IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 16, 2019, accepted May 12, 2019, date of publication May 20, 2019, date of current version July 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2917899

Two-Archive Evolutionary Algorithm Based on Multi-Search Strategy for Many-Objective Optimization

CAI DAI
School of Computer Science, Shaanxi Normal University, Xi'an 710119, China
e-mail: cdai0320@snnu.edu.cn

This work was supported in part by the National Natural Science Foundation of China under Grant 61502290, Grant 61806120, Grant 61672334, Grant 61673251, and Grant 61401263, in part by the China Postdoctoral Science Foundation under Grant 2015M582606, and in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2016JQ6045.

ABSTRACT Taking both convergence and diversity into consideration, this paper proposes a two-archive an evolutionary algorithm based on multi-search strategy (TwoArchM) to cope with many-objective optimization problems. The basic idea is to use two separate archives to balance the convergence and diversity and use a multi-search strategy to improve convergence and diversity. To be specific, two updated strategies are adopted to maintain diversity and improve the convergence, respectively; a multi-search strategy is utilized to balance exploration and exploitation. A search strategy selects convergent solutions from offspring and two archives as parents to enhance the convergence; the goal of another search strategy is to balance exploration and exploitation. The TwoArchM is compared experimentally with several state-of-the-art algorithms on the CEC2018 many-objective benchmark functions with up to 15 objectives and the experimental results verify the competitiveness and effectiveness of the proposed algorithm.

INDEX TERMS Many-objective optimization, two archives, multi-search strategy, evolutionary algorithm.

I. INTRODUCTION

Multi-objective optimization problems (MOPs) are complex and occur in many real-word applications. They usually include two or three objectives which often are inter-conflicting. A minimized MOP can be described as follows [1]:

$$\begin{cases} \min\ F(x) = (f_1(x), f_2(x), \cdots, f_m(x)) \\ \text{s.t. } x \epsilon \Omega \end{cases} \tag{1}$$

where $m = 2$ or $m = 3$, Ω is an n-dimensional decision vector space (i.e., $\Omega \sqsubseteq R^n$), $x = (x_1, \cdots, x_n) \in \Omega$ is a n-dimensional decision variable. In MOPs, the quality of the optimal solution is evaluated by the tradeoffs between multiple conflicting objectives. For two solutions $x, z \in \Omega$, if each $f_i(x) \leq f_i(z)$ and $\|F(x) - F(z)\|_2 \neq 0$, x dominates z (or denoted $x \prec z$). A solution vector x is a Pareto optimal solution or non-dominated solution if there does not exist another solution that dominates it. The set of Pareto optimal solutions (PS) is constituted by all Pareto optimal solutions. The image set of all the Pareto-optimal solutions in the objective space is called Pareto optimal front (PF).

For MOPs, the multi-objective evolutionary algorithms (MOEAs) [2]–[7] based on Pareto dominance have been the state-of-the-art strategies for 2 or 3 objectives. The most significant feature of MOEAs is that they all use population-based search to drive individuals towards different sections of the true PF simultaneously in a single run. In the real word, may MOPs [8], [9] contain more than three objectives, commonly referred to as many-objective optimization problems (MaOPs) [10]. Unlike MOPs, MaOPs often need more emphasis on the convergence ability of MOEAs [11]. Unfortunately, for MaOPs, many MOEAs which are based on Pareto-dominance [2]–[7] may encounter severe loss of selection pressure toward the true Pareto-optimal front, because the number of non-dominated solutions increases exponentially with the number of objectives in the early evolution. How to effectively solve MaOPs has caused the attention of many scholars and has become the hot topic.

Currently, a number of improvement strategies have been developed well in many-objective evolutionary

The associate editor coordinating the review of this manuscript and approving it for publication was Geng-Ming Jiang.

algorithms (MaOEAs) to solve MaOPs [43] and these algorithms can be divided into three categories. The first category is that relaxed dominance-based approaches [12]–[14] aim to increase the probability of one solution being dominated by other solutions, and have been found to have good performances for MaOPs. These modified Pareto dominance designs relaxed the Pareto dominance relation to make one solution dominate others easily in a high-dimensional space. For these approaches, defining new relaxed dominance relations and their relaxation extent for different problems remains challenging.

The second approach is indicator-based evolutionary algorithms which use quality indicators (such as hypervolume [15]) to guide the search towards a PF. High search ability of indicator-based algorithms has been demonstrated in the literature [16]. The hypervolume estimation algorithm (HypE [17]) and the indicator-based evolutionary algorithm (IBEA [18]) are the two representatives in this category. However, these algorithms have high computation complexity [19].

Another approach for solving MaOPs is the decomposition -based method which decomposes a MaOP into a set of tasks and then solves them collaboratively, such as MOEA/D [7] and its variants [20]–[26]. Apart from MOEA/D, some other methods based on decomposition [27]–[29], [44]–[49] also are proposed. However, these approaches are ineffective to tackle highly irregular PFs.

Furthermore, combining existing studies may be a novel avenue to solve MaOPs. As a representative, the two-archive algorithm [30] first uses two independent archives to keep the promising solutions. The convergence archive (CA) pursues the solutions to PF according to the dominance relationship. The diversity archive (DA) seeks solutions which local in sparse areas. When the total size of two archives overflows, the truncation is only operated on DA and crowding solutions are deleted. However, the number of solutions in CA and DA may increase significantly when tackling MaOPs. This is because the number of non-dominated solutions increases exponentially with the number of objectives in the early evolution. The improved version of two-archive algorithm (ITAA) [31] sets a threshold for the size of CA. Then, penalty-based boundary intersection (PBI) function is used to assign a fitness value for each solution in CA. The solutions with smaller fitness values will be deleted from CA. The shift density estimation-based truncation is used to eliminate solutions from DA when the total size still overflows. However, since the size of DA is flexible, the final output of ITAA produces inadequate diversity. To address this problem, two-archive 2 method (TwoArch2) [32] is proposed. The updated strategy of IBEA [18] is used to update CA. When the population overflows, DA iteratively selects boundary solutions which are solutions with extreme objective values. Finally, DA is output as the final output of TwoArch2. Cai and Qu [33] use two updated strategies base on the aggregation-based framework to update CA and DA. However, for MaOPs with partial PFs whose projections do not fully cover the unit hyperplane, the size of DA may be smaller than the given size, this is because some the weight vectors may have the same optimal solution.

Convergence and diversity are two main but conflicting goals. The key question of an effective MaOEA is how to design a mechanism for balancing convergence and diversity. Two-archive methods can well balance convergence and diversity. We aim to improve the convergence of DA by designing a multi-search strategy to produce good offspring. The major contributions of this paper can be summarized as follows.

1. The convergence and diversity is maintained separately.
2. A multi-search strategy is developed to balance exploration and exploitation. The convergent offspring are chosen to generate next offspring, which can improve the convergence of DA.
3. In DA, a new diversity management scheme depends on the product of two-norm and infinity-norm of two solutions is developed. This method maintains the diversity by making the distance and difference of any two solutions are as big as possible.

