IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 18, 2019, accepted August 22, 2019, date of publication August 26, 2019, date of current version September 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2937538

A Coupling Approach With GSO-BFOA for Many-Objective Optimization

JIANGJIANG ZHANG[1], ZHIHUA CUI[1], YECHUANG WANG[1], HUI WANG[2], XINGJUAN CAI[1], JINJUN CHEN[3], AND WUZHAO LI[4]

[1]Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan 030024, China
[2]School of Information Engineering, Nanchang Institute of Technology, Nanchang 330099, China
[3]Department of Computer Science and Software Engineering, Swinburne University of Technology, Melbourne, VIC 3000, Australia
[4]Jiaxing Vocational Technical College, Jiaxing 314001, China

Corresponding authors: Zhihua Cui (cuizhihua@gmail.com) and Xingjuan Cai (caixingjuan@tyust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61806138 and Grant 61663028, in part by the Natural Science Foundation of Shanxi Province under Grant 201801D121127, in part by the Taiyuan University of Science and Technology Scientific Research Initial Funding under Grant 20182002, in part by the Natural Science Foundation of Zhejiang Province under Grant LY18F030010, in part by the Distinguished Young Talents Plan of Jiangxi Province under Grant 20171BCB23075, and in part by the Natural Science Foundation of Jiangxi Province under Grant 20171BAB202035.

ABSTRACT Glowworm swarm optimization (GSO) and bacterial foraging optimization algorithm (BFOA) are two popular swarm intelligence optimization algorithms (SIOAs). However, both GSO and BFOA show some difficulties when solving many-objective optimization problems (MaOPs). To challenge MaOPs, a coupling approach based on GSO and BFOA is proposed in this paper. To implement the coupling method, an external archive is established to save the best solutions found so far. The internal populations in GSO and BFOA can exchange the search information with the external archive in the evolutionary process. Simulation experiments are verified on two benchmark sets (DTLZ and WFG) with 3 to 15 objectives. The performance of our approach is compared with five other famous algorithms including NSGA-III, KnEA, MOEA/D-DE, GrEA and HypE. Results prove the effectiveness of our approach.

INDEX TERMS Swarm intelligence, glowworm swarm optimization, bacterial foraging optimization algorithm, external archive, many-objective optimization.

I. INTRODUCTION

SIOAs are mostly encouraged by the behaviors of biological swarm systems (e.g. bird flocking, foraging and courtship). There are several popular SIOAs [1], [2], such as bacteria foraging optimization algorithm (BFOA) [3], [4], cuckoo search (CS) [5]–[7] and glowworm swarm optimization (GSO) [8], [9]. In the past decades, these SIOAs have been widely applied to diversiform optimization problems [10], [11]. When projects or systems in real-life become large, some very complex optimization problems have emerged, such as large-scale optimization problems [12], [13] and MaOPs [14], [15]. For these problems, the performance of most SIOAs encounters great challenges [16]–[18]. Therefore, strong and effective SIOAs are required [19], [20].

For MaOPs, the non-dominated proportion of individuals in the population will rise quickly [21]. With increasing of objectives, most individuals in the population become non-dominated solutions [22]–[24]. Hence, most multi-objective evolutionary algorithms (MOEAs) is difficult to handle MaOPs. Meanwhile, the number of optimal solutions covering Pareto Front (PF) increases exponentially, and this makes it impossible to find the complete PF [25]. To solve MaOPs, some researchers have proposed different methods in the last few years. Deb and Jain [26] designed a non-dominated sorting algorithm based on preference points (NSGA-III). Combining with spatial decomposition, NSGA-III employs preference points to guide the population evolution. Fewer preference points in a two-layer mode are used to obtain relatively uniform solution sets. In [27], Xiang et al. used a vector angle EA (VaEA) to solve the unconstrained MaOPs, which adopts the angle between the minimum vectors to find a pair of similar individuals. A deletion strategy is employed to delete those with poor convergence or distribution. Lin *et al.* [28] introduced the idea of clustering to classify the population into many clusters, which

The associate editor coordinating the review of this article and approving it for publication was Bora Onat.

will divide similar individuals into a cluster to show the population's diversity. And the convergence will be guaranteed by employing the simple convergence indicator.

Although SIOAs have shown good performances on single and multi-objective optimization problems (MOPs) having up to 3 objectives, their performance is seriously influenced when the number of objectives exceeds three [29], [30]. For the above mentioned many-objective optimization algorithms (MaOEAs), few of them are SIOAs. In order to make SIOAs possible for solving MaOPs, some improved strategies were proposed [31]–[33]. In [34], Xiang et al. introduced decomposition into ABC to solve MaOPs. Moreover, indicator-based set and reference-point are combined with PSO for many-objective optimization [35]–[37]. The above improved SIOAs were extended to handle MaOPs, but there still exist some issues. The performance of balancing the convergence and diversity is usually not ideal. Moreover, the time complexity cost is unacceptable because of needing a lot of function evaluations.

In this paper, we focus on improving SIOAs for solving MaOPs. It is difficult to use one algorithm to solve all kinds of optimization problems. Each algorithm has distinctive search characteristic. Therefore, it is possible to combine different algorithms to solve more optimization problems. To challenge the MaOPs, a coupling approach based on GSO and BFOA is proposed. To implement the coupling method, an external archive is established to store the best solutions found so far. The internal populations in BFOA and GSO can exchange the search information with the external archive during the evolutionary process. Simulation experiments are verified on two benchmark sets (DTLZ and WFG) with 3 to 15 objectives. The performance of our approach is compared with five other outstanding algorithms including NSGA-III, KnEA [38], MOEA/D-DE [39], GrEA [40] and HypE [41].

The rest of the paper is organized as follows. The related work about MaOPs are given in Section 2. In Section 3, the standard GSO and BFOA are briefly explained. The related knowledge of the coupling algorithm is listed in Section 4. Simulation experiment is implemented in Section 5. Finally, the work is summarized in Section 6.

II. RELATED WORK

In general, the MOPs can be explained as follows:

$$\min f(Y) = \min\ [f_1(Y), f_2(Y), \ldots, f_M(Y)] \quad \begin{cases} g_i(Y) \geq 0, & i = 1, 2, \ldots, k \\ h_j(Y) = 0, & j = 1, 2, \ldots, s \end{cases} \tag{1}$$

where Y is the decision variables, M is the objective numbers, f_M is the Mth objective, $g_i(Y) \geq 0$ and $h_j(Y) = 0$ denote the ith inequality and jth equality constraint, respectively. For MOPs, $M \geq 2$ should be satisfied. When $M > 3$, MOPs are called as MaOPs. Many MOEAs that work well for only a few objectives have difficulties in high-dimensional objective spaces. Therefore, how to effectively solve MaOPs is a challenging task.

In the last few years, some excellent approaches have been developed to solve MaOPs. On the basis of their selection approaches, those approaches can be categorized into three types: Pareto approach, decomposition approach, and indicator approach.

The Pareto method aims to overcome the problem that the selection pressure is reduced in MaOPs. The traditional Pareto dominance method is difficult to distinguish candidate solutions. To tackle this issue, some modified dominance methods were developed to strength the selection pressure. Moreover, some researchers designed new diversity mechanisms based on the Pareto dominance to enhance the selection pressure, such as NSGA-III, GrEA [40] and SPEA2+SDE [42]. Coevolutionary technique [43] was also used to explore the effective information of objective space. Wang *et al.* [44] presented the algorithm for solving MaOPs by integrating Pareto dominance and adaptive objective spatial distribution information. Zhang *et al.* [38] proposed a knee point driven evolutionary algorithm (KnEA), in which the knee point was introduced into the search process of the population. In [45], [46], Zhang et al. designed two efficient non-dominated sorting strategies to overcome many needless dominance comparisons. Although the selection pressure is enhanced by using the Pareto-based approach, the corresponding diversity mechanism still needs to be improved.

The decomposition-based approach achieves a set of Pareto optimal solutions by transforming MaOPs into single objective sub-problems. Zhang and Li [47] proposed an MOEA based on decomposition (MOEA/D), in which the uniform weight vectors in the objective space is generated and the problem is decomposed to make the individual converge toward the PF along the direction of the weight vector. With the in-depth exploration of decomposition strategy, many excellent many-objective optimization algorithms based on decomposition were designed. According to the improved strategies, these algorithms can be divided into four types: 1) modify scaling function; 2) design weight adaptive strategy; 3) select candidate solutions with good convergence and distribution; and 4) generate candidate solutions with superior performance. Some representative algorithms are MOEA/D-M2M [48], DBEA [49], MOEA/DD [50], MaOEA/D-2ADV [51], and RVEA [52].

The indicator-based approach focuses on weighing the comprehensive performance of candidate solutions and giving the decision makers preference information. Zitzler and Künzli [53] proposed the IBEA, which can be combined with a variety of indicators. In IBEA, the convergence and diversity are taken into accounts and the additional diversity preservation mechanism does not required fitness sharing. Bader and Zitzler [41] designed a fast hypervolume-based algorithm (HypE) for solving MaOPs, in which the candidate solutions chosen by hypervolume-based are strictly Pareto optimal, but it is difficult to balance the calculation load and the calculation accuracy. To tackle this issue, some modified indicators methods were proposed to replace hypervolume indicators, such as Δp indicator [54], IGD [55],

GD [56], [57] and maxi-min indicators [58], [59]. The performance evaluation indicators based on distance measurement have been widely used in solving MaOPs because they have simple and effective calculation and comprehensively consider the convergence and distribution of the solution [60], [61].

From the above discussions, we have found that there are few SIOAs for solving MaOPs. According to the No-Free-Lunch theorem, it is inevitable to sacrifice the diversity to improve the convergence. Similarly, sacrificing the partial convergence may improve the diversity. Therefore, it is valuable to use the coupled SIOAs to solve MaOPs. In addition, a coupling approach based on GSO and BFOA is signed to solve MaOPs.

III. THE BASIC GSO AND BFOA

GSO and BFOA are two kinds of swarm intelligence optimization algorithms. Due to the strong global and local search capability, they have been triumphantly applied to various optimization problems [62]–[68]. In this section, we briefly introduce the above two basic algorithms.

A. GLOWWORM SWARM OPTIMIZATION (GSO)

GSO [8], [9], [69] is one of novel SIOAs, which simulates the social behaviors of fireflies in nature by using fluorescein to make communications. Generally speaking, the fluorescein is related to the attractiveness of a firefly. Higher fluorescein means larger attractiveness. As a result, most fireflies will move towards other fireflies with higher fluorescein. The standard GSO consists of four steps: initialization, updating fluorescein, updating position and updating perception range, which are described as follows.

1) INITIALIZATION

At this stage, fireflies are randomly generated in the objective feasible region. In addition, the initial fluorescein and the sensing radius are the same for each firefly.

2) UPDATING FLUORESCEIN

The fluorescein of firefly is directly related to its position in the search space. The higher the evaluation value of the firefly's spatial position, the larger the growth after updating fluorescein. The specific updating fluorescein model is given as follows.

$$flu_i(t) = \gamma Eva(x_i(t)) + (1-\mu)flu_i(t-1) \tag{2}$$

where μ denotes the fluorescein volatility of fireflies; $flu_i(t)$ is the fluorescein value of the firefly i; γ is the updating fluorescein rate of fireflies; $Eva(x_i(t))$ means the evaluation value of firefly i at position $x_i(t)$; and t indicates the current iterations number.

3) UPDATING POSITION

At the updating position stage, each firefly needs to find a better one in its sensing range. The updating direction is determined by a roulette approach. In addition, the selection probability of the neighboring firefly is also calculated according to the fluorescein value. The specific equation for the selection probability of the neighboring firefly is defined as follows.

$$prob_{ij}(t+1) = \frac{flu_j(t+1) - flu_i(t+1)}{\sum\limits_{k \in N_i(t+1)} flu_k(t+1) - flu_i(t+1)} \tag{3}$$

where $N_i(t+1) = \{j : dis_{i,j}(t+1) < rad_i(t+1); flu_i(t+1) < flu_j(t+1)\}$ denotes the neighborhood set of firefly i and $j \in N_i(t+1)$; $rad_i(t+1)$ is the decision radius of firefly i; $dis_{i,j}(t+1)$ indicates the space distance between fireflies i and j; and $prob_{ij}(t+1)$ means the probability of firefly i to firefly j.

