference on artificial immune systems (ICARIS’2002). 2002: 212-221.

[10] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II [J]. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.

[11] Deb K, Agrawal S, Pratap A, et al. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II[C]//International Conference on Parallel Problem Solving From Nature. Springer, Berlin, Heidelberg, 2000: 849-858.

[12] Zitzler E., Deb K, et al. Comparison of multiobjective evolutionary algorithms: Empirical results, IEEE Transactions on Evolutionary Computation, 2000, vol. 8, no. 2, pp.173-195.

[13] K. Deb. Scalable test problems for evolutionary multi-objective optimization. 2001.

[14] Zhou A, Zhang Q, Jin Y, et al. Global multiobjective optimization via estimation of distribution algorithm with biased initialization and crossover. Proceedings of the 9th annual conference on Genetic and evolutionary computation. ACM, 2007: 617-623.

[15] Margarita Reyes Sierra, Carlos A. Coello Coello. Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ε-Dominance. Evolutionary Multi-Criterion Optimization. Third International Conference, Emo. 2005:505-519.