The rest of this paper is organized as follows: Section 2 displays the proposed algorithm in detail; while experimental results of the proposed algorithm and the related analysis are given in Section 3; finally, Section 4 provides the conclusions and proposes the future work.

II. THE PROPOSED ALGORITHM

In this paper, an improvement two-archive evolutionary algorithm based on multi-search strategy (TwoArchM) is developed to solve MaOPs. The major components of TwoArchM are that two updated strategies for two-archive are used to balance convergence and diversity, a multi-search strategy is proposed to enhance the convergence and balance exploration and exploitation.

A. BASIC IDEA

The main purpose of population is that maintains the diversity of the population, enhances the convergence of the population and provides good parents to generate excellent offspring. A MaOEA should output a population with good convergence and diversity. The convergence archive also needs to maintain diversity to balance exploration and exploitation. However, the convergence and diversity are very difficult to balance for MaOPs. Some convergence solutions which can't be kept in the convergence archive or the diversity archive are used to produce offspring. The main motivation of this paper is that enhance the quality of offspring by using theconvergence solutions.

B. CONVERGENCE ARCHIVE

In this subsection, the details of the updated strategy for the CA are present. Solutions in CA are firstly classified to maintain the diversity. The aggregation function value for

updating the solutions in CA is used to improve the convergence. For a given set of weight vectors $(\gamma^1, \gamma^2, \cdots, \gamma^N)$ and the convergence archive CA (where N is the number of the weight vectors), the solutions in CA are classified by the following equations:

$$\begin{aligned} CA^i &= \{x | x \in CA, \Delta\left(F(x), \gamma^i\right) \\ &= \max_{1 \le j \le N} \{\Delta(F(x), \gamma^j)\}\} \\ \Delta\left(F(x), \gamma^i\right) &= \frac{\gamma^i * (F(x) - Z)^T}{\left\|\gamma^i\right\| * \|F(x) - Z)\|}, \quad i = 1, \cdots N \end{aligned} \tag{2}$$

where $Z = (Z_1, \cdots Z_m)$ is a reference point with $Z_i = min\{f_i(x) | x \in \Omega\}$; $\Delta\left(F(x), \gamma^i\right)$ is the cosine of the angle between γ^i and $F(x) - Z$. After the solutions in CA are classified, a modified version of the Tchebycheff function is adopted to delete some solutions. Specifically, the function for the weight vector γ^i can be defined as follows:

$$\underset{x \in \Omega}{\text{minimize}}\ g^{TE}\left(x \mid \gamma^i, Z\right) = \max_{1 \le j \le m} \left\{\left|f_j(x) - Z_j\right| / \gamma_j^i\right\} \tag{3}$$

The optimal solution x_i^* of (3) must be the Pareto optimal solution of (1). If CA^i is not empty, the solution in CA^i with the minimum value $g^{TE}\left(x \mid \gamma^i, Z\right)$ will be kept and other solutions in CA^i will be deleted. Each CA^i has more than one solution.

C. DIVERSITY ARCHIVE

The diversity maintenance plays an important role in solving the MaOPs. For MaOPs, to maintain the diversity, the distance and difference of any two solutions should be as big as possible. Based on this idea, the distance value of any solution x of the population POP is calculated as following equation:

$$\begin{aligned} d(x) = \min \big\{ &\|F(x) - F(y)\|_2 * \|F(x) - F(y)\|_\infty | y \\ &\in POP \cap y \ne x \big\} \end{aligned} \tag{4}$$

In the diversity archive, some solutions with larger distance values are selected as the next diversity archive. $\|F(x) - F(y)\|_2$ and $\|F(x) - F(y)\|_\infty$ indicate the distance and difference of two solutions, respectively. The relation of $\|F(x) - F(y)\|_2$ and $\|F(x) - F(y)\|_\infty$ is $\|F(x) - F(y)\|_\infty * \sqrt{\frac{m}{m-1}} \le \|F(x) - F(y)\|_2 \le \|F(x) - F(y)\|_\infty * \sqrt{2}$. Thus, the two-norm of two solutions is bigger, these two solutions are kept with bigger probability.

D. MULTI-SEARCH STRATEGY

A good strategy should help algorithms to balance exploration and exploitation. In this work, a multi-search strategy is designed to achieve the goal. The multi-search strategy contains two search strategies. The first search strategy selects some good convergent solutions from offspring and CA (or DA) as the parents. For convenience, O indicates the set of offspring. To select good convergent solutions, a threshold value t is determined by the following formula:

$$t = \frac{1}{|CA|} \sum_{x \in CA} \|F(x) - Z\| \tag{5}$$

Then some solutions are selected by the following formula:

$$pp = \{x | \|F(x) - Z\| \le t, x \in CA \cup O\} \tag{6}$$

We use the Eq. (4) to calculate the crowding degree of each solution in pp. Then, we use the roulette wheel selection to some better solutions from pp as the parents.

The first search strategy uses Eq. (6) to improve the convergence. To reduce the probability of falling into local optimum and enhance exploration, we select sparse solutions in pp to produce offspring. The first search strategy emphasizes local search and takes into account global search.

The second search strategy based on decomposition is designed. The details are as follows. We find the T closet weight vectors to each weight vector according to the Euclidean distances of any two weight vectors. For each $i = 1, \cdots N$, set $B(i) = \{i_1, \cdots, i_T\}$ and $BB(i) = \{x^{i_1}, \cdots x^{i_T}$ where N is the number of weight vectors, $\lambda^{i_1}, \cdots \lambda^{i_T}$ are the T closet weight vectors to λ^i, x^{i_1} is a solution in CA^{i_1}. If CA^{i_1} is empty, we delete x^{i_1} from $BB(i)$; if CA^i is empty, we make $BB(i)$ also empty. If CA^i is not empty, one or more solutions are chosen randomly from $BB(i)$ as parents. This search strategy selects solutions which are neighbors from CA as parents, which may enhance the local search. To enhance the global search, this search strategy also chooses solutions from CA at random to generate offspring.

In the paper [4], a multi-search strategy is proposed improving the search efficiency. This multi-search strategy contains three search strategies. One search strategy selects sparse solutions from obtained non-dominated solutions as parents; another search strategy selects some a dominated solution and its adjacent non-dominated solution as parents to help the dominated solution to become is a non-dominated solution; the third search strategy selects non-dominated solutions and their adjacent solutions as parents to enhance the local search. The main difference of the multi-search strategy [4] and this proposed multi-search strategy is that this proposed multi-search strategy selects some good convergent solutions from offspring and CA (or DA) according to Eqs. (5-6) as the parents, the multi-search strategy [4] selects convergent solutions (the non-dominated solutions) only from the population as the parent.

E. STEPS OF THE PROPOSED ALGORITHM

Based on all above, an improvement two-archive evolutionary algorithm based on multi-search strategy (TwoArchM) is devolved; the general framework of TwoArchM is as follows.

In TwoArchM, the polynomial mutation [40] operator and the simulated binary crossover (SBX [40]) are used as the genetic operations. TwoArchM uses two archives to balance the diversity and convergence. And, TwoArchM adopts a multi-search strategy to balance exploration and exploitation.