In the neighborhood of firefly i, if firefly j is selected, firefly i will change its position as follows.

$$x_i(t) = x_i(t-1) + step\left(\frac{x_j(t-1) - x_i(t-1)}{\left\| x_j(t-1) - x_i(t-1) \right\|}\right) \tag{4}$$

where $step$ represents the moving step size of firefly; and $\left\| x_j(t-1) - x_i(t-1) \right\|$ indicates the Euclidean space distance between firefly i and firefly j.

4) UPDATING PERCEPTION RANGE

After the position of the firefly is updated, the range of perception is dynamically adjusted. The size of the perceived radius is determined by the number of fireflies in the perceived radius.

$$rad_i(t) = \min\{sr_p, \max\{0, \delta(\varepsilon - |N_i(t-1)|) + rad_i(t-1)\}\} \tag{5}$$

where sr_p represents the perception radius; ε indicates the threshold for firefly neighborhood set; and δ is the parameter to adjust the size of firefly's dynamic perception range.

B. BACTERIAL FORAGING OPTIMIZATION ALGORITHM (BFOA)

BFOA [3] simulates the foraging behavior of Escherichia coli, which has three typical patterns, they can be described as follows.

1) CHEMOTAXIS

The foraging behavior of Escherichia coli is mainly based on its own characteristics. The flagella rotates in the direction during the process of foraging. If the flagella rotates counterclockwise, the bacteria will move forward rapidly. The chemotaxis behavior is generally recognized by tumble and swim. Tumble operation can determine the direction of bacteria foraging, and swimming operation is the continuous movement of bacteria in the same direction during the foraging process. The specific equation for foraging behavior is given as below.

$$x^i(j,k,l) = C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} + x^i(j-1,k,l) \tag{6}$$

where $x^i(j, k, l)$ denotes the location of the ith bacteria when it approaches the jth reproduction and the lth elimination and dispersal; T is the largest number iterations; $C(i)$ is the step size of chemotaxis; and $\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$ is a random forward direction of movement.

Algorithm 1 MaGSO($\boldsymbol{P}$) Operator

Begin
Initialize the fluorescein values in $\boldsymbol{P}$;
for $i = 1$ to N
Calculate the objective values of the ith individual;
Update the fluorescein values of the ith individual according to Eq. (2);
Execute the non-dominated sorting according to the fluorescein values;
Randomly select an individual from the first rank based on the non-dominant sorting;
Update the position of the ith individual according to Eq. (8);
end for
Output the updated population $\boldsymbol{P}$;
End

Assume that the objective function value of the ith bacteria at $x^i(j, k, l)$ is $f(x^i(j, k, l))$, and it will continue to move in the same direction, and stop when the objective function value no longer decreases or the largest steps number is reached when $f(x^i(j, k, l)) < f(x^i(j-1, k, l))$. In a sense, the chemotaxis operation is a complex movement process combining the operations of tumble and swimming. The tumble represents the direction of optimization and the swimming indicates the degree of searching feasible solutions in a certain direction.

2) REPRODUCTION

With the continuous absorption of nutrients, Escherichia coli will gradually grow longer. Under appropriate conditions, each bacteria will asexually split into two bacteria. However, the bacteria will be eliminated for those bacteria with poor nutrition. In the reproduction, J^i_{health} is used to represent the energy value of the ith bacteria, which determines the foraging ability of bacteria. And then the bacteria are sorted according to their energy values. The bacteria with energy values ranked in the first half are used for reproduction, and the other half of bacteria are eliminated. The new reproduction has exactly the same foraging ability as the original bacteria. The value of J^i_{health} is calculated by:

$$J^i_{health} = \sum_{j=1}^{N_c} f(x^i(j, k, l)) \tag{7}$$

where J^i_{health} represents the energy value of the ith bacteria; N_c indicates the number of chemotaxis; $f(x^i(j, k, l))$ is the fitness value o f the ith bacteria after the jth chemotaxis, the kth reproduction and the lth elimination and dispersal operations.

3) ELIMINATION AND DISPERSAL

After the reproduction, the bacteria will execute the elimination and dispersal operation with a certain probability. The basic principle of elimination and dispersal operation is similar to the mutation operation in genetic algorithm, which can continue to search in unexploited areas and prevent the population from falling into local minima.

IV. A COUPLING APPROACH BASED ON GSO AND BFOA FOR SOLVING MaOPs

In this section, a coupling approach based on GSO and BFOA is proposed to solve MaOPs. The new approach is called MaGSO-MaBFOA, which employs three important operators: many-objective based GSO (MaGSO) and many-objective based BFOA (MaBFOA), and archive updating. The details of MaGSO-MaBFOA are described as follows.

A. MaGSO OPERATOR

The original GSO was employed to solve simple problems. Although many different GSO variants have been proposed in previous research, few of them focuses on solving MaOPs. To extend GSO to solve MaOPs, a new operator based on GSO (called MaGSO) is proposed as below.

In MaOPs, the dominant percentage of individuals in the population will rise rapidly. Most individuals in the population turn into non-dominated solutions with increasing of objectives. To determine the updating direction of fireflies, GSO needs to compare the fluorescein values in the range of the perception of fireflies. But the original position updating method is not suitable for solving MaOPs. To improve this case, a modified method for position updating is defined by

$$x_i(t) = x_i(t-1) + step\left(\frac{x_j(t-1) - x_i(t-1)}{\left\| x_j(t-1) - x_i(t-1) \right\|}\right) + \varphi r_1\left(\frac{T-t}{T}\right) \tag{8}$$

where φ controls the range of disturbance and it is set to 0.001; r_1 is an uniformly distributed random value; t is the number of iterations; T is the largest number of iterations; $x_j(t)$ is an individual that is randomly selected from the first rank based on the non-dominated sorting at the tth iteration. It is helpful to guide other fireflies to update their positions. Moreover, the disturbance aims to balance diversity and convergence.

The main steps of the MaGSO operator are listed in Algorithm 1. In addition, P and Q1 are the input and output population, respectively. As seen, the fluorescein values are updated in the same way as GSO. In addition, the non-dominated sorting can be obtained by executing on the basis of the fluorescein values.

B. MaBFOA OPERATOR

Like GSO, the original BFOA is usually employed to solve MOPs. To handle MaOPs, some operations in BFOA should be modified. In the following, we design an operator based on BFOA (called MaBFOA) to solve MaOPs.

Algorithm 2 MaBFOA($\boldsymbol{V}$) Operator

```
Begin
  Set the parameters N_ed, N_re, N_c, and N_s;
  for L = 1 to N_ed
    for k = 1 to N_re
      for j = 1 to N_c
        for i = 1 to N
          Evaluate the objective values of V_i as fit_last;
          Generate a random vector Δ_i ∈ [−1, 1];
          Tumble: Make movement for ith bacterium in
            direction of the tumble using Eq. (6);
          Evaluate the tumble objective values of
            V_i as fit_new;
          Flag = Domination(fit_last fit_new);
          if Flag==1 or sum(fit_new − fit_last) >0 then
            The tumbling solution is replaced with the
              solution of not tumbling;
          else
            Remain the tumbling solution unchanged;
          end If
          Swim: Let con=0 (initialize the swim length
              counter);
          while con<   N_s
            con=con+1;
            Conduct the tumbling according to Eq. (6);

            if the next tumbling solution is superior than
              the last tumbling solution
             Exchange the last tumbling solution with
                the next tumbling solution;
            else
              Remain the last tumbling solution
                unchanged and set con= N_s;
            end if
          end while
        end for
      end for
      A = UpdateArchive(A, V);
      Conduct the DE strategy on the current
        population V according to Eq. (9);
    end for
    Conduct the PM strategy on the current population V;
  end for
  Output the updated population V;
End
```

For the chemotaxis , if the new individual X_{new} dominates the original individual X_{origin}, X_{origin} is replaced by X_{new}. If X_{origin} dominates X_{new}, X_{origin} is retained. In addition, if they do not dominate each other and $\sum_{i=1}^{M}\left(f_i(X_{new}) - f_i(X_{origin})\right) > 0$, X_{origin} remains unchanged ($f_i(.)$ is the ith fitness function and M is the objective number).

For the reproduction, if the original reproduction strategy in BFOA is used in MaBFOA, it will be not be helpful to keep the diversity of population. In order to maintain the population diversity, the idea of differential evolution (DE) is introduced to execute the reproduction operation. The specific method is described as follows.

$$X_i = \begin{cases} X_i + Con(X_j - X_i) + Con(X_k - X_i), & if\ r_c \le cross \\ X_j or X_k, & otherwise \end{cases} \tag{9}$$

where Con is the contraction factor; X_i is randomly selected from the current population; X_j and X_k are randomly chosen from the top 10% individuals in the external archive; r_c is an random number within [0,1]; and $cross$ is called crossover probability.

Polynomial mutation (PM) [70] has been widely adopted to solve MaOPs and it provides a promising search ability. To keep the solution having better quality on convergence and diversity, PM is used for the elimination and dispersal. Due to the page limitations, we did not give the detailed implementation of PM. More information of PM can be found in [70].

The main process of the MaBFOA operator are given in Algorithm 2, where V is the input population, UpdateArchive() is the external archive updating operator, N_{ed}, N_{re}, and N_c are the elimination-dispersal, the replicate, chemotactic number, respectively. N_s is the swim length. As seen, the updating methods for the tumble and swim are not changed in MaBFOA.

C. ARCHIVE UPDATING OPERATOR

Our coupling approach employs two important operators: MaGSO(P) and MaBFOA(V), which are run on two independent populations P and V, respectively. To exchange the search experiences of these two populations and store some best solutions, an external archive A is established based on a balanceable fitness estimation (BFE) technique [71]. For MaOPs, the BFE method has been proved to be effective for overcoming the limitations of both Pareto ranking and decomposition approache s [71]. The convergence distance and the diversity distance are considered in the BFE method t o balance the convergence and diversity of population.

Assume that the population $P = \{P_1, P_2, \ldots, P_N\}$ consists of N individuals and the BFE method c an be described as follows.

$$fit(P_i, P) = \alpha Cd(P_i, P) + \beta Cv(P_i, P) \tag{10}$$

where $fit(P_i, P)$ is the BFE value, $Cv(P_i, P)$ represents the convergence distance of the ith individual P_i,$Cd(P_i, P)$ is the normalized diversity of the ith individual P_i. α and β are two weighting factors, which are employed to coordinate the impacts of the distance diversity and convergence, respectively. The weighting factors α and β are adaptively adjusted to balance the diversity and convergence distance. Here we only briefly introduce the BFE method and more details can be found in [71], [72].

It is worth noting that the largest and smallest objective values should be used for normalization when evaluating the

BFE value. The normalization can help to reduce the impact of different amplitudes on many objectives.

Algorithm 3 UpdateArchive(A, K) Operator

```
Begin
 for i = 1 to |K|
   for j = 1 to |A|
       Check the dominance relation between A_i and K_i;
    if A_i dominates   K_i then
          Mark K_i as a dominated solution;
          Mark A_i as a non-dominated solution;
    else
          Mark A_i as a dominated solution;
          Mark K_i as a non-dominated solution;
       end If
   end for
    Remove the dominated solutions from A;
    Add the non-dominated solutions to A;
   if |A|   >   |N| then
       Calculate the fitness values using the BFE method;

       Remove some solutions with the worst fitness
       values;
   end if
 end for
  Output the updated external archive A;
End
```

Based on the updating method, some excellent solutions are chosen from P and V to form the external archive A. If the size of A more than the population size, a selection method is adopted to decide which solutions are retained or deleted. This can guide the search direction to close the true PF. For the current external archive A and the new population K, the selection mechanism firstly checks the dominance relation between the solutions in A and the solutions in K. Then, the dominated solutions and the non-dominated solutions are labeled, respectively. Thirdly, the non-dominated solutions are added to A and the dominated solutions are delated from A. Finally, the worst solution is deleted according to the BFE value. When the size of the external archive equals the population size, the iteration will stop.

The main steps of the archive updating operator UpdateArchive(A, K) are listed in Algorithm 3. In addition, A is the external archive, and K is the internal population.

