Information Sciences 522 (2020) 1–16

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Enhancing MOEA/D with information feedback models for large-scale many-objective optimization

Yin Zhang[a,b], Gai-Ge Wang[b,*], Keqin Li[c], Wei-Chang Yeh[d], Muwei Jian[e], Junyu Dong[a,b]

[a] School of Computer Science and Technology, Ocean University of China, 266100 Qingdao, China
[b] Department of Computer Science and Technology, Ocean University of China, 266100 Qingdao, China
[c] Department of Computer Science, State University of New York, New Paltz, New York, 12561, USA
[d] Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu, Taiwan
[e] School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan, China

ARTICLE INFO

Article history:
Received 24 July 2019
Revised 21 February 2020
Accepted 24 February 2020
Available online 26 February 2020

Keywords:
Benchmark
Decomposition
Evolutionary algorithms
Information feedback models
Many-objective
Multi-objective 0-1 knapsack problem

ABSTRACT

A multi-objective evolutionary algorithm based on decomposition (MOEA/D) is a classic decomposition-based multi-objective optimization algorithm. In the standard MOEA/D algorithm, the update process of individuals is a forward search process without using the information of previous individuals. However, there is a lot of useful information in the previous iteration. Information Feedback Models (IFM) is a new strategy which can incorporate the information from previous iteration into the updating process. Therefore, this paper proposes a MOEA/D algorithm based on information feedback model, called MOEA/D-IFM. According to the different information feedback models, this paper proposes six variants of MOEA/D, and these algorithms can be divided into two categories according to the way of selecting individuals whether it is random or fixed. At the same time, a new selection strategy has been introduced to further improve the performance of MOEA/D-IFM. The experiments were carried out in four aspects. MOEA/D-IFM were compared with other state-of-the-art multi-objective evolutionary algorithms using CEC 2018 problems in two aspects. The best one of the six improved algorithms was chosen to test on large-scale many-objective problems. In addition, we also use MOEA/D-IFM to solve multi-objective backpack problems.

1. Introduction

Multi-objective problems are widely used in real life, so it is very necessary and practical to solve multi-objective problems. The tricky part of dealing with multi-objective problems is that different objectives contradict with each other. It makes difficult for us to get the optimal solutions. Multi-objective problems need to get the minimum or maximum value of each objective at the same time. It is very difficult to solve multi-objective problems using traditional mathematical methods. Evolutionary algorithms can perform well on multi-objective problems.

* Corresponding author.
E-mail addresses: zhangyin@stu.ouc.edu.cn (Y. Zhang), gaigewang@gmail.com, gaigewang@163.com (G.-G. Wang), lik@newpaltz.edu (K. Li), yeh@ieee.org (W.-C. Yeh), jianmuweihk@163.com (M. Jian), dongjunyu@ouc.edu.cn (J. Dong).

https://doi.org/10.1016/j.ins.2020.02.066
0020-0255/

Pareto dominance is one of the efficient methods to solve multi-objective problems. If x dominates y, then $\forall i \in \{1, 2, ..., m\}$, $f_i(x) \le f_j(y)$, $\exists j \in \{1, 2, ..., m\}$, $f_j(x) < f_j(y)$. When there is a solution that is not subject to any other solutions, it becomes a non-dominated solution, and these mutually non-dominated solutions are integrated into a Pareto-optimal Set (PS). The set of objective function values corresponding to all solutions in the Pareto-optimal Set called Pareto Front (PF). One of the most classic algorithms based on the Pareto dominance relationship is a fast and elitist multi-objective genetic algorithm (NSGA-II), proposed by Deb et al. in 2002 [7]. NSGA-II is an improved NSGA [23], which is layered according to the dominance relationship between individuals. Each layer has a virtual fitness value. NSGA-II mainly includes fast non-dominated sorting, combining the solution generated by cross and mutation operators with the solution of the previous generation, and then layering with non-dominated sorting. Based on NSGA-II, Deb et al. [8] proposed an evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, called NSGA-III, and applied NSGA-III to a number of many-objective test problems. Then Deb et al. [13] extended NSGA-III to solve generic constrained many-objective optimization problems. Arash et al. [19] proposed an open source C++ genetic algorithm library called openGA, which allows users to freely design custom solutions.

The multi-objective artificial bee colony algorithm (MOABC) [1], the employer bees adjusted the trajectory according to the non-dominated relationship. Improved strength Pareto evolutionary algorithm (SPEAII) [50] used an external archive to hold better solutions and an improved fitness allocation method. However, since the multi-objective algorithm based on Pareto dominance has a sharp decline in performance when the number of objectives increases, many improvement strategies have emerged. Zhang et al. [47] proposed a knee point-driven evolutionary algorithm (KnEA). Xiang et al. [36] proposed a vector angle-based evolutionary algorithm for unconstrained many-objective optimization. Chen et al. [2] proposed a new local search-based multi-objective optimization algorithm (NSLS).