TwoArchM firstly calculate the threshold value t according to the convergence archive and Eq. (5). This threshold value is used to select convergent solutions as parents. For the convergence archive (CA), convergent solutions are selected from CA and its offspring according to the threshold value;

The framework of the algorithm TwoArchM

Input :
MaOP (1)
A stopping criterion
N : the number of weight vectors
T : the number of weight vectors in the neighborhood of each weight vector, $0 < T < N$
$\lambda^1, \lambda^2, \cdots, \lambda^N$: a set of N uniformly distributed weight vectors
Output: Diversity archive DA
Initialization: Generate an initial population $\{x^1, x^2, \cdots x^N$ randomly; determine $Z = (z_1, \cdots, z_m)$ by a problem-specific method; determine the convergence archive CA by Eqs. (2-3); determine $B(i) = \{i_1, \cdots, i_T\}, (i = 1, \cdots, N)$ and $BB(i)$; set $O_1 = \{x^1, x^2, \cdots x^N\}$, $O_1 = O_2$.
While *the stopping criterion is not met* **do**
Calculate the threshold value t by Eq. (5).
Determine the sets $pp_1 = \{x| \|F(x) - Z\| \leq t, x \in CA \cup O_1\}$ and $pp_2 = \{x| \|F(x) - Z\| \leq t, x \in DA \cup O_2\}$; set $O_1 = \emptyset$ and $O_2 = \emptyset$.
Select N better solutions from pp_1 by Eq. (4) and the roulette wheel selection to generate offspring which are put into O_1.
Select N better solutions from pp_2 by Eq. (4) and the roulette wheel selection to generate offspring which are put into O_2.
Set $O = \emptyset$.
For $i = 1, \cdots, N$, **do**
if $CA^i \neq \emptyset$ **then**
if rand $<$ J **then**
$E = BB(i)$
else
$E = CA$
end if
Randomly select a solution x from E; use x and $x^i \in CA^i$ to generate offspring x^{new} by genetic operations, and set $O = O \cup x^{new}$.
end if
End for
Set $O_1 = O \cup O_1 \cup CA, O_2 = O \cup O_2 \cup DA, O = O_2 \cup O_1$.
Use O **to Update** Z: For $j = 1, \cdots, m$, if $z_j < \min\{f_j(x^{new}) | x^{new} \in O\}$, then set $z_j = \min\{f_j(x^{new}) | x^{new} \in O\}$.
Use the updated strategy of Section 2.2 to update CA from O_1.
Use the updated strategy of Section 2.3 to update DA from O_2.
End while

then, N better solutions are selected from these convergent solutions as parents according to Eq. (4) which is used to measure the crowding degree of each solution of these convergent solutions; finally, these parents use the genetic operations to generate offspring. In the same way, the diversity archive (DA) produces offspring according to the threshold value. This search strategy selects convergent solutions to generate offspring, which can improve the convergence. This search strategy also uses the crowding degree (Eq. (4)) to maintain the diversity. Moreover, TwoArchM classifies CA into some classes by weight vectors and Eq. (2), and for each non-empty class, only one solution with the minimum aggregate function value; then, for the solution of each non-empty class, it and one of its neighbor solutions which is chosen at random are as parents to generate offspring. These offspring are also considered as the offspring of DA, which is to improve the quality of DA's offspring. The goal of this strategy is to balance diversity and convergence globally.

TwoArchM uses two updated strategies to update CA and DA. For CA, CA and its offspring are classified by Eq. (2), each non-empty class only keep one solution with the minimum aggregate function value, these kept solutions make up the next CA. For DA, non-dominated solutions are firstly found from DA and its offspring; then, some solutions are selected as the next DA according to the crowding degree (Eq. (4)).

In this algorithm TwoArchM, $5N$ solutions(O_1, O_2, O and two archives), $B(i)(i = 1, \cdots, N)$, $CA^i(i = 1, \cdots, N)$ and N weight vectors need to be stored, thus the space complexity of TwoArchM is $O(5N * n) + O(N * T) + O(N * N) + O(N * m) = O(N^2)$ (in this paper, $n, T, m < N$). So, the space complexity of TwoArchM is $O(N^2)$. The major computation in this algorithm involved is in the determine the sets (pp_1 and pp_2) and the update step of TwoArchM. To determine the sets (pp_1 and pp_2), $O(3N * n)$ basic operations (i.e., $+$, $-$, $\times$, $\div$ and comparison) are needed. To update CA, $O(3N * N)$ basic operations are needed to classify $3N$ solutions, and at most $O(N * N)$ basic operations to determine the best solution of each class. To update DA, $O(3N * 6 * N)$ basic operations are needed to calcite the values $d(x)$ of $3N$ solutions, and at most $O(3N * lg3 * N)$ basic operations to choose N best solutions with larger distance values. Totally, the computation complexity of the algorithm is $O(3N * n) + O(3N * N) + O(N * N) + O(3N * 6 * N) + O(3N * lg3 * N) = O(N^2)$.

III. COMPUTATIONAL STUDIES AND RESULTS

A. EXPERIMENTAL SETTINGS

This section presents our experimental study for the performance of the TwoArchM by with five art-of-the-state algorithms, i.e., NSGAIII [27], MOEA/DD [36], KnEA [37], RVEA [38], TwoArch2 [32], on a set of the benchmark suite from CEC2018 MaOP competition [35]. There are fifteen many-objective benchmark functions (MaF) with box constraints in the solution space in this benchmark suite. For each test problem, the number of objectives can be set to 5, 10 and 15. NSGAIII [27] employs a set of preference directions to guide the evolution of population towards different parts of the PF, promoting the diversity of population. MOEA/DD [36] combines dominance and

TABLE 1. The mean and standard deviation values of IGD obtained by TwoArchM, NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2. "+" means that TwoArchM outperforms its competitor algorithm, "−" means that TwoArchM is worse than its competitor algorithm, and "=" means that the competitor algorithm has the same performance as TwoArchM.