D. FRAMEWORK OF THE COUPLING ALGORITHM

As mentioned before, the proposed algorithm MaGSO-MaBFOA combines two basic algorithms MaGSO and MaBFOA for solving MaOPs. By exchanging the search information of GSO and BFOA, an external archive is constructed to save some best solutions during the search process. First, MaGSO-MaBFOA defines two initial populations P and V. The MaGSO and MaBFPA operators evolve the populations P and V, respectively. Then, the archive updating operator is employed to update the external archive A. Some better solutions in P and V are selected and stored in A.

Algorithm 4 Framework of the Coupling Algorithm

```
Begin
Initialize the populations P and V, respectively;
Initialize the external archive A;
while t < T do
    P=MaGSO(P);
    A=Update Archive(A P);
    V=MaBFOA(V);
    A=Update Archive(A V);
   Applying the SBX and PM on A to gain a new
    population R;
    Evaluate the objectives of solutions in R;
    A=Update Archive(A, R);
    t = t+2;
 end while
  Output the external archive A;
End
```

To further enhance the quality of solutions in the external archive, two well-known evolutionary operators, simulated binary crossover (SBX) [73] and polynomial mutation (PM), are utilized. By the suggestions of **[74]**, these evolutionary operators can effectively improve the search ability on various MOPs. By conducting the SBX and PM on A, a new population R is gained. In the next step, the archive updating operator is used between A and R to update the external archive A.

TABLE 1. The population size setting.

Number of Objectives (M)	Divisions ($Point$)	Population Size (N)
3	12	91
5	6	210
8	$Point_1$=3, $Point_2$=2	156
10	$Point_1$=3, $Point_2$=2	275
15	$Point_1$=2, $Point_2$=1	135

TABLE 2. The MOEA/D-DE and NSGA-III settings.

Parameters	MOEA/D-DE	NSGA-III
pr_c	1	1
pr_m	1/ N_{var}	1/ N_{var}
dr_c	20	30
dr_m	20	20

The framework of our approach MaGSO-MaBFOA is showed in Algorithm 4. In addition, the t and T are the current and largest iteration number, respectively.

E. COMPLEXITY ANALYSIS

In our proposed approach MaGSO-MaBFOA, the circulation time is set by the largest number of generations. There are three major components in the loop sequences, i.e., MaGSO operator, MaBFOA operator and archive updating operator.

TABLE 3. IGD value of different algorithms on the DTLZ test set.

Problem	M	NSGA-III	KnEA	MOEA/D-DE	GrEA	HypE	MaGSO-MaBFOA
DTLZ1	3	2.1035×10^{1} (6.48×10^{0})=	2.0461×10^{1} (5.55×10^{0})=	**1.2756×10^{1} (9.79×10^{0})+**	2.1485×10^{1} (6.45×10^{0})=	1.4046×10^{1} (3.99×10^{0})+	2.7198×10^{1} (1.27×10^{1})
	5	2.5627×10^{1} (5.68×10^{0})-	2.2296×10^{1} (6.27×10^{0})-	2.5440×10^{1} (8.80×10^{0})-	2.5069×10^{1} (6.97×10^{0})-	1.9700×10^{1} (5.46×10^{0})-	**1.5809×10^{1} (4.29×10^{0})**
	8	2.2097×10^{1} (6.49×10^{0})-	2.3568×10^{1} (6.45×10^{0})-	1.7883×10^{1} (8.76×10^{0})=	2.8273×10^{1} (7.09×10^{0})-	1.8052×10^{1} (6.70×10^{0})=	**1.6816×10^{1} (5.75×10^{0})**
	10	2.9086×10^{1} (9.62×10^{0})-	2.6466×10^{1} (8.39×10^{0})-	2.3444×10^{1} (6.85×10^{0})-	2.6331×10^{1} (7.70×10^{0})-	2.2622×10^{1} (8.75×10^{0})-	**1.0·572×10^{1} (4.05×10^{0})**
	15	2.5056×10^{1} (7.54×10^{0})=	2.0249×10^{1} (7.68×10^{0})=	**1.7185×10^{1} (8.08×10^{0})=**	2.6916×10^{1} (1.02×10^{1})-	1.8413×10^{1} (5.34×10^{0})=	2.0729×10^{1} (7.82×10^{0})
DTLZ2	3	4.3935×10^{-1} (3.50×10^{-2})-	4.0042×10^{-1} (3.42×10^{-2})-	3.9535×10^{-1} (5.13×10^{-3})-	4.1952×10^{-1} (3.30×10^{-2})-	3.9100×10^{-1} (4.51×10^{-2})-	**2.2111×10^{-1} (2.18×10^{-2})**
	5	5.9558×10^{-1} (3.23×10^{-2})=	5.8394×10^{-1} (3.01×10^{-2})=	6.4317×10^{-1} (3.76×10^{-3})-	5.8758×10^{-1} (2.48×10^{-2})=	**5.6949×10^{-1} (2.75×10^{-3})=**	5.8321×10^{-1} (2.77×10^{-2})
	8	8.8693×10^{-1} (4.03×10^{-2})=	8.9226×10^{-1} (2.50×10^{-2})=	9.6259×10^{-1} (5.36×10^{-3})-	**8.7472×10^{-1} (3.10×10^{-2})=**	8.7662×10^{-1} (3.94×10^{-2})=	8.8315×10^{-1} (3.30×10^{-2})
	10	1.0037×10^{0} (2.58×10^{-2})-	9.9660×10^{-1} (3.56×10^{-2})-	1.0010×10^{0} (2.64×10^{-2})-	9.9092×10^{-1} (3.84×10^{-2})-	9.9396×10^{-1} (3.65×10^{-2})-	**9.6008×10^{-1} (2.16×10^{-2})**
	15	1.2050×10^{0} (3.69×10^{-2})=	1.2020×10^{0} (4.93×10^{-2})=	1.2393×10^{0} (4.62×10^{-2})-	**1.1732×10^{0} (5.31×10^{-2})=**	1.1843×10^{0} (4.18×10^{-2})=	1.1855×10^{0} (4.64×10^{-2})
DTLZ3	3	2.1852×10^{2} (3.90×10^{1})+	1.6604×10^{2} (2.26×10^{1})+	**7.7993×10^{1} (5.28×10^{1})+**	1.7254×10^{2} (4.03×10^{1})+	1.3862×10^{2} (3.30×10^{1})+	2.5382×10^{2} (4.43×10^{1})
	5	3.4287×10^{2} (5.80×10^{1})-	3.0434×10^{2} (6.93×10^{1})=	**2.2024×10^{2} (2.31×10^{1})+**	3.1341×10^{2} (4.42×10^{1})-	2.5534×10^{2} (5.48×10^{1})=	2.6681×10^{2} (4.89×10^{1})
	8	3.1615e×10^{2} (6.96×10^{1})=	2.7317×10^{2} (5.74×10^{1})=	**1.7112×10^{2} (4.31×10^{1})+**	3.3913×10^{2} (4.63×10^{1})-	2.1381×10^{2} (3.21×10^{1})+	2.7215×10^{2} (5.22×10^{1})
	10	3.8441×10^{2} (5.78×10^{1})-	3.3056×10^{2} (5.10×10^{1})-	**2.3078×10^{2} (2.09×10^{1})+**	3.5526×10^{2} (6.53×10^{1})-	3.0572×10^{2} (6.27×10^{1})-	2.5534×10^{2} (3.71×10^{1})
	15	3.3048×10^{2} (8.22×10^{1})-	2.4648×10^{2} (6.47×10^{1})=	**1.3820×10^{2} (6.61×10^{1})+**	3.6519×10^{2} (9.66×10^{1})-	1.9724×10^{2} (2.77×10^{1})+	2.6206×10^{2} (5.69×10^{1})
DTLZ4	3	8.3927×10^{-1} (1.49×10^{-1})=	7.8861×10^{-1} (1.64×10^{-1})=	**7.1468×10^{-1} (6.90×10^{-2})=**	7.3758×10^{-1} (1.42×10^{-1})=	8.4284×10^{-1} (1.50×10^{-1})=	7.4635×10^{-1} (1.64×10^{-1})
	5	9.5030×10^{-1} (7.71×10^{-2})-	9.2273×10^{-1} (9.53×10^{-2})=	**8.8224×10^{-1} (4.16×10^{-2})=**	8.9168×10^{-1} (7.73×10^{-2})=	9.2284×10^{-1} (7.50×10^{-2})=	9.0461×10^{-1} (5.53×10^{-2})
	8	1.0707×10^{0} (5.90×10^{-2})+	1.2674×10^{0} (8.49×10^{-2})-	**9.4787×10^{-1} (4.30×10^{-2})+**	1.0613×10^{0} (6.59×10^{-2})=	1.1002×10^{0} (6.82×10^{-2})=	1.1058×10^{0} (7.88×10^{-2})
	10	1.1477×10^{0} (6.78×10^{-2})-	1.1559×10^{0} (3.71×10^{-2})-	1.1381×10^{0} (5.41×10^{-2})-	1.1356×10^{0} (3.10×10^{-2})-	1.1449×10^{0} (5.53×10^{-2})-	**1.0750×10^{0} (4.73×10^{-2})**
	15	1.2258×10^{0} (5.12×10^{-2}) =	1.3976×10^{0} (7.30×10^{-2})-	**1.1080×10^{0} (3.99×10^{-2})+**	1.1915×10^{0} (3.53×10^{-2})+	1.1924×10^{0} (3.93×10^{-2})+	1.2358e+0 (4.15×10^{-2})
DTLZ5	3	3.7334×10^{-1} (4.51×10^{-2})-	3.5058×10^{-1} (4.88×10^{-2})-	2.5175×10^{-1} (3.20×10^{-2})-	3.5111×10^{-1} (4.71×10^{-2})-	3.1178×10^{-1} (4.12×10^{-2})-	**1.4612×10^{-1} (2.73×10^{-2})**
	5	3.5180×10^{-1} (3.63×10^{-2})=	**3.2082×10^{-1} (2.96×10^{-2})=**	3.4762×10^{-1} (3.80×10^{-2})=	3.4915×10^{-1} (3.65×10^{-2})=	3.3071×10^{-1} (2.74×10^{-2})=	3.3896×10^{-1} (3.57×10^{-2})
	8	3.7567×10^{-1} (5.09×10^{-2})=	**3.5108×10^{-1} (3.25×10^{-2})+**	3.5651×10^{-1} (4.80×10^{-2})=	3.6239×10^{-1} (3.68×10^{-2})=	3.5278×10^{-1} (2.45×10^{-2})+	3.7685×10^{-1} (3.03e-2)
	10	3.7717×10^{-1} (3.96×10^{-2})-	3.7665×10^{-1} (3.09×10^{-2})-	3.7212×10^{-1} (3.66×10^{-2})-	3.6606×10^{-1} (3.62×10^{-2})-	3.5427×10^{-1} (2.64×10^{-2})-	**3.2082×10^{-1} (3.43×10^{-2})**
	15	3.9289×10^{-1} (4.36×10^{-2})=	3.7570×10^{-1} (5.24×10^{-2})=	3.7340×10^{-1} (5.06×10^{-2})=	3.9272×10^{-1} (4.65×10^{-2})=	**3.7108×10^{-1} (4.00×10^{-2})=**	3.7864×10^{-1} (3.07×10^{-2})
DTLZ6	3	8.0102×10^{0} (2.29×10^{-1})=	7.8105×10^{0} (2.39×10^{-1})=	**4.6792×10^{0} (7.17×10^{-1})+**	7.8245×10^{0} (2.31×10^{-1})=	7.9044×10^{0} (2.83×10^{-1})=	7.9046×10^{0} (1.99×10^{-1})
	5	8.1075×10^{0} (2.60×10^{-1})+	7.9198×10^{0} (2.93×10^{-1})+	**4.9035×10^{0} (6.08×10^{-1})+**	7.9776×10^{0} (2.86×10^{-1})+	7.9348×10^{0} (2.32×10^{-1})+	8.3420×10^{0} (1.24×10^{-1})
	8	8.1847×10^{0} (2.13×10^{-1})+	8.1055×10^{0} (1.80×10^{-1})+	**5.2774×10^{0} (7.54×10^{-1})+**	8.0484×10^{0} (2.83×10^{-1})+	8.0832×10^{0} (2.30×10^{-1})+	8.3731×10^{0} (1.15×10^{-1})
	10	8.3693×10^{0} (1.34×10^{-1})-	8.3882×10^{0} (6.72×10^{-2})-	8.3913×10^{0} (1.04×10^{-1})-	8.3612×10^{0} (1.05×10^{-1})-	8.4115×10^{0} (7.42×10^{-2})-	**8.2763×10^{0} (1.12×10^{-1})**
	15	8.1698×10^{0} (2.81×10^{-1})+	8.2072×10^{0} (2.48×10^{-1})+	**5.3222×10^{0} (8.40×10^{-1})+**	8.2405×10^{0} (2.29×10^{-1})=	8.1339×10^{0} (2.36×10^{-1})+	8.3719×10^{0} (1.24×10^{-1})
DTLZ7	3	8.3730×10^{0} (6.99×10^{-1})=	8.0308×10^{0} (5.34×10^{-1})=	**7.3507×10^{0} (1.05×10^{0})=**	7.9137×10^{0} (8.25×10^{-1})=	8.1405×10^{0} (7.76×10^{-1})=	7.8233×10^{0} (9.43×10^{-1})
	5	1.3873×10^{1}(1.02×10^{0})=	1.3766×10^{1} (9.13×10^{-1})=	**1.3390×10^{1} (1.40×10^{0})=**	1.3556×10^{1} (1.21×10^{0})=	1.3411×10^{1} (1.04×10^{0})=	1.3848×10^{1} (5.94×10^{-1})
	8	2.3472×10^{1}(1.02×10^{0})=	2.4464×10^{1} (1.38×10^{0})-	2.2484×10^{1} (1.84×10^{0})=	2.3642×10^{1} (1.83×10^{0})=	2.2493×10^{1} (2.45×10^{0})=	**2.2242×10^{1} (2.61×10^{0})**
	10	2.9387×10^{1} (2.63×10^{0})-	2.8478×10^{1} (2.26×10^{0})=	2.8743×10^{1} (2.53×10^{0})=	2.8989×10^{1} (2.35×10^{0})-	2.7266×10^{1} (2.90×10^{0})=	**2.7226×10^{1} (2.72×10^{0})**
	15	4.7406×10^{1} (3.20×10^{0})=	4.7591×10^{1} (4.10×10^{0})=	**4.5127×10^{1} (3.17×10^{0})+**	4.7376×10^{1} (3.27×10^{0})=	4.8852×10^{1} (2.85×10^{0})=	4.7636×10^{1} (2.75×10^{0})
Best/All		0/35	2/35	18/35	2/35	2/35	11/35
+/-/=		5/14/16	5/13/17	13/11/11	4/15/16	9/9/17	---