Decomposition is also a very good way to solve multi-objective problems, which is to decompose multi-objective problems into single objective problems. By simulating some biological behaviors, the whole population is continuously evolved, and finally an approximate PF is obtained. Cheng et al. [3] proposed a reference vector guided evolutionary algorithm (RVEA) for many-objective optimization in 2016. Wang et al. [34] proposed an adaptive replacement strategies for MOEA/D. Zhang et al. [43] proposed a method of adjusting weight vectors in MOEA/D for bi-objective optimization problems with discontinuous Pareto fronts. Zhang et al. [44] proposed a self-organizing multi-objective evolutionary algorithm.

Although the evolutionary algorithms have performed well on many problems, they still have some defects, so many improvement strategies have been proposed. Among them, the information feedback models (IFM) [29] are included. The information feedback models have achieved good results in the single objective evolutionary algorithm. Because at each iterative update of the algorithm, there may be a lot of useful information. We usually do not consider this information when updating the current individuals, so it is possible to waste a lot of useful information. As far as we know, this model has not been applied to solve multi-objective problems. So, we propose a multi-objective optimization algorithm based on information feedback models, called MOEA/D-IFM. Unlike most optimization algorithms, MOEA/D-IFM takes the importance of this information into account and will preserve the information from previous populations. There are two ways to select individuals: fixed and random. The individuals selected from the historical information and the individual obtained from the basic algorithm are used for the update of the current individual. The fitness value of individual is used as determinants of individual weight. Six different information feedback models were used in this paper. The experiments were carried out in four aspects. On one hand, the six improved algorithms were compared with multi-population-driven evolutionary algorithm (AMPDEA) [6] and decomposition-based algorithms using pareto adaptive scalarizing methods (MOEA/D-PaS) [32] on CEC 2018 problems. On the other hand, the best one of six improved algorithms was chosen to test on large-scale many-objective problems. MOEA/D-IFM also made a comparison with CVEA3 [42] on CEC 2018 problems. In addition, we use MOEA/D-IFM to solve multi-objective backpack problems. The experimental results show MOEA/D-IFM outperformed some state-of-the-art multi-objective algorithms.

The rest of this paper is organized as follows. In Section 2, we discuss the research of related work. Section 3 introduces the basic principle of MOEA/D and two decomposition approaches in multi-objective optimization problems. Section 3 also has an introduction to the information feedback model. In Section 4, the proposed MOEA/D-IFM algorithm is described in detail. The experimental results are given in Section 5. Section 6 gives a general summary of the paper.

2. Related work

Most of the problems in real life are multi-objective problems. We should have a comprehensive consideration when dealing with multi-objective problems, such as production cost, time, and technology. The multi-objective problems are difficult to solve because it cannot get the optimal solution for each objective at the same time. When solving multi-objective problems, we are required to weigh the pros and cons according to the actual situations. This is the famous no-free lunch theorem [35]. Multi-objective evolutionary algorithms have been applied to solve many practical problems. Mohammadi et al. [20] proposed a multi-objective genetic algorithm to obtain the best model predictive control (MPC) weighting in driving simulators. This method can find the best tune of the MPC cost function weights and reduce the user burden for weight tuning while receiving feedback from the user satisfaction.

MOEA/D has a good performance on solving multi-objective problems, and researchers conducted a deeper exploration of it. In MOEA/D, the choice of reference points is very important. In the standard MOEA/D, the reference point is usually the optimal value for each sub-objective. For the study of reference points, Wang et al. [33] discussed the influence of reference points on the MOEA/D, and divided the effects of reference points into three categories, which were pessimistic, optimistic, and dynamic. For bi-objective optimization problems with discontinuous Pareto fronts, Zhang et al. [43] proposed a method to adjust the weight vector. This method was to find the weight vector that is required to be adjusted, then these weight vectors were divided into several groups and the number of ideal points for each weight vector was calculated. Finally, the solution set was updated by linear interpolation method. In order to improve the diversity and convergence of the solution set, Zheng et al. [49] proposed a new hybrid decomposition method, which combined the weighted sum method with the Tchebycheff to improve NSGA-II performance. Ke et al. [14] integrated ACO into MOEA/D. The ants were divided into several groups, and each ant solved a sub-problem. The ants recorded the historical optimal solution of the sub-problems. Wang et al. [34] proposed a global replacement scheme which assigned a new solution to its most suitable sub-problems, called adaptive replacement strategies for MOEA/D. Tan et al. [25] proposed a modification to MOEA/D-DE for multi-objective optimization problems with complicated Pareto sets. A uniform design was applied to the vector of the coefficient of aggregation so that the sub-problems of the decomposition can be equally distributed. Ma et al. [17] proposed a MOEA/D algorithm based on opposite learning that is a machine learning method. Convergence speed of MOEA/D can be improved by adding opposite learning. For the Tchebycheff decomposition method, the weight vector and the optimal sub-problem solution are nonlinear. Ma et al. [18] proposed a uniform decomposition method. MOEA/D performs the same processing for all sub-problems. However, in general, the computational resources are equally allocated for each sub-problem. Wang et al. [31] proposed a new resource allocation strategy based on the relationship between sub-problems, a probability vector is established to guide the optimization of sub-problems.