Problems	TwoArchM	NSGAIII	MOEA/DD	KnEA	RVEA	TwoArch2
F1-5	**1.1333e-1 (9.08e-4)**	2.0756e-1 (5.91e-3)(+)	2.9017e-1 (2.80e-2)(+)	1.2371e-1 (1.83e-3)(+)	4.0018e-1 (1.75e-1)(+)	1.2324e-1 (1.23e-3) (+)
F2-5	**9.2401e-2 (1.35e-3)**	1.2910e-1 (4.25e-4)(+)	1.3664e-1 (3.38e-3)(+)	1.2766e-1 (3.92e-3)(+)	1.2787e-1 (1.53e-3)(+)	9.9752e-2 (1.52e-3) (+)
F3-5	**8.2392e-3 (4.27e-3)**	9.6731e-2 (1.80e-3)(+)	1.1809e-1 (1.43e-3)(+)	1.5308e-1 (6.73e-2)(+)	8.2950e-2 (7.68e-3)(+)	8.7646e-2 (1.96e-2) (+)
F4-5	**1.8821e+0 (5.60e-2)**	3.0035e+0 (2.56e-1)(+)	7.6920e+0 (3.22e-1)(+)	2.9191e+0 (3.14e-1)(+)	4.9814e+0 (1.38e+0)(+)	1.9017e+0 (1.18e-2) (+)
F5-5	**1.7764e+0 (3.14e-2)**	2.2085e+0 (5.09e-4)(+)	6.3810e+0 (3.09e-1)(+)	2.4229e+0 (4.43e-2)(+)	2.3478e+0 (1.73e-1)(+)	1.9100e+0 (1.95e-2) (+)
F6-5	**3.3807e-3 (1.63e-4)**	9.1571e-2 (2.57e-2)(+)	7.9096e-2 (4.66e-4)(+)	6.9374e-3 (1.65e-3)(+)	9.3646e-2 (2.29e-2)(+)	6.8065e-3 (6.29e-5) (+)
F7-5	**2.8687e-1 (8.65e-3)**	3.2498e-1 (1.27e-2)(+)	3.0005e+0 (1.37e-6)(+)	2.9897e-1 (3.51e-3)(+)	4.4937e-1 (3.54e-3)(+)	2.8792e-1 (9.17e-3) (+)
F8-5	**1.1223e-1 (2.05e-2)**	2.1374e-1 (4.99e-3)(+)	3.0704e-1 (1.95e-2)(+)	2.6409e-1 (2.90e-2)(+)	4.4649e-1 (7.36e-2)(+)	1.1291e-1 (1.72e-3) (+)
F9-5	**1.0878e-1 (9.41e-3)**	2.6409e-1 (5.47e-3)(+)	2.4667e-1 (3.57e-3)(+)	7.8574e-1 (1.29e-1)(+)	3.8989e-1 (8.60e-2)(+)	1.2218e-1 (1.48e-2) (+)
F10-5	8.2108e-1 (1.34e-1)	4.6073e-1 (1.46e-2)(-)	7.7469e-1 (1.54e-1)(-)	5.2086e-1 (3.94e-2)(-)	4.4844e-1 (1.51e-2)(-)	**4.3020e-1 (2.99e-3) (-)**
F11-5	**6.5169e-1 (6.34e-2)**	8.2588e-1 (3.81e-3)(-)	4.5150e+0 (6.44e-2)(+)	6.5471e-1 (1.16e-1)(=)	1.7788e+0 (5.67e-1)(+)	6.5373e-1 (4.77e-2) (=)
F12-5	**1.0486e+0 (1.78e-2)**	1.1169e+0 (5.17e-3)(+)	1.2902e+0 (2.17e-2)(+)	1.1662e+0 (9.79e-3)(+)	1.1274e+0 (1.00e-2)(+)	1.0935e+0 (1.16e-2) (=)
F13-5	**1.1117e-1 (1.83e-2)**	2.3375e-1 (7.27e-3)(+)	2.6117e-1 (3.68e-2)(+)	2.2829e-1 (7.28e-3)(+)	6.6361e-1 (1.41e-1)(+)	1.5088e-1 (3.23e-3) (+)
F14-5	**3.3928e-1 (5.29e-1)**	7.0040e-1 (2.44e-1)(+)	3.8364e-1 (5.18e-2)(+)	4.8626e-1 (5.30e-2)(+)	8.3614e-1 (1.70e-1)(+)	1.5913e+0 (1.61e-2) (+)
F15-5	9.8853e-1 (5.28e-2)	1.1500e+0 (9.48e-3)(+)	**5.7556e-1 (2.44e-2)(-)**	3.7846e+0 (2.20e+0)(+)	6.0225e-1 (4.49e-2)(-)	1.1227e+0 (8.99e-3) (+)
F1-10	**2.2223e-1 (2.89e-3)**	2.7014e-1 (5.94e-3)(=)	3.9192e-1 (2.21e-2)(+)	2.3097e-1 (5.20e-3)(=)	6.5904e-1 (7.38e-2)(+)	2.3648e-1 (1.07e-3) (=)
F2-10	**1.6360e-1 (3.62e-3)**	2.0240e-1 (1.56e-2)(+)	2.2270e-1 (4.80e-3)(+)	1.6817e-1 (1.37e-2)(+)	2.6052e-1 (7.97e-2)(+)	1.6567e-1 (6.03e-4) (=)
F3-10	**1.1391e-1 (5.16e-3)**	2.7372e+2 (1.68e+2)(+)	1.1529e-1 (2.30e-3)(=)	2.3297e+6 (1.86e+6)(+)	1.9320e-1 (2.20e-1)(+)	2.3023e-1 (2.25e-2) (+)
F4-10	6.0732e+1 (3.56e+0)	8.8824e+1 (2.60e+0)(+)	4.0451e+2 (7.49e+0)(+)	8.0210e+1 (8.89e+0)(+)	2.0455e+2 (4.12e+1)(+)	**5.1828e+1 (3.60e+0) (-)**

TABLE 1. *(Continued.)* **The mean and standard deviation values of IGD obtained by TwoArchM, NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2. "+" means that TwoArchM outperforms its competitor algorithm, "−" means that TwoArchM is worse than its competitor algorithm, and "=" means that the competitor algorithm has the same performance as TwoArchM.**