For the MaGSO operator, the complexity is $O(MN^2)$, which is governed by the non-dominated sorting approach . For the archive updating operator, the step of dominance checking and ranking are involved and the entire process takes at most $O(MN^2)$ time. For the MaBFOA operator , the complexity is $O(N_c(N + N_sN))$ after the chemotaxis. Then the archive updating operator and DE strategy are implemented in turn, and the complexity will change to $O(M(N_c(N+N_sN)+N^2+N^2)N_{re})$ after the reproduction. Therefore, the complexity of the whole MaBFOA operator is $O(M((N_c(N + N_sN) + N^2 + N^2)N_{re} + N^2)N_{ed})$. However, the complexity will be depended on the maximum computational complexity. And it is no doubt that the product of N_c and N_s will be less than the population size N. As a result, the complexity of MaBFOA operator will be approximately denoted as $O(MN^2)$.

Therefore, it is easy to summarize that the overall complexity of the MaGSO-MaBFOA is $O(MN^2+MN^2+MN^2+MN^2+MN^2+MN^2)$. Therefore, the total complexity will be expressed as $O(MN^2)$ by simplifying.

V. SIMULATION STUDY

A. TEST FUNCTIONS

To verify the effectiveness of our approach, two famous benchmark functions, DTLZ and WFG, are adopted in the following experiments. The first benchmark set consists of 7 test problems (DTLZ1-DTLZ7) and the second one contains 9 test functions (WFG1-WFG9). The objective number of these functions varies from 3 to 15. For the test problem DTLZ1, its objective functions $f_i \in [0, 0.5]$. For other test problems of the DTLZ set, the objective functions $f_i \in [0, 1]$. The true PF of WFG1 is mixed and biased. For WFG2, its true PF is convex and disconnected. For WFG3 and WFG4-WFG9, their true PFs are linear and concave, respectively.

B. PERFORMANCE METRICS

To measure the performance of MaOEAs, Inverse generation distance (IGD) [55] and hypervolume (HV) [75] are two popular indicators, which have been proved to consider both convergence and diversity of non-dominated individuals [76]. In this paper, the IGD is deemed as the performance evaluation indicator. The specific IGD is described as follows.

$$IGD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{PF^*} \tag{11}$$

where n is the solutions number in the true PF^*, and d_i denotes the Euclidean distance from the solution i of PF^* to

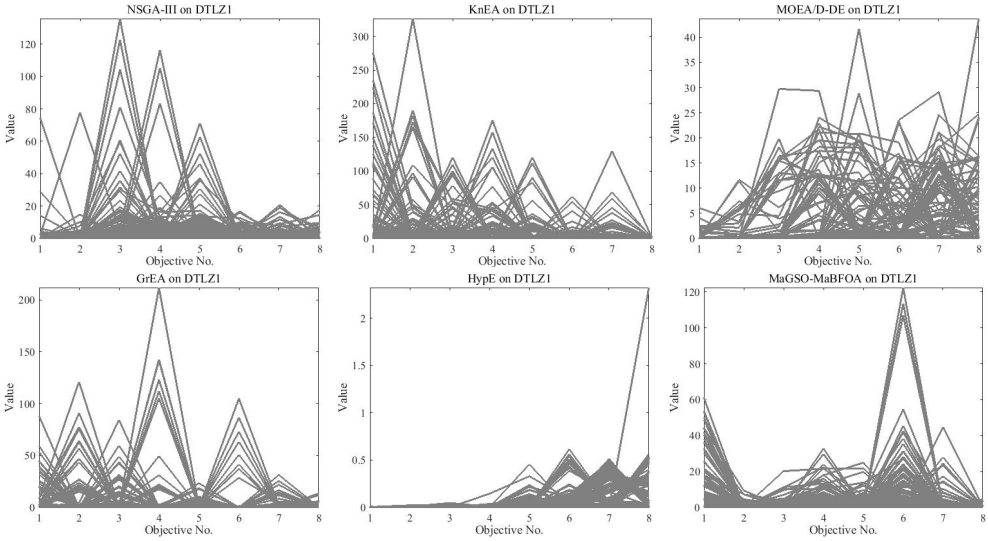


FIGURE 1. The solution of different algorithms on DTLZ1 with 8 objectives.

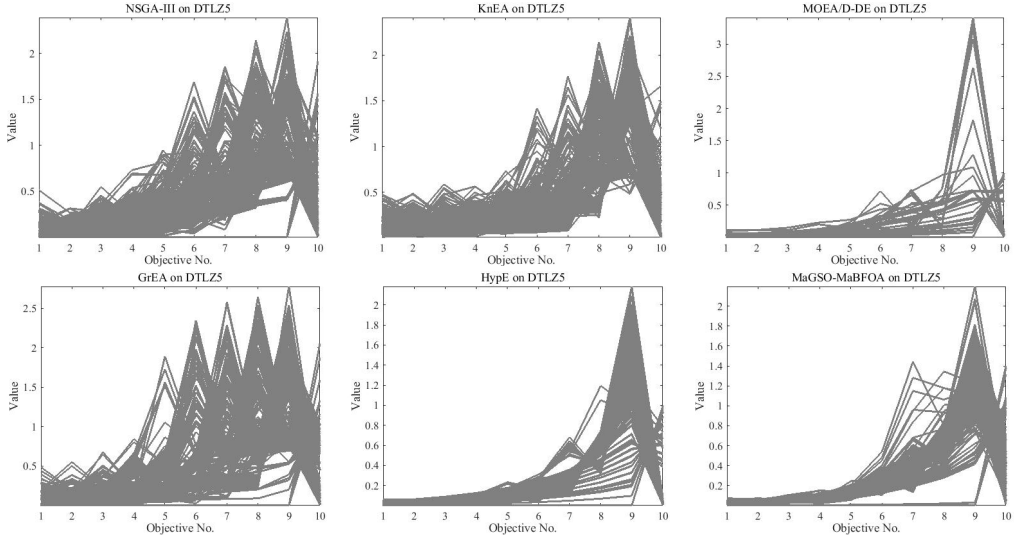


FIGURE 2. The solution of different algorithms on DTLZ5 with 10 objectives.

the closest solution of the approximated PF. The value of IGD is smaller, the performance is the better.

C. INVOLVED ALGORITHMS AND PARAMETER SETTINGS

In the experiments, our approach is compared with five advanced many-objective evolutionary algorithms. The involved algorithms are listed as below.

- NSGA-III [50];
- MOEA/D-DE [39];
- KnEA [38];
- GrEA [40];
- HypE [41];
- Our approach MaGSO-MaBFOA.

To compare in an appropriate environment, the same parameter value are employed by the suggestions of [38]–[41], [50]. For problems with different objective numbers, different values of population size are employed. Table 1 presents the population size settings, where $Point_1$ denotes the boundary layer number of divisions and $Point_2$ indicates the inside layer number of divisions. Table 2 shows the parameter settings of NSGA-III and MOEA/D-DE, where pr_c represents the probability of the SBX, pr_m denotes the probability of the PM, dr_c is the distribution index of population crossover, dr_m is the distribution index of population mutation, and N_{var} is the number of variables. For all algorithms, the largest iterations (T) number is viewed as the stopping condition. For DTLZ1 and DTLZ3, T is set to 700 and 1000, respectively. For the rest of 5 problems (DTLZ2 and DTLZ4-DTLZ7), T is equal to 250. $T = 1000$ and $T = 700$ are used for WFG1 and WFG2, respectively. For the rest of 7 problems (WFG3-WFG9), T is set to 250. For other parameters of GrEA, KnEA, and HypE, the same settings are adopted for these algorithms as in their original papers [38], [40], [41]. In the MaGSO

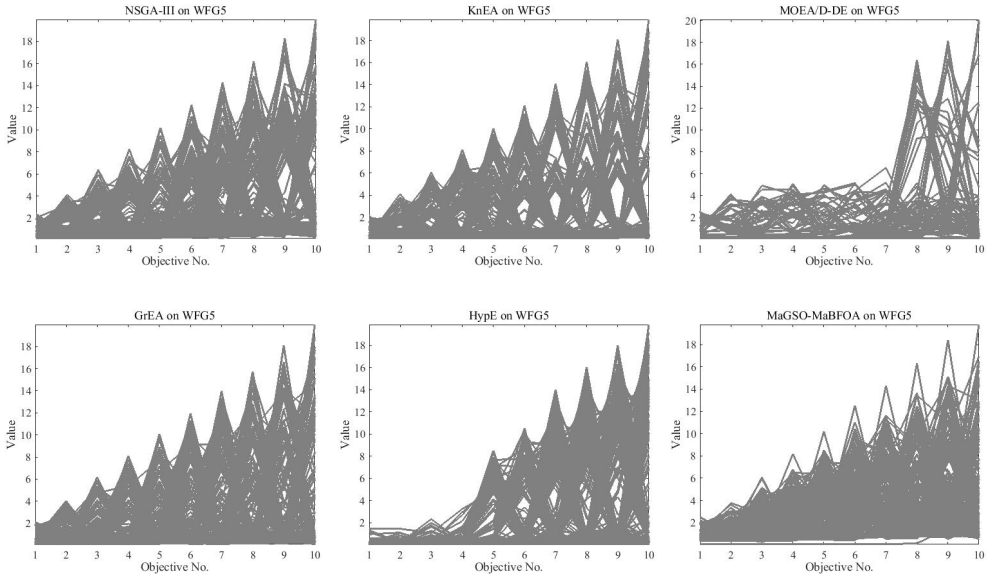


FIGURE 3. The solution of different algorithms on WFG5 with 10 objectives.