There are still many in-depth studies on the MOEA/D by researchers, but most of these algorithms only focus on the reference point of the MOEA/D, the improvement of the decomposition method, or the addition of some other strategies. None of these methods use the population information before, but this information is very likely to be very useful.

Information feedback models have been applied to many single objective optimization algorithms, such as PSO [11], ACO [9], bat algorithm (BA) [38], biogeography-based optimization (BBO) [22], cuckoo search (CS) [37], differential evolution (DE) [24], krill herd (KH) [12,26], and monarch butterfly optimization (MBO) [27]. And the algorithms that have been added to the information feedback models have been experimentally proven to improve the performance of the algorithm itself. These optimization algorithms that were incorporated into the information feedback model have better performance than their basic ones. As far as we know, the information feedback models have not been added to the multi-objective algorithms, so we consider adding the information feedback model to the multi-objective algorithms. The information feedback models are based on the fitness function. We use the information in the previous iterations to update the current individual, and the fitness functions are used to balance the current individual and the previous information.

In this paper, we have introduced information feedback models to MOEA/D to make up for this gap, and then MOEA/D-IFM was proposed. MOEA/D-IFM improves the performance of MOEA/D by reusing the information before. Six different information feedback models were added to the basic MOEA/D. The improved algorithm has better performance in all comparative multi-objective optimization algorithms. Specific details will be explained later in this paper.

3. Preliminaries

In this section, we will introduce the basic MOEA/D algorithm and the two decomposition methods of MOEA/D. In addition, we will introduce the fundamental of the information feedback models, because our work is based on the six formation feedback models. The detailed introduction to these six models will be given before proposing the algorithm.

3.1. MOEA/D

Multi-objective optimization problems can be stated as:

$$\begin{aligned} &\min f(\mathrm{x}) = (f_1(x), f_2(x), \ldots, f_m(x)) \\ &\text{subject to } x \in \Omega, f \in R^m \end{aligned} \tag{1}$$

where $x = \{x_1, x_2, \ldots, x_n\} \in \Omega$ is a decision vector, Ω is the decision space, and n is the dimensions of the decision vector. R^m is the objective space, and m is the number of objectives.

The main idea of MOEA/D is relatively simple, which is to decompose a multi-objective problem into multiple single objective problems. After getting multiple single objective problems, we can process these single-objective problems at the same time, which greatly reduces the computational complexity. This is also a major advantage of a multi-objective algorithm based on decomposition. In this process, the decomposition method is used to decompose multi-objective problems. There are three common decomposition methods in multi-objective optimization. They are weighted sum approach, Tchebycheff approach, and Penalty-based Boundary Intersection (PBI) approach [21]. Pareto optimal vectors are not always obtained by weighted sum approach when the PF are nonconcave [23]. In this paper, the penalty-based boundary intersection (PBI) approach are used.

3.1.1. Tchebycheff approach

The method is as follows:

$$\min\ g^{te}(x|\lambda, z^*) = \max_{1\le i\le m} \{\lambda_i (f_i(x) - z_i^*)\} \quad s.t.\ x \in \Omega \tag{2}$$

where $\lambda=(\lambda_1, \lambda_2, ..., \lambda_m)$ is the weight vector, $z^* = (z_1^*, z_2^*, ..., z_m^*)$ is the reference point, the reference point is the coordinate composed of the minimum value of each objective. The goal of optimization is to find individuals on the PF. Tchebycheff approach can continuously approach the PF by calculating the distance from the reference points. When we take a different value for λ, we can get other solutions on PF.

3.1.2. Penalty-based Boundary Intersection (PBI) approach

The Boundary Intersection (BI) approach can be given as follows:

$$\min\ g^{bi}(x|\lambda, z^*) = d \quad s.t. z^* - f(x) = d\lambda, \quad x \in \Omega \tag{3}$$

where d is a scalar, indicating the distance between $f(x)$ and λ. The meaning of the constraint is to ensure $f(x)$ and λ located at the same line. This limitation is not realistic [45], so it must be improved. As an improvement to the previous algorithm, a penalty function is introduced in the constraint.

$$\begin{aligned} \min\ g^{bip}(x|\lambda, z^*) &= d_1 + d_2\theta \quad s.t. x \in \Omega \\ d_1 &= \frac{\left\| (z^* - f(x))^T \lambda \right\|}{\|\lambda\|} \\ d_2 &= \|f(x) - (z^* - d_1\lambda)\| \end{aligned} \tag{4}$$

where θ is a pre-set parameter, and usually its value is 0.5. d_1 is a scalar, indicating the distance that is projected on the direction λ, d_2 is a value indicating punishment, which indicates the distance that $f(x)$ is projected in the vertical direction of λ. The farther away from λ, the larger the penalty value obtained. Compared with the Tchebycheff approach, the PBI obtains a more uniform distribution of solutions when the objective number is higher. If the generated solution and weight vector are not in the same direction, then the solution will be punished.