F5-10	4.8290e+1 (1.22e+0)	8.9061e+1 (2.65e-1)(+)	2.9689e+2 (1.23e+0)(+)	7.3017e+1 (5.21e-1)(+)	9.9679e+1 (6.17e+0)(+)	**4.8217e+1 (7.40e-1) (=)**
F6-10	**3.0904e-3 (2.32e-4)**	2.9435e-1 (7.45e-2)(+)	1.1321e-1 (3.54e-3)(+)	1.3068e+1 (4.24e+0)(+)	1.2161e-1 (1.75e-2)(+)	3.7318e-1 (5.22e-1) (+)
F7-10	9.5205e-1 (1.93e-2)	1.0316e+0 (9.09e-2)(=)	2.9641e+0 (6.69e-2)(+)	**8.6974e-1 (9.59e-3)(-)**	2.9783e+0 (3.07e-1)(+)	9.4187e-1 (2.13e-1) (=)
F8-10	1.4154e-1 (7.03e-3)	3.3710e-1 (1.24e-2)(+)	9.0462e-1 (7.04e-3)(+)	1.6873e-1 (3.32e-2)(+)	7.7009e-1 (1.13e-1)(+)	**1.1729e-1 (5.18e-4) (-)**
F9-10	**1.8987e-1 (7.44e-2)**	6.2161e-1 (1.18e-1)(+)	4.4369e-1 (4.18e-3)(+)	4.7242e+1 (4.44e+1)(+)	7.8129e-1 (2.29e-1)(+)	1.0350e+0 (6.52e-2) (+)
F10-10	1.4618e+0 (1.50e-1)	1.0680e+0 (2.53e-2)(-)	1.9032e+0 (1.25e-1)(+)	1.1648e+0 (7.41e-2)(-)	1.2659e+0 (6.46e-2)(+)	**9.5737e-1 (9.05e-3) (-)**
F11-10	**1.4304e+0 (7.68e-1)**	4.3444e+0 (9.03e-2)(+)	1.5214e+0 (2.89e-2)(+)	2.7405e+0 (5.88e-1)(+)	8.8117e+0 (1.82e+0)(+)	2.7661e+0 (6.91e-1) (+)
F12-10	**4.1411e+0 (3.27e-2)**	4.5939e+0 (8.38e-3)(+)	6.6451e+0 (1.17e-1)(+)	4.5421e+0 (1.12e-2)(+)	4.5082e+0 (6.74e-2)(+)	4.2537e+0 (7.27e-2) (+)
F13-10	**1.4190e-1 (2.14e-2)**	2.1097e-1 (1.32e-2)(+)	3.0229e-1 (1.74e-2)(+)	1.5108e-1 (1.41e-2)(+)	9.5107e-1 (2.16e-1)(+)	1.5807e-1 (8.16e-3) (+)
F14-10	9.2641e-1 (3.40e+0)	1.6855e+0 (3.11e-1)(+)	**5.2303e-1 (2.26e-2)(-)**	4.5108e+1 (2.78e+1)(+)	6.0231e-1 (2.88e-2)(-)	1.1874e+0 (1.61e-1)
F15-10	2.4387e+0 (1.18e+0)	1.6332e+0 (2.60e-1)(-)	9.8570e-1 (8.70e-3)(-)	6.6282e+0 (3.63e+0)(+)	**9.5090e-1 (6.06e-2)(-)**	2.5201e+0 (6.47e-1) (=)
F1-15	**2.6678e-1 (3.12e-3)**	3.1410e-1 (1.20e-4)(+)	5.2549e-1 (5.74e-3)(+)	2.7435e-1 (1.72e-3)(=)	7.0604e-1 (7.76e-2)(+)	2.8129e-1 (2.14e-3) (+)
F2-15	**1.4515e-1 (8.32e-3)**	2.0260e-1 (4.33e-4)(+)	3.9623e-1 (1.46e-3)(+)	1.8233e-1 (1.41e-3)(+)	5.3873e-1 (2.32e-1)(+)	1.6394e-1 (1.01e-3) (+)
F3-15	1.2521e-1 (1.28e-3)	1.8318e-1 (4.09e-2)(+)	1.1059e-1 (1.87e-3)(-)	4.9121e+9 (5.03e+9)(+)	**9.5038e-2 (5.67e-3)(-)**	3.4021e-1 (4.90e-2) (+)
F4-15	1.9040e+3 (8.40e+1)	4.1050e+3 (1.99e+2)(+)	1.5425e+4 (1.10e+3)(+)	1.4869e+4 (2.14e+4)(+)	8.4500e+3 (1.96e+3)(+)	**1.4787e+03 (8.89e+1) (-)**
F5-15	1.3936e+3 (8.84e+1)	2.6072e+3 (1.33e+1)(+)	7.3113e+3 (8.35e+0)(+)	1.5267e+3 (6.25e+1)(+)	3.3958e+3 (7.04e+2)(+)	**1.0695e+3 (1.28e+2) (-)**
F6-15	**3.5905e-3 (9.36e-4)**	3.2105e-1 (3.01e-2)(+)	1.3327e-1 (1.88e-3)(+)	2.0123e+1 (3.29e+0)(+)	2.4134e-1 (1.81e-1)(+)	7.4209e-1 (1.18e-6) (+)
F7-15	1.5777e+0 (5.36e-2)	4.3214e+0 (4.75e-1)(+)	3.4471e+0 (2.51e-2)(+)	2.4327e+0 (1.54e-1)(+)	4.1309e+0 (6.74e-1)(+)	**1.5094e+0 (8.22e-2) (=)**
F8-15	1.9105e-1 (1.90e-2)	4.1812e-1 (4.99e-3)(+)	1.3421e+0 (3.91e-3)(+)	1.3876e-1 (1.13e-3)(-)	1.2975e+0 (3.08e-1)(+)	**1.1051e-1 (1.56e-3) (-)**
F9-15	1.5590e-1 (8.87e-4)	1.4357e+0 (1.51e+0)(+)	9.6572e-1 (2.15e-3)(+)	9.5128e-1 (6.43e-1)(+)	1.2312e+0 (2.31e-1)(+)	**1.0715e-1 (2.40e-3) (-)**

TABLE 1. *(Continued.)* **The mean and standard deviation values of IGD obtained by TwoArchM, NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2. "+" means that TwoArchM outperforms its competitor algorithm, "−" means that TwoArchM is worse than its competitor algorithm, and "=" means that the competitor algorithm has the same performance as TwoArchM.**

F10-15	2.0834e+0 (1.43e-1)	1.5352e+0 (6.98e-3)(-)	2.3803e+0 (2.88e-2)(-)	1.7252e+0 (5.88e-2)(-)	1.7982e+0 (7.57e-2)(-)	**1.4136e+0 (2.23e-2) (-)**
F11-15	1.2023e+0 (1.18e+0)	6.8030e+0 (4.07e+0)(+)	2.5675e+1 (2.77e-1)(+)	4.6837e+0 (4.21e-1)(+)	1.9912e+1 (2.71e+0)(+)	**3.5574e-1 (3.31e-1) (-)**
F12-15	7.3710e+0 (3.54e-2)	8.0325e+0 (5.71e-2)(+)	8.5474e+0 (2.94e-1)(+)	**6.5516e+0 (1.13e-1)(-)**	7.0894e+0 (2.10e-1)(-)	7.7344e+0 (5.81e-2) (+)
F13-15	1.7086e-1 (9.96e-3)	2.8986e-1 (4.22e-2)(+)	2.8823e-1 (3.23e-2)(+)	**1.1691e-1 (1.15e-2)(-)**	1.0959e+0 (3.59e-1)(+)	1.4698e-1 (6.84e-3) (-)
F14-15	1.2505e+0 (4.37e-1)	1.1236e+0 (1.29e-1)(+)	**5.1146e-1 (5.54e-3)(-)**	1.4711e+1 (6.85e+0)(+)	8.1407e-1 (1.53e-1)(-)	1.1334e+0 (4.47e-2) (+)
F15-15	2.7527e+0 (2.14e-1)	3.7230e+0 (0.00e+0)(+)	**1.1226e+0 (1.92e-2)(-)**	1.1760e+2 (7.36e+0)(-)	1.1599e+0 (2.94e-2)(-)	1.2555e+0 (1.82e-1) (-)
+/=/-		38/2/5	37/1/7	34/3/8	35/0/9	26/8/11

decomposition-based approaches, which exploits the merits of both dominance-and decomposition-based approaches to balance the convergence and diversity of the evolutionary process. KnEA [37] is a knee point-driven EA to enhance the convergence performance in many-objective optimization. RVEA [38] uses the reference vectors to decompose the original multiobjective optimization problem into a number of single-objective subproblems and elucidate user preferences to target a preferred subset of the whole Pareto front. These algorithms can be grouped into three classes: 1) the reference points/weight vectors based algorithms (MOEAD/DD, NSGAIII and RVEA); 2) knee points based algorithm (KnEA); 3) Two-archive based algorithm (TwoArch2).