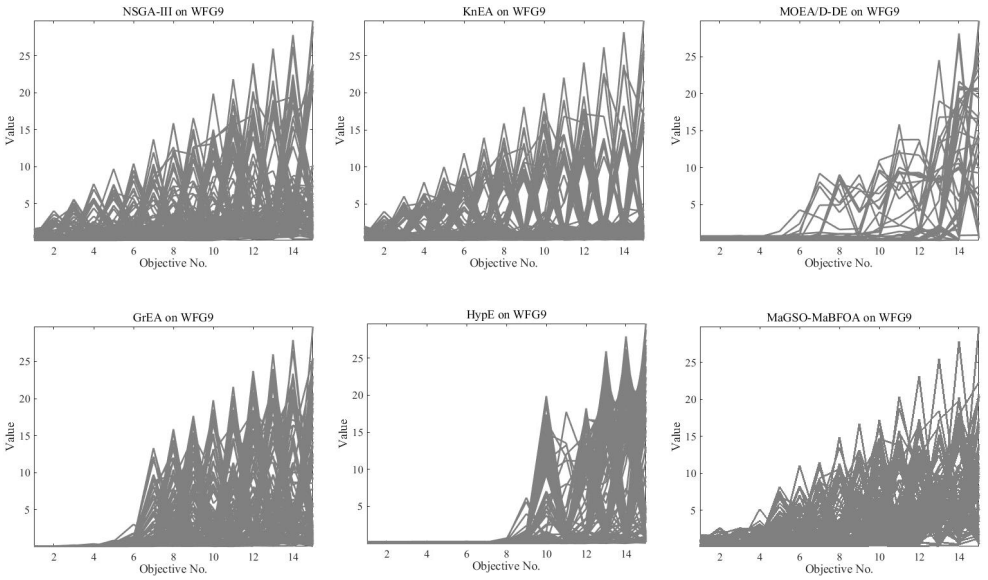


FIGURE 4. The solution of different algorithms on WFG9 with 15 objectives.

operator, the fluorescein volatility μ, the updating fluorescein rate γ, and the moving step size *step* are set to 0.4, 0.6, and 0.03, respectively. In the MaBFOA operator, the step size of chemotaxis $C(i)$ is equal to 0.001. For the reproduction operation, $cross = 1.0$ and $Con = 0.5$ are used. For each test problem, each algorithm is run 20 times.

D. RESULTS ON DTLZ FUNCTION

Table 3 described the comparison results of MaGSO-MaBFOA, NSGA-III, KnEA, MOEA/D-DE, GrEA and HypE on the DTLZ benchmark set. For each test problem, the best result is highlighted and the labels '+', '−', and '=' express that the compared algorithm is significantly better, worse, and equal than (to) our approach. The comparison results on the whole DTLZ benchmark are summarized '*Better/Worse/Similar*', which means that the competitor is better, worse, and similar than (to) our approach on *Better*, *Worse*, and *Similar* problems, respectively. Figs. 1 and 2 list the non-dominated fronts obtained by each algorithm on DTLZ1 with 8 objectives and DTLZ5 with 10 objectives, respectively.

From the results of Table 3, MaGSO-MaBFOA exceeds NSGA-III on 14 problems, while NSGA-III achieves better solutions than MaGSO-MaBFOA on 5 problems. For the rest of 16 problems, both of them obtain similar performance. Compared to KnEA, our approach is better on 13 problems, but KnEA achieves better results on 5 problems. MOEA/D-DE is mildly better than MaGSO-MaBFOA, because MOEA/D-DE executes better on 13 problems and

TABLE 4. IGD value of different algorithms on the WFG test set.

Problem	M	NSGAIII	KnEA	MOEAD-DE	GrEA	HypE	MaGSO-MaBFOA
WFG1	3	1.9955×10^0(5.85×10^-2) +	1.9217×10^0 (7.01×10^-2) +	2.0199×10^0 (4.39×10^-2) +	**1.7824×10^0(7.37×10^-2) +**	2.0806×10^0 (8.49×10^-2) +	2.2722×10^0 (9.37×10^-2)
	5	2.5433×10^0(4.39×10^-2) +	2.4719×10^0 (4.52×10^-2) +	2.6928×10^0 (6.29×10^-2) -	**2.4701×10^0(6.24×10^-2) +**	2.5094×10^0 (4.32×10^-2) +	2.5835×10^0 (4.39×10^-2)
	8	3.0667×10^0(3.40×10^-2) +	2.9983×10^0 (5.78×10^-2) +	3.1789×10^0 (4.77×10^-2) -	**2.9185×10^0(3.17×10^-2) +**	3.0332×10^0 (4.45×10^-2) +	3.1079×10^0 (3.03×10^-2)
	10	3.3549×10^0(2.69×10^-2) +	3.3558×10^0 (3.20×10^-2) +	3.4711×10^0 (3.88×10^-2) -	**3.2885×10^0(3.90×10^-2) +**	3.3578×10^0 (3.64×10^-2) =	3.3758×10^0 (1.53×10^-2)
	15	4.3476×10^0(4.85×10^-2) =	4.2992×10^0 (6.45×10^-2) +	4.4142×10^0 (2.92×10^-2) -	**4.2585×10^0(6.07×10^-2) +**	4.2886×10^0 (4.68×10^-2) +	4.3521×10^0 (3.93×10^-2)
WFG2	3	5.9698×10^-1(8.01×10^-2) +	5.2078×10^-1 (6.76×10^-2) +	5.9026×10^-1 (3.49×10^-2) +	5.4696×10^-1 (7.70×10^-2) +	**5.1803×10^-1 (8.40×10^-2) +**	6.8161×10^-1 (9.45×10^-2)
	5	1.2433×10^0(1.83×10^-1) -	1.0791×10^0 (9.32×10^-2) =	1.2154×10^0 (7.98×10^-2) -	1.0945×10^0 (1.14×10^-1) =	1.0661×10^0 (1.12×10^-1) =	**1.0257×10^0 (2.06×10^-1)**
	8	2.1720×10^0(3.73×10^-1) -	2.0759×10^0 (4.23×10^-1) -	1.8848×10^0 (1.74×10^-1) =	1.8826×10^0 (2.28×10^-1) =	2.1016×10^0 (3.22×10^-1) -	**1.8236×10^0 (2.70×10^-1)**
	10	3.2190×10^0(6.43×10^-1) -	3.0622×10^0 (5.41×10^-1) -	2.9720×10^0 (4.17×10^-1) -	2.9280×10^0 (5.91×10^-1) -	3.0004×10^0 (3.86×10^-1) -	**2.3932×10^0 (4.76×10^-1)**
	15	5.1632×10^0(8.44×10^-1) -	5.8393×10^0 (1.08×10^0) -	4.8876×10^0 (8.64×10^-1) -	**3.8122×10^0 (5.29×10^-1) =**	5.4460×10^0 (8.34×10^-1) -	3.8127×10^0 (8.14×10^-1)
WFG3	3	6.5145×10^-1(3.15×10^-2) -	6.3652×10^-1 (4.19×10^-2) -	6.9134×10^-1 (5.96×10^-2) -	6.3313×10^-1 (3.91×10^-2) -	6.0651×10^-1 (3.92×10^-2) -	**4.4803×10^-1 (3.83×10^-2)**
	5	8.8066×10^-1(4.24×10^-2) =	8.7736×10^-1 (4.06×10^-2) =	9.7489×10^-1 (5.84×10^-2) -	8.6898×10^-1 (3.32×10^-2) =	**8.5293×10^-1 (4.62×10^-2) +**	8.8484×10^-1 (2.94×10^-2)
	8	1.3395×10^0(6.06×10^-2) -	1.3432×10^0 (1.03×10^-1) -	1.3957×10^0 (9.01×10^-2) -	1.3100×10^0 (5.48×10^-2) -	**1.2532×10^0 (7.04×10^-2) =**	1.2749×10^0 (3.38×10^-2)
	10	1.5308×10^0(5.16×10^-2) -	1.5239×10^0 (4.16×10^-2) -	1.5235×10^0 (5.64×10^-2) -	1.5088×10^0 (6.47×10^-2) -	1.5064×10^0 (5.61×10^-2) -	**1.4097×10^0 (5.36×10^-2)**
	15	2.2869×10^0(1.29×10^-1) -	2.3624×10^0 (2.28×10^-1) -	2.3314×10^0 (1.45×10^-1) -	2.1631×10^0 (1.10×10^-1) =	**2.0867×10^0 (1.15×10^-1) =**	2.1012×10^0 (7.47×10^-2)
WFG4	3	6.3062×10^-1(6.57×10^-2) -	6.3774×10^-1 (7.79×10^-2) -	7.1975×10^-1 (5.24×10^-2) -	6.0841×10^-1 (4.89×10^-2) -	5.6433×10^-1 (4.66×10^-2) -	**4.9095×10^-1 (3.41×10^-2)**
	5	2.2428×10^0(1.61×10^-1) =	2.1910×10^0 (1.38×10^-1) =	**1.7418×10^0 (8.75×10^-2) +**	2.0341×10^0 (1.57×10^-1) +	2.1650×10^0 (2.02×10^-1) =	2.1673×10^0 (1.94×10^-1)
	8	6.6547×10^0(4.87×10^-1) -	6.8151×10^0 (4.52×10^-1) -	**5.0167×10^0 (3.86×10^-1) +**	6.3069×10^0 (3.51×10^-1) =	6.9374×10^0 (5.44×10^-1) -	6.3605×10^0 (2.69×10^-1)
	10	9.5593×10^0(4.74×10^-1) -	9.5311×10^0 (5.18×10^-1) -	9.4960×10^0 (4.78×10^-1) -	9.5528×10^0 (4.80×10^-1) -	9.4353×10^0 (4.40×10^-1) -	**8.8940×10^0 (3.29×10^-1)**
	15	1.8692×10^1(8.82×10^-1) =	1.8990×10^1 (1.09×10^0) =	**1.6488×10^1 (1.35×10^0) +**	1.8052×10^1 (7.54×10^-1) =	1.9694×10^1 (6.04×10^-1) -	1.8423×10^1 (6.58×10^-1)
WFG5	3	7.3663×10^-1(3.17×10^-2) -	7.2479×10^-1 (2.77×10^-2) -	6.8697×10^-1 (3.65×10^-2) -	7.1598×10^-1 (1.98×10^-2) -	6.8523×10^-1 (1.80×10^-2) -	**6.5259×10^-1 (2.96×10^-2)**
	5	1.6745×10^0(8.31×10^-2) -	1.6899×10^0 (6.75×10^-2) -	1.7460×10^0 (9.16×10^-2) -	1.6115×10^0 (4.46×10^-2) =	1.6129×10^0 (7.76×10^-2) =	**1.6064×10^0 (5.13×10^-2)**
	8	5.1092×10^0(2.34×10^-1) -	5.0046×10^0 (2.19×10^-1) -	5.2751×10^0 (2.11×10^-1) -	**4.7828×10^0 (2.14×10^-1) =**	4.9550×10^0 (2.93×10^-1) -	4.7867×10^0 (1.47×10^-1)
	10	7.3640×10^0(3.80×10^-1) -	7.1683×10^0 (2.33×10^-1) -	7.1130×10^0 (2.77×10^-1) -	7.2512×10^0 (2.68×10^-1) -	7.3273×10^0 (2.73×10^-1) -	**6.6828×10^0 (2.28×10^-1)**
	15	1.4855×10^1(6.28×10^-1) =	1.5835×10^1 (1.03×10^0) -	1.5953×10^1 (8.15×10^-1) -	1.5377×10^1 (6.98×10^-1) -	1.6558×10^1 (7.16×10^-1) -	**1.4541×10^1 (5.83×10^-1)**
WFG6	3	8.5689×10^-1(2.60×10^-2) -	8.2949×10^-1 (4.30×10^-2) -	9.3171×10^-1 (6.02×10^-2) -	8.3002×10^-1 (3.42×10^-2) -	8.2544×10^-1 (3.11×10^-2) -	**7.1249×10^-1 (4.06×10^-2)**
	5	1.8285×10^0(8.46×10^-2) =	1.8521×10^0 (5.99×10^-2) -	1.9180×10^0 (1.39×10^-1) -	**1.7789×10^0 (5.67×10^-2) =**	1.7937×10^0 (9.75×10^-2) =	1.7900×10^0 (3.67×10^-2)
	8	5.3587×10^0(4.09×10^-1) -	5.2313×10^0 (2.95×10^-1) =	5.3324×10^0 (4.71×10^-1) =	**5.0700×10^0 (2.97×10^-1) =**	5.3669×10^0 (3.77×10^-1) -	5.0700×10^0 (2.15×10^-1)
	10	7.6676×10^0(3.44×10^-1) -	7.4849×10^0 (2.48×10^-1) -	7.5057×10^0 (2.49×10^-1) -	7.5499×10^0 (3.85×10^-1) -	7.5846×10^0 (2.71×10^-1) -	**7.0197×10^0 (2.72×10^-1)**
	15	1.5645×10^1(5.03×10^-1) =	1.5873×10^1 (1.38×10^0) =	1.6745×10^1 (9.63×10^-1) -	1.5702×10^1 (6.00×10^-1) =	1.6835×10^1 (1.11×10^0) -	**1.5334×10^1 (4.88×10^-1)**
WFG7	3	6.9011×10^-1(3.37×10^-2) -	6.6432×10^-1 (2.77×10^-2) -	7.8009×10^-1 (4.25×10^-2) -	6.7080×10^-1 (3.54×10^-2) -	6.4073×10^-1 (3.02×10^-2) -	**5.1371×10^-1 (2.57×10^-2)**
	5	1.7781×10^0(1.03×10^-1) =	**1.7090×10^0 (7.21×10^-2) +**	1.9270×10^0 (1.32×10^-1) -	1.7161×10^0 (1.02×10^-1) +	1.7366×10^0 (1.13×10^-1) =	1.7755×10^0 (7.69×10^-2)
	8	5.6086×10^0(4.07×10^-1) -	5.4192×10^0 (3.21×10^-1) =	5.8374×10^0 (5.41×10^-1) -	**5.3346×10^0 (2.91×10^-1) =**	5.6148×10^0 (4.12×10^-1) -	5.3943×10^0 (2.45×10^-1)
	10	8.1099×10^0(3.70×10^-1) -	7.9771×10^0 (4.61×10^-1) -	8.2108×10^0 (4.72×10^-1) -	8.0167×10^0 (3.94×10^-1) -	7.8904×10^0 (3.91×10^-1) -	**7.4127×10^0 (3.65×10^-1)**
	15	1.6272×10^1(5.50×10^-1) =	1.6326×10^1 (9.19×10^-1) =	1.7182×10^1 (8.43×10^-1) -	**1.5964×10^1 (6.93×10^-1) =**	1.7304×10^1 (7.15×10^-1) -	1.6005×10^1 (4.79×10^-1)
WFG8	3	9.0430×10^-1(3.92×10^-2) -	8.6399×10^-1 (3.52×10^-2) -	9.9796×10^-1 (5.91×10^-2) -	8.8754×10^-1 (3.69×10^-2) -	8.3003×10^-1 (4.00×10^-2) -	**7.5421×10^-1 (3.97×10^-2)**
	5	2.0495×10^0(8.38×10^-2) -	1.9989×10^0 (8.67×10^-2) =	2.0620×10^0 (1.15×10^-1) -	1.9757×10^0 (9.10×10^-2) =	**1.9514×10^0 (5.31×10^-2) =**	1.9631×10^0 (7.32×10^-2)
	8	5.4770×10^0(2.74×10^-1) -	5.4357×10^0 (2.28×10^-1) -	5.6252×10^0 (3.18×10^-1) -	5.2716×10^0 (2.46×10^-1) =	**5.5264×10^0 (1.99×10^-1) -**	5.2729×10^0 (2.13×10^-1)
	10	7.8211×10^0(3.20×10^-1) -	7.6673×10^0 (3.37×10^-1) -	7.7696×10^0 (3.86×10^-1) -	7.7766×10^0 (3.50×10^-1) -	7.8607×10^0 (3.00×10^-1) -	**7.1670×10^0 (3.11×10^-1)**
	15	1.5523×10^1(6.25×10^-1) =	1.6346×10^1 (1.15×10^0) -	1.6716×10^1 (1.20×10^0) -	1.6047×10^1 (7.43×10^-1) -	1.7456×10^1 (7.51×10^-1) -	**1.5395×10^1 (5.09×10^-1)**
WFG9	3	8.5548×10^-1(4.46×10^-2) -	8.0330×10^-1 (6.10×10^-2) -	9.0904×10^-1 (6.44×10^-2) -	8.3255×10^-1 (4.10×10^-2) -	8.0321×10^-1 (4.47×10^-2) -	**5.9704×10^-1 (3.82×10^-2)**
	5	2.0229×10^0(7.81×10^-2) -	1.9783×10^0 (9.40×10^-2) =	2.1819×10^0 (9.92×10^-2) -	1.9390×10^0 (9.38×10^-2) =	**1.9645×10^0 (9.80×10^-2) =**	1.9635×10^0 (5.69×10^-2)
	8	5.6797×10^0(4.06×10^-1) =	5.4755×10^0 (3.26×10^-1) =	6.0743×10^0 (3.61×10^-1) -	5.3029×10^0 (2.44×10^-1) +	**5.6500×10^0 (3.03×10^-1) =**	5.5057×10^0 (3.39×10^-1)
	10	7.9414×10^0(3.28×10^-1) -	7.8318×10^0 (3.05×10^-1) -	7.9109×10^0 (4.34×10^-1) -	7.9317×10^0 (3.73×10^-1) -	7.9494×10^0 (4.33×10^-1) -	**7.4463×10^0 (3.56×10^-1)**
	15	1.6263×10^1(6.22×10^-1) -	1.7336×10^1 (1.41×10^0) -	1.7056×10^1 (6.14×10^-1) -	1.5957×10^1 (8.92×10^-1) =	1.7491×10^1 (1.10×10^0) -	**1.5768×10^1 (7.91×10^-1)**
Best/All		0/45	1/45	3/45	14/45	5/45	22/45
+/-/=		5/29/11	7/28/10	5/36/4	8/19/18	6/28/11	---

MaGSO-MaBFOA is better on 11 problems. GrEA is superior to our approach on 4 problems, while it achieves worse results on 15 problems. Both HypE and MaGSO-MaBFOA obtain similar performance on this benchmark set.

From Fig.1 and Table 3, NSGA-III and GrEA have better distribution than other algorithms on DTLZ1, and MOEA/D-DE and HypE fail to obtain a set of well distributed solutions. It demonstrates that NSGA-III and GrEA can achieve good diversity, while MOEA/D-DE and HypE obtain good convergence. Among six algorithms, our approach MaGSO-MaBFOA obtains better convergence than other algorithms on this problem. For DTLZ2 and DTLZ4, MaGSO-MaBFOA is better than other five algorithms and NSGA-III, KnEA and HypE obtain similar performance. MOEA/D-DE achieves poor convergence and diversity on DTLZ2. Both GrEA and MOEA/D-DE have a similar performance on DTLZ4. For DTLZ3 and DTLZ6, MaGSO-MaBFOA is better than the four algorithms, but it is mildly worse than MOEA/D-DE. The main reason is that the differential evolution strategy is introduced to make the solution quickly approach the true PF. For DTLZ5, it is hard to obtain solutions having good convergence and distribution. From Fig. 2, NSGA-III, KnEA, and GrEA can achieve uniformly distributed solutions, but the solution distributions of MOEA/D-DE and HypE are poor. For DTLZ7, the PF is discontinuous, which is challenge to keep diversity

TABLE 5. IGD value of MaGSO, MaBFOA and MaGSO-MaBFOA on the WFG test set.

Problem	M	MaGSO	MaBFOA	MaGSO-MaBFOA
WFG1	3	2.4199×10^{0} (4.46×10^{-2}) -	2.4186×10^{0} (7.01×10^{-2}) -	**2.2853×10^{0} (7.66×10^{-2})**
	5	2.6864×10^{0} (4.41×10^{-2}) -	2.6893×10^{0} (3.84×10^{-2}) -	**2.5754×10^{0} (4.00×10^{-2})**
	8	3.1964×10^{0} (4.90×10^{-2}) -	3.1873×10^{0} (4.33×10^{-2}) -	**3.1226×10^{0} (2.64×10^{-2})**
	10	3.4459×10^{0} (4.02×10^{-2}) -	3.4484×10^{0} (2.90×10^{-2}) -	**3.3846×10^{0} (2.92×10^{-2})**
	15	4.3968×10^{0} (3.91×10^{-2}) -	4.3781×10^{0} (2.86×10^{-2}) -	**4.3315×10^{0} (3.35×10^{-2})**
WFG2	3	1.0170×10^{0} (9.74×10^{-2}) -	9.3421×10^{-1} (8.60×10^{-2}) -	**7.0563×10^{-1} (9.95×10^{-2})**
	5	1.7464×10^{0} (2.42×10^{-1}) -	1.6423×10^{0} (2.27×10^{-1}) -	**1.0004×10^{0} (1.34×10^{-1})**
	8	2.8450×10^{0} (2.18×10^{-1}) -	3.3160×10^{0} (4.07×10^{-1}) -	**1.9514×10^{0} (4.15×10^{-1})**
	10	3.2730×10^{0} (4.56×10^{-1}) -	3.7081×10^{0} (6.53×10^{-1}) -	**2.4836×10^{0} (5.14×10^{-1})**
	15	4.2346×10^{0} (7.65×10^{-1}) -	7.1976×10^{0} (9.54×10^{-1}) -	**3.9071×10^{0} (9.06×10^{-1})**
WFG3	3	7.4709×10^{-1} (2.08×10^{-2}) -	5.3592×10^{-1} (3.40×10^{-2}) -	**4.2251×10^{-1} (3.43×10^{-2})**
	5	9.5783×10^{-1} (3.42×10^{-2}) -	**6.8414×10^{-1} (3.67×10^{-2})** +	8.6790×10^{-1} (2.64×10^{-2})
	8	1.3949×10^{0} (4.95×10^{-2}) -	**1.0088×10^{0} (4.71×10^{-2})** +	1.2696×10^{0} (4.15×10^{-2})
	10	1.5195×10^{0} (4.09×10^{-2}) -	1.5116×10^{0} (4.92×10^{-2}) -	**1.4252×10^{0} (5.11×10^{-2})**
	15	2.3432×10^{0} (1.06×10^{-1}) -	**1.7303×10^{0} (8.49×10^{-2})** +	2.0548×10^{0} (7.89×10^{-2})
WFG4	3	7.5063×10^{-1} (7.03×10^{-2}) -	7.9606×10^{-1} (1.13×10^{-1}) -	**5.0347×10^{-1} (3.65×10^{-2})**
	5	2.1771×10^{0} (1.76×10^{-1}) =	2.3627×10^{0} (2.19×10^{-1}) -	**2.0800×10^{0} (1.20×10^{-1})**
	8	**6.2189×10^{0} (7.67×10^{-1})** =	6.9821×10^{0} (3.46×10^{-1}) -	6.2729×10^{0} (2.50×10^{-1})
	10	9.4820×10^{0} (4.02×10^{-1}) -	9.6749×10^{0} (3.94×10^{-1}) -	**8.9650×10^{0} (4.23×10^{-1})**
	15	**1.6227×10^{1} (3.03×10^{0})** =	1.9788×10^{1} (9.13×10^{-1}) -	1.8402×10^{1} (5.21×10^{-1})
WFG5	3	8.2002×10^{-1} (2.75×10^{-2}) -	8.2329×10^{-1} (3.45×10^{-2}) -	**6.6569×10^{-1} (2.59×10^{-2})**
	5	1.7886×10^{0} (7.38×10^{-2}) -	1.6440×10^{0} (3.06×10^{-2}) -	**1.6111×10^{0} (3.44×10^{-2})**
	8	5.3368×10^{0} (2.65×10^{-1}) -	**4.6374×10^{0} (1.76×10^{-1})** +	4.7810×10^{0} (1.88×10^{-1})
	10	7.1834×10^{0} (2.27×10^{-1}) -	7.1357×10^{0} (2.23×10^{-1}) -	**6.7788×10^{0} (2.08×10^{-1})**
	15	1.5867×10^{1} (7.04×10^{-1}) -	**1.3723×10^{1} (6.32×10^{-1})** +	1.4690×10^{1} (6.17×10^{-1})
WFG6	3	1.0408×10^{0} (5.03×10^{-2}) -	8.3928×10^{-1} (3.75×10^{-2}) -	**7.1955×10^{-1} (4.30×10^{-2})**
	5	1.9698×10^{0} (7.80×10^{-2}) -	1.9097×10^{0} (6.88×10^{-2}) -	**1.7937×10^{0} (6.05×10^{-2})**
	8	5.3536×10^{0} (3.03×10^{-1}) =	5.5808×10^{0} (3.27×10^{-1}) -	**5.1600×10^{0} (2.76×10^{-1})**
	10	7.4416×10^{0} (2.79×10^{-1}) -	7.6432×10^{0} (2.48×10^{-1}) -	**6.9351×10^{0} (3.09×10^{-1})**
	15	**1.4298×10^{1} (1.54×10^{0})** =	1.6590×10^{1} (7.88×10^{-1}) -	1.5215×10^{1} (4.86×10^{-1})
WFG7	3	8.1153×10^{-1} (4.31×10^{-2}) -	7.0629×10^{-1} (5.91×10^{-2}) -	**5.1808×10^{-1} (2.65×10^{-2})**
	5	1.9158×10^{0} (9.36×10^{-2}) -	1.8482×10^{0} (1.10×10^{-1}) -	**1.7474×10^{0} (9.21×10^{-2})**
	8	**5.3200×10^{0} (4.53×10^{-1})** =	5.7076×10^{0} (3.23×10^{-1}) -	5.4409×10^{0} (2.87×10^{-1})
	10	8.0492×10^{0} (3.71×10^{-1}) -	8.0557×10^{0} (3.29×10^{-1}) -	**7.4100×10^{0} (3.35×10^{-1})**
	15	**1.4564×10^{1} (2.29×10^{0})** =	1.7229×10^{1} (9.00×10^{-1}) -	1.6020×10^{1} (6.19×10^{-1})
WFG8	3	9.9298×10^{-1} (3.77×10^{-2}) -	9.4249×10^{-1} (4.19×10^{-2}) -	**7.5295×10^{-1} (3.65×10^{-2})**
	5	2.0913×10^{0} (8.37×10^{-2}) -	2.0897×10^{0} (8.13×10^{-2}) -	**1.9408×10^{0} (5.43×10^{-2})**
	8	**5.2876×10^{0} (4.12×10^{-1})** =	5.9172×10^{0} (3.49×10^{-1}) -	5.3502×10^{0} (1.83×10^{-1})
	10	7.8208×10^{0} (2.69×10^{-1}) -	7.7899×10^{0} (3.11×10^{-1}) -	**7.3160×10^{0} (2.23×10^{-1})**
	15	**1.4695×10^{1} (1.91×10^{0})** =	1.6770×10^{1} (8.56×10^{-1}) -	1.5492×10^{1} (5.05×10^{-1})
WFG9	3	1.0215×10^{0} (7.08×10^{-2}) -	7.8428×10^{-1} (4.02×10^{-2}) -	**5.8978×10^{-1} (5.27×10^{-2})**
	5	2.1672×10^{0} (8.28e-2) -	**1.7442×10^{0} (6.39×10^{-2})** +	1.9610×10^{0} (6.65×10^{-2})
	8	5.9862×10^{0} (4.48×10^{-1}) -	**4.9626×10^{0} (2.49×10^{-1})** +	5.4368×10^{0} (3.08×10^{-1})
	10	7.7797×10^{0} (3.44×10^{-1}) -	8.0186×10^{0} (3.27×10^{-1}) -	**7.4614×10^{0} (2.76×10^{-1})**
	15	1.6557×10^{1} (6.11×10^{-1}) -	**1.4513×10^{1} (5.79×10^{-1})** +	1.5737×10^{1} (6.23×10^{-1})
Best/All		7/45	8/45	30/45
+/-/=		0/36/9	8/37/0	---