For these five compared algorithms, all data of each algorithm given in this section are averaged over 20 independent runs for each test case on PlatEMO [39]. In TwoArchM, Polynomial mutation [40] operator and simulated binary crossover (SBX [40]) are used; distribution index is 20 and crossover probability is 1 in the SBX operator; distribution index is 20 and mutation probability is 0.1 in mutation operator; the size of neighborhood list T is set to 0.1N; J is set to 0.9. The settings of the experimental studies in this paper are identical to the standard for CEC2018 MaOP competition, which can be found in [35], together with the details of the benchmark functions. The algorithms are implemented by using the MATLAB language on a PC with Intel Xeon CPU E3-1226 (3.30 GHz for a single core and the Windows 10 operating system).

B. PERFORMANCE METRICS

In this paper, the inverted generational distance (IGD) [41] is used to quantitative measurement the performances of algorithms. For an algorithm, a smaller IGD value means the better quality of the objective vectors of obtained solutions for approximating the PF. The benefits of IGD lie in its computational efficiency and generality for measuring both convergence and diversity of solutions. IGD requires a set of reference Pareto optimal solutions. Roughly 10000 points uniformly sampled on the Pareto fronts are used in the calculation of IGD for each test problems. Wilcoxon Rank-Sum test [42] is used in the sense of statistics to compare the mean IGD of the compared algorithms. It tests whether the performance of TwoArchM on each test problem is better ("+"), same ("="), or worse ("−") than/as that of the compared algorithms at a significance level of 0.05 by a two-tailed test.

C. COMPARATIVE STUDIES

The computational results of involved algorithms over 5-, 10, 15-objective test benchmarks are reported in Table I. In the table, the mean and standard deviation values in term of IGD obtained by the six MaOEAs over 20 independent runs are reported. The difference significance between TwoArchM and compared algorithms is evaluated by Wilcoxons rank sum test. For each test problem, the result of with the best performance is marked in bold. F1-k represents that the number of objectives adopted in F1 is k.

As shown in Table I, on all forty-five test problems under consideration, TwoArchM performs statistically better than the other compared algorithms on twenty-eight test problems; TwoArchM obtains the best performance on all the instances of F1, F2 and F6; NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2 perform statistically better than TwoArchM on five, seven, eight, nine and eleven problems, respectively; TwoArchM outperforms NSGAIII,

TABLE 2. The mean and standard deviation values of IGD obtained by TwoArchM, TwoArchM1, and TwoArchM2. "+" means that TwoArchM outperforms its competitor algorithm, "−" means that TwoArchM is worse than its competitor algorithm, and "=" means that the competitor algorithm has the same performance as TwoArchM.

Problems	TwoArchM	TwoArchM1	TwoArchM2
F1-5	**1.1333e-1(9.08e-4)**	1.1662e-1(5.495-3)(+)	1.1440e-1(1.41e-4) (+)
F2-5	**9.2401e-2(1.35e-3)**	9.8105e-2(7.10e-3) (+)	9.4445e-2(4.21e-3) (+)
F3-5	**8.2392e-3(4.27e-3)**	7.0058e-2(2.51e-2) (+)	6.0627e+1(9.15e+0) (+)
F4-5	**1.8821e+0(5.60e-2)**	2.1928e+0(5.77e-2) (+)	9.0245e+0(7.91e-2) (+)
F5-5	**1.7764e+0(3.14e-2)**	1.7902e+0(3.71e-2) (=)	1.8042e+0(9.59e-2) (=)
F6-5	3.3807e-3(1.63e-4)	**3.3560e-3(7.78e-4) (=)**	8.8425e-3(6.55e-4) (+)
F7-5	**2.8687e-1(8.65e-3)**	3.0118e-1(5.49e-2) (=)	3.5229e-1(3.57e-3) (+)
F8-5	1.1223e-1(2.05e-2)	**9.9672e-2(7.11e-2) (=)**	2.8944e-1(8.49e-2) (+)
F9-5	**1.0878e-1(9.41e-3)**	1.1491e-1(6.89e-2) (=)	2.4121e-1(9.33e-2) (+)
F10-5	**8.2108e-1(1.34e-1)**	1.0629e+0(7.24e-1) (+)	1.4593e+0(6.78e-1) (+)
F11-5	**6.5169e-1(6.34e-2)**	7.3796e-1(5.57e-2) (+)	1.2512e+0(7.57e-1) (+)
F12-5	1.0486e+0(1.78e-2)	**1.0437e+0(9.74e-1) (=)**	1.0460e+0(7.43e-1) (=)
F13-5	**1.1117e-1(1.83e-2)**	2.0399e-1(3.91e-2) (+)	2.0285e-1(3.92e-2) (+)
F14-5	**3.3928e-1(5.29e-1)**	8.8471e-1(5.91e-1) (+)	9.1865e-1(6.55e-1) (+)
F15-5	**9.8853e-1(5.28e-2)**	1.0192e+0(7.28e-2) (+)	9.9046e-1(1.71e-2) (=)
F1-10	**2.2223e-1(2.89e-3)**	2.5462e-1(5.52e-2) (+)	2.4112e-1(7.06e-3) (+)
F2-10	**1.6360e-1(3.62e-3)**	1.8149e-1(7.00e-3) (+)	2.9871e-1(3.18e-3) (+)
F3-10	**1.1391e-1(5.16e-3)**	4.8890e-1(5.88e-2) (+)	1.0057e+0(2.76e-2) (+)
F4-10	**6.0732e+1(3.56e+0)**	6.4629e+1(9.77e-1) (+)	7.2830e+1(4.61e-1) (+)
F5-10	**4.8290e+1(1.22e+0)**	5.5235e+1(6.80e+0) (+)	4.9092e+1(9.71e+0) (=)
F6-10	**3.0904e-3(2.32e-4)**	**2.9510e-3(5.73e-4) (=)**	5.4808e-3(8.23e-4) (+)
F7-10	**9.5205e-1(1.93e-2)**	1.0251e+0(8.42e-1) (+)	1.0535e+0(6.94e-1) (+)
F8-10	1.4154e-1(7.03e-3)	**1.3691e-1(3.26e-2) (=)**	5.9655e-1(3.17e-2) (+)
F9-10	**1.8987e-1(7.44e-2)**	4.5590e+0(1.05e-1) (+)	2.0744e+1(9.50e-2) (+)
F10-10	1.4618e+0(1.50e-1)	1.7662e+0(8.89e-1) (+)	2.9023e+0(3.44e-1) (+)
F11-10	**1.4304e+0(7.68e-1)**	3.1610e+0(5.92e-1) (+)	1.1167e+1(4.38e-1) (+)
F12-10	**4.1411e+0(3.27e-2)**	4.3092e+0(8.68e-2) (+)	5.0180e+0(3.81e-2) (+)
F13-10	**1.4190e-1(2.14e-2)**	1.4970e-1(3.73e-2) (+)	1.8457e-1(7.65e-2) (+)
F14-10	**9.2641e-1(3.40e+0)**	3.3266e+0(6.64e-2) (+)	9.7009e-1(7.95e-2) (+)
F15-10	**2.4387e+0(1.18e+0)**	3.2861e+0(5.52e-1) (+)	1.9591e+0(1.86e-1) (+)
F1-15	**2.6678e-1(3.12e-3)**	4.2043e-1(3.56e-2) (+)	2.6866e-1(4.89e-2) (=)
F2-15	**1.4515e-1(8.32e-3)**	1.8729e-1(3.85e-2) (+)	3.8999e-1(4.45e-2) (+)
F3-15	**1.2521e-1(1.28e-3)**	1.3749e-1(1.74e-2) (+)	1.5829e-1(6.46e-2) (+)
F4-15	**1.9040e+3(8.40e+1)**	2.1537e+3(4.96e+1) (+)	2.6416e+3(7.09e+1) (+)
F5-15	**1.3936e+3(8.84e+1)**	1.5843e+3(6.64e+1) (+)	2.3004e+3(7.54e+1) (+)
F6-15	**3.5905e-3(9.36e-4)**	3.7705e-3(1.32e-2) (+)	1.0566e-2(2.76e-2) (+)
F7-15	**1.5777e+0(5.36e-2)**	2.1283e+0(8.79e-2) (+)	2.8911e+0(6.79e-2) (+)
F8-15	**1.9105e-1(1.90e-2)**	1.9778e-1(1.01e-2) (=)	6.0880e-1(6.55e-2) (+)
F9-15	1.5590e-1(8.87e-4)	**1.5588e-1(1.86e-1) (=)**	1.1689e+1(1.62e-1) (+)
F10-15	2.0834e+0(1.43e-1)	**2.0434e+0(4.23e-1) (=)**	3.2684e+0(1.19e-1) (+)
F11-15	**1.2023e+0(1.18e+0)**	4.0369e+0(1.14e+0) (+)	9.4041e+0(4.98e+0) (+)
F12-15	**7.3710e+0(3.54e-2)**	7.4099e+0(4.57e-2) (=)	8.4083e+0(9.59e-2) (+)
F13-15	**1.7086e-1(9.96e-3)**	2.3458e-1(2.26e-2) (+)	1.6233e-1(3.40e-2) (+)
F14-15	**1.2505e+0(4.37e-1)**	2.5540e+0(8.33e-2) (+)	1.6304e+0(5.14e-1) (+)
F15-15	**2.7527e+0(2.14e-1)**	2.9245e+0(6.14e-1) (+)	3.1465e+0(3.27e-1) (+)
+/=/-		33/12/0	40/5/0

MOEA/DD, KnEA, RVEA and TwoArch2 on thirty-eight, thirty-seven, thirty-four, thirty-five and twenty-six problems, respectively. These indicate that the quality of solutions obtained by TwoArchM is better than those obtained by NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2 on most problems.

For the 8 problems (F1, F2, F4, F5, F7, F8, F9 and F15) with partial PFs whose projections do not fully cover the unit hyperplane, the mean values of IGD obtained by TwoArchM are smaller than those obtained by NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2 on twenty-three, twenty-one, twenty-one, twenty-one and fifteen problems, which indicate that the proposed algorithm obtains the best overall performance in the form of IGD on most problems. For the problem F6 with degraded PF, the performances of TwoArchM outperform than NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2 on three problems. These comparison results demonstrating that the updated strategy of DA has brought versatility for the diverse PFs.

When dealing with problems with PF projection fully cover the unit hyperplane (F3, F10, F11, F12, F13 and F14), the mean values of IGD obtained by TwoArchM are smaller than those obtained by NSGAIII, MOEA/DD, KnEA, RVEA and TwoArch2 on fourteen, eleven, twelve, fourteen and fourteen problems, these imply that TwoArchM obtains the best overall performance in the form IGD on most problems and the multi-search strategy has good search performance.

D. ROLES OF MULTI-SEARCH STRATEGY

The roles of multi-search strategy are to improve the convergence and balance to exploration and exploitation. The first search strategy selects convergent solutions as parents to enhance the local search. The second search strategy is to balance the global search and local search. To identify this, TwoArchM compares with TwoArchM seriatim without the first search strategy or the second search strategy which denoted as TwoArchM1, or TwoArchM2 on forty-five problems. The parameters are the same as in Section 3.1.

Table II shows the mean and standard deviation values of IGD metric obtained by TwoArchM, TwoArchM1 and TwoArchM2 on these forty-five problems. It can be seen from Table II that, TwoArchM outperforms TwoArchM1 on thirty-three problems and TwoArchM is worse than TwoArchM1 on no problem, which hint that the first search strategy can help this algorithm to improve convergence on these problems; TwoArchM outperforms TwoArchM2 on forty problems and TwoArchM is worse than TwoArchM2 on no problem, which indicates that the second search strategy can help TwoArchM to improve the search efficiency. Moreover, for three MaOPs with many local PFs (F3, F4, F7), the values of IGD obtained by both of TwoArchM and TwoArchM1 are smaller than those obtained by TwoArchM2, which suggests that the second search strategy can help TwoArchM to avoid the remaining local optima. Comparison results illustrate that these two search strategies can help the algorithm to improve the performance.

IV. CONCLUSIONS

To address the problems in the existing MOEAs for MaOPs, an improvement two-archive evolutionary algorithm based on multi-search strategy (TwoArchM) is designed to obtain a set of solutions with good diversity and convergence. In this algorithm, a multi-search strategy is used to improve the convergence and balance the exploration and exploitation; two archives respectively to save convergent solutions and diversity solutions; two updated strategies are respectively proposed to update the diversity archive and convergence archive. Compare experimental results with the state-of-the-art algorithms also indicate that the proposed algorithm obtains competitive results on the CEC2018 many-objective benchmark functions.

COMPETING INTERESTS

The authors declare that they have no competing interests.

ACKNOWLEDGMENT

In this work, Ren Aihong provided professional writing services. Thanks for her help.

DATA AVAILABILITY

The [DATA TYPE] data used to support the findings of this study are available from the corresponding author upon request.

ETHICAL APPROVAL

This article does not contain any studies with human participants performed by any of the authors.

REFERENCES

[1] D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," Air Force Inst. Technol., Wright-Patterson AFB, OH, USA, 1999.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[3] B. Tang, Z. Zhu, H.-S. Shin, A. Tsourdos, and J. Luo, "A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm," *Inf. Sci.*, vol. 420, pp. 364–385, Dec. 2017.

[4] N. Dong and C. Dai, "An improvement decomposition-based multi-objective evolutionary algorithm using multi-search strategy," *Knowl.-Based Syst.*, vol. 163, pp. 572–580, Jan. 2019.

[5] R. Shang, L. Jiao, F. Liu, and W. Ma, "A novel immune clonal algorithm for MO problems," *IEEE Trans. Evol. Comput.*, vol. 16, no. 1, pp. 35–50, Feb. 2012.

[6] Z.-H. Zhan, J. Li, J. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.

[7] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[8] R. J. Lygoe, M. Cary, and P. J. Fleming, "A real-world application of a many-objective optimisation complexity reduction process," in *Proc. 7th Int. Conf. Evol. MultiCriterion Optim.*, Sheffield, U.K., 2013, pp. 641–655.