and convergence. From the above analysis, our approach can achieve promising performance on most problems of the DTLZ benchmark set. It confirms the effectiveness of the proposed coupling method.

E. RESULTS ON WFG FUNCTION

Table 4 described the comparison results of MaGSO-MaBFOA, NSGA-III, KnEA, MOEA/D-DE, GrEA and HypE on the WFG benchmark set. The comparison results on the whole WFG benchmark are summarized '*Better/Worse/Similar*', which means that the competitor is better, worse, and similar than (to) our approach on *Better*, *Worse*, and *Similar* problems, respectively. Figs. 3 and 4 present the approximate PF gained by different algorithm on WFG5 with 10 objectives and DTLZ9 with 15 objectives, respectively.

From the results of Table 4, MaGSO-MaBFOA exceeds NSGA-III on 29 problems, while NSGA-III achieves better solutions than MaGSO-MaBFOA on 5 problems. For the rest of 11 problems, both of them obtain similar performance. Compared to KnEA, our approach is better on 28 problems, but KnEA achieves better results on 7 problems. For the rest of 10 problems, both of them obtain similar performance. Our approach performs better than MOEA/D-DE on 36 problems, but MOEA/D-DE obtains better results on 5 problems. GrEA is superior to our approach on 8 problems, while it achieves worse results on 19 problems. For the rest of 18 problems,

both of them obtain similar performance. HypE achieves better solutions than MaGSO-MaBFOA on 6 problems, while our method is better than HypE on 28 problems.

For WFG1, MaGSO-MaBFOA obtains better convergence than MOEA/D-DE. The four other algorithms is better than MaGSO-MaBFOA on this problem. For WFG2 and WFG3, MaGSO-MaBFOA outperforms other five algorithms. MOEA/D-DE is the best of six algorithms on WFG4 with 3, 8 and 15 objectives, and MaGSO-MaBFOA obtain a better convergence than other four algorithms. For the rest of WFG test problems, MaGSO-MaBFOA performs better than, or at least similar to other five algorithms. From Fig. 3, NSGA-III, KnEA, GrEA, and MaGSO-MaBFOA have similar performance on WFG5 with 10 objectives. For WFG9 with 15 objectives, NSGA-III, KnEA and MaGSO-MaBFOA have similar performance. From the above results, it can be found that the proposed MaGSO-MaBFOA has better performance than other five algorithms on the WFG benchmark set.

F. INVESTIGATE THE EFFECTIVENESS OF OUR COUPLE APPROACH

It is difficult to use a single GSO or BFOA to solve MaOPs. To challenge the MaOPs, a coupling approach based on GSO and BFOA is proposed in this paper. Our coupling approach employs three important operators: many-objective based GSO (MaGSO), many-objective based BFOA (MaBFOA), and archive updating. To implement the coupling method, an external archive is established to store the best solutions found so far. The internal populations in BFOA and GSO can exchange the search information with the external archive during the search process.

To test the effectiveness of our coupling approach, we compare it with the single GSO (MaGSO) and BFOA (MaBFOA) on MaOPs. Table 5 shows the comparison results of MaGSO-MaBFOA, MaGSO, and MaBFOA.

It is obvious that the proposed coupling approach is better than the single GSO and BFOA on most test problems.

MaGSO-MaBFOA outperforms MaGSO on the WFG functions with all objectives. MaBFOA is slightly better than the MaGSO-MaBFOA on WFG3 and WFG9 with 5, 8, 10 objectives, while MaGSO-MaBFOA is superior to MaBFOA on other WFG functions with all objectives. The above results demonstrate that the coupling approach can effectively improve the performance of GSO and BFOA on MaOPs.

VI. CONCLUSION AND FUTURE WORK

The original GSO or BFOA shows some difficulties in solving MaOPs. To challenge these problems, a coupling approach on basis of GSO and BFOA is designed. The new approach is called MaGSO-MaBFOA, which employs two populations. Based on GSO and BFOA, two modified operators MaGSO and MaBFOA are designed for handling MaOPs. Each operator evolves a population to generate new solutions. By exchanging the search information of these two populations, an external archive is constructed. An archive updating method is proposed to update the external archive and store some best solutions in the evolutionary process. Simulation are guided on two benchmark sets (DTLZ and WFG) with 3 to 15 objectives.

For the DTLZ benchmark set, MaGSO-MaBFOA performs better than NSGA-III, KnEA, and GrEA on most test functions. MOEA/D-DE is slightly better than MaGSO-MaBFOA. Both HypE and MaGSO-MaBFOA obtain similar performance. For the WFG benchmark set, MaGSO-MaBFOA obtains better results than other five algorithms on the most test problems. Results confirm that the coupling strategy can effectively help GSO and BFOA to solve MaOPs. In the future work, the coupling approach will be applied to solve some practical problems [77]–[79].

REFERENCES

[1] P. Cortés, J. Muñuzuri, L. Onieva, and J. Guadix, "A discrete particle swarm optimisation algorithm to operate distributed energy generation networks efficiently," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 4, pp. 226–235, 2018.

[2] A. Ehsan and M. N. Dehkordi, "Dynamic data clustering by combining improved discrete artificial bee colony algorithm with fuzzy logic," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 3, pp. 164–172, 2018.

[3] C. Yang, J. Ji, J. Liu, and B. Yin, "Bacterial foraging optimization using novel chemotaxis and conjugation strategies," *Inf. Sci.*, vol. 363, pp. 72–95, Oct. 2016.

[4] W. Yu and J. Wang, "A new method to solve optimisation problems via fixed point of firefly algorithm," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 4, pp. 249–256, 2018.

[5] M. Zhang, H. Wang, Z. Cui, and J. Chen, "Hybrid multi-objective cuckoo search with dynamical local search," *Memetic Comput.*, vol. 10, no. 2, pp. 199–208, 2018.

[6] Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen, "A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber–physical systems," *J. Parallel Distrib. Comput.*, vol. 103, pp. 42–52, May 2017.

[7] Y. Zhou, M. Abdel-Baset, and M. Ismail, "An improved cuckoo search algorithm for integer programming problems," *Int. J. Comput. Sci. Math.*, vol. 9, no. 1, pp. 66–81, 2018.

[8] S. Dong and J. Zhou, "Hybrid glowworm swarm optimization for task scheduling in the cloud environment," *Eng. Optim.*, vol. 50, no. 6, pp. 949–964, Jun. 2018.

[9] Y. Feng and G. Yu, "Improving firefly algorithm using hybrid strategies," *Int. J. Comput. Sci. Math.*, vol. 9, no. 2, pp. 163–170, 2018.

[10] X. Cai, H. Wang, Z. Cui, J. Cai, Y. Xue, and L. Wang, "Bat algorithm with triangle-flipping strategy for numerical optimization," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 2, pp. 199–215, Feb. 2018.

[11] M. Bougherara, N. Nedjah, L. De M. Mourelle, R. Rahmoun, A. Sadok, and D. Bennouar, "IP assignment for efficient NoC-based system design using multi-objective particle swarm optimisation," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 4, pp. 203–213, 2018.

[12] R. Wang, F. Zhang, T. Zhang, and P. J. Fleming, "Cooperative co-evolution with improved differential grouping method for large-scale global optimisation," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 4, pp. 214–225, 2018.

[13] R. Zhou, G. Zhou, and Y. Zhou, "Solving large-scale 0-1 knapsack problem by the social-spider optimisation algorithm," *Int. J. Comput. Sci. Math.*, vol. 9, no. 5, pp. 433–441, 2018.

[14] Z. Cui, J. Zhang, Y. Chang, X. Cai, and W. Zhang, "Improved NSGA-III with selection-and-elimination operator," *Swarm Evol. Comput.*, vol. 49, pp. 23–33, Sep. 2019.

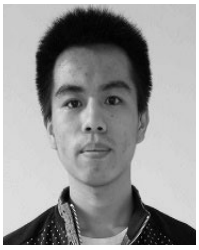
[15] P. Wang, F. Xue, H. Li, Z. Cui, L. Xei, and J. Chen, "A multi-objective DV-Hop localization algorithm based on NSGA-II in Internet of Things," *Mathematics*, vol. 7, no. 2, p. 184, 2019.

[16] L. Lv, T. Fan, Q. Li, Z. Sun, and L. Xu, "Object tracking with improved firefly algorithm," *Int. J. Comput. Sci. Math.*, vol. 9, no. 3, pp. 219–231, 2018.

[17] G.-G. Wang, S. Deb, and L. dos S. Coelho, "Earthworm optimisation algorithm: A bio-inspired metaheuristic algorithm for global optimisation problems," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 1, pp. 1–22, Jan. 2018.