[9] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and Y. Jin, "Evolutionary many-objective optimization of hybrid electric vehicle control: From general optimization to preference articulation," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 2, pp. 97–111, Apr. 2017.

[10] E. J. Hughes, "Evolutionary many-objective optimisation: Many once or one many?" in *Proc. IEEE Congr. Evol. Comput.*, Edinburgh, U.K., Sep. 2005, pp. 222–227.

[11] Z. He and G. G. Yen, "Many-objective evolutionary algorithm: Objective space reduction and diversity improvement," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 145–160, Feb. 2016.
[12] K. Ikeda, H. Kita, and S. Kobayashi, "Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal?" in *Proc. Congr. Evol. Comput.*, vol. 2, 2001, pp. 957–962.
[13] C. Dai, Y. Wang, and M. Ye, "A new evolutionary algorithm based on contraction method for many-objective optimization problems," *Appl. Math. Comput.*, vol. 245, pp. 191–205, Oct. 2014.
[14] H. Sato, H. E. Aguirre, and K. Tanaka, "Controlling dominance area of solutions and its impact on the performance of mOEAs," in *Evolutionary Multi-Criterion Optimization* (Lecture Notes in Computer Science), vol. 4403, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. Berlin, Germany: Springer, 2007, pp. 5–20.
[15] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: Optimal μ-distributions and the choice of the reference point," in *Proc. Found. Genetic Algorithm*, 2009, pp. 87–102.
[16] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *Evolutionary Multi-Criterion Optimization* (Lecture Notes in Computer Science) vol. 4403. Berlin, Germany: Springer, 2007, pp. 742–756.
[17] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, Mar. 2011.
[18] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving From Nature—PPSN VIII* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2004, pp. 832–842.
[19] B. Li, K. Tang, J. Li, and X. Yao, "Stochastic ranking algorithm for many-objective optimization based on multiple indicators," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 924–938, Dec. 2016.
[20] M. Elarbi, S. Bechikh, A. Gupta, L. Ben Said, and Y.-S. Ong, "A new decomposition-based NSGA-II for many-objective optimization," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 7, pp. 1191–1210, Jul. 2018.
[21] L. Wang, Q. Zhang, M. Gong, L. Jiao, and A. Zhou, "Constrained subproblems in a decomposition-based multiobjective evolutionary algorithm," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 475–480, Jun. 2016.
[22] H. Zhu, Z. He, and Y. Jia, "A novel approach to multiple sequence alignment using multiobjective evolutionary algorithm based on decomposition," *IEEE J. Biomed. Health Inform.*, vol. 20, no. 2, pp. 717–727, Mar. 2016.
[23] S. Jiang and S. Yang, "An improved multiobjective optimization evolutionary algorithm based on decomposition for complex Pareto fronts," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 421–437, Mar. 2016.
[24] A. Zhou and Q. Zhang, "Are all the subproblems equally important? Resource allocation in decomposition-based multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 52–64, Feb. 2016.
[25] H. Zhang, X. Zhang, S. Song, and X.-Z. Gao, "Self-organizing multiobjective optimization based on decomposition with neighborhood ensemble," *Neurocomputing*, vol. 173, pp. 1868–1884, Jan. 2016.
[26] N. Al Mpubayed, A. Petrovski, and J. McCall, "D^2MOPSO: MOPSO based on decomposition and dominance with archiving using crowding distance in objective and solution spaces," *Evol. Comput.*, vol. 22, no. 1, pp. 47–78, 2014.
[27] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Apr. 2013.
[28] Y. Zhou, Y. Xiang, Z. Chen, J. He, and J. Wang, "A scalar projection and angle-based evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2073–2084, Jun. 2019.
[29] M. Asafuddoula, H. K. Singh, and T. Ray, "An enhanced decomposition-based evolutionary algorithm with adaptive reference vectors," *IEEE Trans. Cybern.*, vol. 48, no. 8, pp. 2321–2334, Aug. 2017.
[30] K. Praditwong and X. Yao, "A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm," in *Proc. Int. Conf. Comput. Intell. Secur.*, 2006, pp. 286–291.
[31] B. Li, J. Li, K. Tang, and X. Yao, "An improved two archive algorithm for many-objective optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 2869–2876.
[32] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 524–541, Aug. 2014.
[33] L. Cai, S. Qu, and G. Cheng, "Two-archive method for aggregation-based many-objective optimization," *Inf. Sci.*, vol. 422, pp. 305–317, Jan. 2018.
[34] J. G. Falcón-Cardona and C. A. C. Coello, "A new indicator-based many-objective ant colony optimizer for continuous search spaces," *Swarm Intell.*, vol. 11, no. 1, pp. 71–100, 2017.
[35] R. Cheng, M. Li, Y. Tian, X. Xiang, X. Zhang, S. Yang, Y. Jin, and X. Yao, "Benchmark functions for the CEC'2018 competition on many-objective optimization," CERCIA, School Comput. Sci., Univ. Birmingham Edgbaston, Birmingham, U.K., Tech. Rep. CSR-17-01, 2017.
[36] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 694–716, Oct. 2015.
[37] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 761–776, Dec. 2015.
[38] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 773–791, Oct. 2016.
[39] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.
[40] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: Wiley, 2001.
[41] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.
[42] R. D. G. Steel, J. H. Torrie, and D. A. Dickey, *Principles and Procedures of Statistics: A Biometrical Approach*. New York, NY, USA: McGraw-Hill, 1997.
[43] K. Li, R. Wang, T. Zhang, and H. Ishibuchi, "Evolutionary many-objective optimization: A comparative study of the state-of-the-art," *IEEE Access*, vol. 6, pp. 26194–26214, 2018.
[44] H.-L. Lin, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 3, pp. 433–448, Jun. 2018.
[45] B. Khan, S. Hanoun, M. Johnstone, C. P. Lim, D. Creighton, and S. Nahavandi, "A scalarization-based dominance evolutionary algorithm for many-objective optimization," *Inf. Sci.*, vol. 474, pp. 236–252, Feb. 2019.
[46] M. Asafuddoula, B. Verma, and M. Zhang, "A divide-and-conquer-based ensemble classifier learning by means of many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 5, pp. 762–777, Oct. 2018.
[47] M. Zhang and H. Li, "A reference direction and entropy based evolutionary algorithm for many-objective optimization," *Appl. Soft Comput.*, vol. 70, pp. 108–130, Sep. 2018.
[48] F. Li, R. Cheng, J. Liu, and Y. Jin, "A two-stage R2 indicator based evolutionary algorithm for many-objective optimization," *Appl. Soft Comput.*, vol. 67, pp. 245–260, Jun. 2018.
[49] J. Liu, F. Li, X. Kong, and P. Huang, "Handling many-objective optimisation problems with R2 indicator and decomposition-based particle swarm optimiser," *Int. J. Syst. Sci.*, vol. 50, no. 2, pp. 320–336, 2019. doi: 10.1080/00207721.2018.1552765.