[18] W. Arloff, K. R. B. Schmitt, and L. J. Venstrom, "A parameter estimation method for stiff ordinary differential equations using particle swarm optimisation," *Int. J. Comput. Sci. Math.*, vol. 9, no. 5, pp. 419–432, 2018.
[19] Z. Cui, F. Li, and W. Zhang, "Bat algorithm with principal component analysis," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 3, pp. 603–622, Mar. 2019.
[20] Y. Ren, Y. Sun, X. Jing, Z. Cui, and Z. Shi, "Adaptive makeup transfer via bat algorithm," *Mathematics*, vol. 7, no. 3, p. 273, 2019.
[21] X. Cai, X.-Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *Int. J. Bio-Inspired Comput.*, vol. 8, no. 4, pp. 205–214, 2016.
[22] Z. Cui, Y. Wang, Y. Cao, X. Cai, J. Zhang, W. Zhang, and J. Chen, "A pigeon-inspired optimization algorithm for many-objective optimization problems," *SCI. CHINA Inf. Sci.*, vol. 62, no. 7, 2019, Art. no. 070212.
[23] Z. Cui, Y. Cao, C. Yang, X. Cai, J. Cai, and J. Chen, "Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things," *Computing*, vol. 132, pp. 217–229, Oct. 2019.
[24] X. Cai, P. Wang, L. Du, Z. Cui, W. Zhang, and J. Chen, "Multi-objective 3-dimensional DV-hop localization algorithm with NSGA-II," *IEEE Sensors J.*, to be published. doi: 10.1109/JSEN.2019.2927733.
[25] S. Mohammadi, M. A. S. Monfared, and M. Bashiri, "An improved evolutionary algorithm for handling many-objective optimization problems," *Appl. Soft Comput.*, vol. 52, pp. 1239–1252, Mar. 2017.
[26] 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.
[27] Y. Xiang, Y. Zhou, M. Li, and Z. Chen, "A vector angle-based evolutionary algorithm for unconstrained many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 131–152, Jul. 2016.
[28] Q. Lin, S. Liu, K.-C. Wong, M. Gong, C. A. C. Coello, J. Chen, and J. Zhang, "A clustering-based evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 391–405, Jun. 2019.
[29] Y. Cao, Z. Ding, F. Xue, and X. Rong, "An improved twin support vector machine based on multi-objective cuckoo search for software defect prediction," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 4, pp. 282–291, 2018.
[30] B. Zhao, Y. Xue, B. Xu, T. Ma, and J. Liu, "Multi-objective classification based on NSGA-II," *Int. J. Comput. Sci. Math.*, vol. 9, no. 6, pp. 539–546, 2018.
[31] Y. Niu, Z. Tian, M. Zhang, X. Cai, and J. Li, "Adaptive two-SVM multi-objective cuckoo search algorithm for software defect prediction," *Int. J. Comput. Sci. Math.*, vol. 9, no. 6, pp. 547–554, 2018.
[32] K. E. Heraguemi, N. Kamel, and H. Drias, "Multi-objective bat algorithm for mining numerical association rules," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 4, pp. 239–248, 2018.
[33] Wang Jie and Y. Jiangjun, "A high-efficient multi-deme genetic algorithm with better load-balance," *Int. J. Comput. Sci. Math.*, vol. 9, no. 3, pp. 240–246, 2018.
[34] Y. Xiang, Y. Zhou, L. Tang, and Z. Chen, "A decomposition-based many-objective artificial bee colony algorithm," *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 287–300, Jan. 2019.
[35] X. Sun, Y. Chen, Y. Liu, and D. Gong, "Indicator-based set evolution particle swarm optimization for many-objective problems," *Soft Comput.*, vol. 20, no. 6, pp. 2219–2232, Jun. 2016.
[36] M. Han, Y. He, and D.-C. Zheng, "Reference-point-based particle swarm optimization algorithm for many- objective optimization," *Control Decis.*, vol. 32, no. 4, pp. 607–612, Apr. 2017.
[37] Z. Cui, X. Fei, X. Cai, C. Yang, G. G. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3187–3196, Jul. 2018.
[38] 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.
[39] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 284–302, Apr. 2009.
[40] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.
[41] 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.
[42] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 348–365, Jun. 2014.
[43] K. C. Tan, Y. J. Yang, and C. K. Goh, "A distributed cooperative coevolutionary algorithm for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 527–549, Oct. 2006.
[44] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired coevolutionary algorithms for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 4, pp. 474–494, Aug. 2013.
[45] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "An efficient approach to nondominated sorting for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 201–213, Apr. 2015.
[46] X. Zhang, Y. Tian, and Y. Jin, "Approximate non-dominated sorting for evolutionary many-objective optimization," *Inf. Sci.*, vol. 369, pp. 14–33, Nov. 2016.
[47] 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.
[48] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.
[49] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 445–460, Jun. 2015.
[50] 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.
[51] X. Cai, Z. Mei, and Z. Fan, "A decomposition-based many-objective evolutionary algorithm with two types of adjustments for direction vectors," *IEEE Trans. Cybern.*, vol. 48, no. 8, pp. 2335–2348, Aug. 2018.
[52] 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.
[53] E. Zitzler, and S. Kunzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving From Nature* (Lecture Notes in Computer Science), vol. 3242. Berlin, Germany: Springer-Verlag, 2004, pp. 832–842.
[54] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello, "Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 4, pp. 504–522, Aug. 2012.
[55] H. Ishibuchi, R. Imada, Y. Setoguchi, and Y. Nojima, "Reference point specification in inverted generational distance for triangular linear Pareto front," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 961–975, Dec. 2018.
[56] A. Menchaca-Mendez and C. A. C. Coello, "GD-MOEA: A new multi-objective evolutionary algorithm based on the generational distance indicator," presented at the Int. Conf. Evol. Multi-Criterion Optim., Mar. 2015, pp. 156–170.
[57] A. Menchaca-Mendez and C. A. C. Coello, "GDE-MOEA: A new MOEA based on the generational distance indicator and ϵ-dominance," presented at the Evol. Comput., May 2015, pp. 947–955.
[58] A. Menchaca-Mendez and C. A. C. Coello, "MD-MOEA : A new MOEA based on the maximin fitness function and Euclidean distances between solutions," presented at the Evol. Comput., Jul. 2014, pp. 2148–2155.
[59] A. Menchaca-Mendez and C. A. C. Coello, "MH-MOEA: A new multi-objective evolutionary algorithm based on the maximin fitness function and the hypervolume indicator," presented at the Int. Conf. Parallel Problem Solving Nature, Sep. 2014, pp. 652–661.
[60] H. Wang and X. Yao, "Corner sort for Pareto-based many-objective optimization," *IEEE Trans. on*, vol. 44, no. 1, pp. 92–102, Jan. 2014.
[61] 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. 2015.
[62] K. S. Kumar and T. Jayabarathi, "Power system reconfiguration and loss minimization for an distribution systems using bacterial foraging optimization algorithm," *Int. J. Electr. Power Energy Syst.*, vol. 36, no. 1, pp. 13–17, Mar. 2012.
[63] H. Shen, Y. Zhu, X. Zhou, H. Guo, and C. Chang, "Bacterial foraging optimization algorithm with particle swarm optimization strategy for global numerical optimization," presented at the Acm/SIGEVO Summit Genetic Evol. Comput., Jun. 2009, pp. 497–504.

[64] I. A. A. Al-Hadi, S. Z. M. Hashim, and S. M. H. Shamsuddin, "Bacterial foraging optimization algorithm for neural network learning enhancement," presented at the Int. Conf. Hybrid Intell. Syst., Dec. 2011, pp. 200–205.
[65] K. N. Krishnanand and D. Ghose, "A glowworm swarm optimization based multi-robot system for signal source localization," *Stud. Comput. Intell.*, vol. 177, no. 177, pp. 49–68, 2009.
[66] Y. Zhou, Z. Huang, and H. Liu, "Discrete glowworm swarm optimization algorithm for TSP problem," *Acta Electronica Sinica*, vol. 40, no. 6, pp. 1164–1170, 2012.
[67] Q. F. Luo, Q. Q. Gong, and Y. Q. Zhou, "Artificial glowworm swarm optimization algorithm for solving multi-objective constrained optimization," *Appl. Mech. Mater.*, vols. 220–223, pp. 2393–2397, Nov. 2012.
[68] W.-H. Liao, Y. Kao, and Y.-S. Li, "A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12180–12188, 2011.
[69] K. N. Krishnanand and D. Ghose, "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions," *Swarm Intell.*, vol. 3, no. 2, pp. 87–124, Jun. 2009.
[70] G.-Q. Zeng, J. Chen, L.-M. Li, M.-R. Chen, L. Wu, Y.-X. Dai, and C.-W. Zheng, "An improved multi-objective population-based extremal optimization algorithm with polynomial mutation," *Inf. Sci.*, vol. 330, pp. 49–73, Feb. 2016.
[71] Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, C. A. C. Coello, K.-C. Wong, and J. Zhang, "Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 32–46, Feb. 2018.
[72] J. Zhang, F. Xue, X. Cai, Z. Cui, Y. Chang, W. Zhang, and W. Li, "Privacy protection based on many-objective optimization algorithm," *Concurrency Comput., Pract. Exper.*, May 2019. doi: 10.1002/cpe.5342.
[73] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.
[74] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 4, pp. 495–511, Aug. 2013.
[75] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.
[76] S. Jiang, Y.-S. Ong, J. Zhang, and L. Feng, "Consistencies and contradictions of performance metrics in multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2391–2404, Dec. 2014.
[77] Z. Cui, L. Du, P. Wang, X. Cai, and W. Zhang, "Malicious code detection based on CNNs and multi-objective algorithm," *J. Parallel Distrib. Comput.*, vol. 129, pp. 50–58, Jul. 2019.
[78] P. Wang, J. Huang, Z. Cui, L. Xie, and J. Chen, "A Gaussian error correction multi-objective positioning model with NSGA-II," *Concurrency Comput., Pract. Exper.*, Aug. 2019. doi: 10.1002/cpe.5464.
[79] G. G. Wang, X. Cai, Z. Cui, G. Min, and J. Chen, "High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm," *IEEE Trans. Emerg. Topics Comput.*, to be published. doi: 10.1109/TETC.2017.2703784.

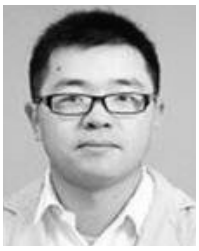


JIANGJIANG ZHANG is currently pursuing the master's degree in computer science and technology with the Taiyuan University of Science and Technology, China. His research interests include computational intelligence and combinatorial optimization.

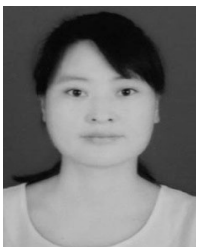
ZHIHUA CUI received the Ph.D. degree in control theory and engineering from Xi'an Jiaotong University, Xi'an, China, in 2008.

He is currently a Professor of Computer Science and Technology and the Director of the Complex System and Computational Intelligence Laboratory with the Taiyuan University of Science and Technology, Taiyuan, China. He is also the Editor-in-Chief of the *International Journal of Bio-inspired Computation*. His research interests include computational intelligence, stochastic algorithm, and combinatorial optimization.

YECHUANG WANG is currently pursuing the M.A. degree in Computer Science and Technology with the Taiyuan University of Science and Technology, China. His research interests include computational intelligence and combinatorial optimization.

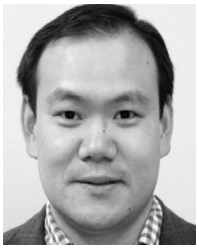


HUI WANG received the B.Sc. and M.Sc. degrees in computer science from the China University of Geosciences, Wuhan, China, in 2005 and 2008, respectively, and the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, in 2011. He is currently a Professor with the Nanchang Institute of Technology, Nanchang, China. He has published more than 50 international journal/conference articles. His research interests include evolutionary optimization, swarm intelligence, and their applications.



XINGJUAN CAI received the Ph.D. degree in control science and engineering from Tongji University, China, in 2017.

She is currently an Associate Professor with the School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, China. Her research interest includes bio-inspired computation and application.



JINJUN CHEN (SM'13) received the Ph.D. degree in computer science and software engineering from Swinburne University and Technology, Melbourne, VIC, Australia.

He is currently a Professor with the Department of Computer Science and Software Engineering, Swinburne University of Technology. He is also the Director of the Laboratory of Cloud Computing and Distributed Systems. His research interests include cloud computing, big data, workflow management, privacy and security, and various related research topics.

WUZHAO LI received the M.Sc. degree in computer science from Jiangnan University, in 2006, and the Ph.D. degree in control science and engineering from Tongji University, China, in 2018. His research interests include computational intelligence, stochastic algorithm, and combinatorial optimization.