3.2. Information feedback models

Information feedback model is a strategy for improving the performance of metaheuristic algorithms. Its central idea is to use the information in the previous iteration to update the current generated solution. There are two ways to select individuals in the previous iteration, which are random selection and fixed selection. The number of iterations for selecting individuals can be any numbers k ($k<n$) before the current number of iterations. For the simple example, we set the value of k to 3. The feedback model that selects individuals in fixed manner is called M1, and the feedback model that selects individuals in a random manner is called M2

3.2.1. Fixed selection

In this model, we select individuals in a fixed way. This fixed way is to select the same position as the individual at the current iteration to select the individual from the previous iteration. Then, the information of the individual in the previous iteration is used to update the current individual. θ_1, θ_2, θ_3, and θ_4 are the weight vectors, $\theta_i > 0, \sum_{i=1}^{m} \theta_i = 1$.

(1) M1-1: This is the simplest model, using the individual information from the last previous iteration. The current number of iterations is t, x_i^t is the i-th individual at the current iteration, and the corresponding to fitness value is f_i^t, y_i is the individual obtained in the basic algorithm, and its fitness value is F^t.

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_i^t + \theta_2 y_i^{t+1} \\ \theta_1 &= \frac{f_i^t}{F^{t+1} + f_i^t}, \\ \theta_2 &= \frac{F^{t+1}}{F^{t+1} + f_i^t}. \end{aligned} \tag{5}$$

(2) M1-2: In this model, we select two individuals from two previous iterations which is t and t-1. We can get the following update equation:

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_i^{t-1} + \theta_2 x_i^t + \theta_3 y_i^{t+1} \\ \theta_1 &= \frac{1}{2} \bullet \frac{f_i^{t-1} + F^{t+1}}{F^{t+1} + f_i^t + f_i^{t-1}} \\ \theta_2 &= \frac{1}{2} \bullet \frac{f_i^t + F^{t+1}}{F^{t+1} + f_i^t + f_i^{t-1}} \\ \theta_3 &= \frac{1}{2} \bullet \frac{f_i^t + f_i^{t-1}}{F^{t+1} + f_i^t + f_i^{t-1}} \end{aligned} \tag{6}$$

where x_i^{t-1} is the individual selected from t-1 iterations, the position is the same as the current individual x_i^t, and $f_i^{t-1}$is the fitness value corresponding to x_i^{t-1}.

(3) M1-3: In this model, we select three individuals from three previous iterations which is *t*, *t*-1, and *t*-2. We can get the following update equation:

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_i^{t-2} + \theta_2 x_i^{t-1} + \theta_3 x_i^t + \theta_4 y_i^{t+1} \\ \theta_1 &= \frac{1}{3} \bullet \frac{f_i^{t-1} + f_i^t + F^{t+1}}{f_i^{t-2} + f_i^{t-1} + f_i^t + F^{t+1}} \\ \theta_2 &= \frac{1}{3} \bullet \frac{f_i^{t-2} + f_i^t + F^{t+1}}{f_i^{t-2} + f_i^{t-1} + f_i^t + F^{t+1}} \\ \theta_3 &= \frac{1}{3} \bullet \frac{f_i^{t-2} + f_i^{t-1} + F^{t+1}}{f_i^{t-2} + f_i^{t-1} + f_i^t + F^{t+1}} \\ \theta_4 &= \frac{1}{3} \bullet \frac{f_i^{t-2} + f_i^{t-1} + f_i^t}{f_i^{t-2} + f_i^{t-1} + f_i^t + F^{t+1}} \end{aligned} \tag{7}$$

Here we selected three individuals x_i^t, x_i^{t-1}, and x_i^{t-2} to update the current solution, and f_i^t, f_i^{t-1}, and f_i^{t-2} are their fitness values, respectively.

3.2.2. Random selection

In this model, we use a random approach to select individuals in previous iterations. In this process, we use the method of generating random numbers to determine the location of the selected individual. Similar to the fixed selection method, we only list three models here.

(1) M2-1: It is obvious that in the random selection model we use numbers randomly generated between 1 and *N* to determine the location of the selected individual in the population.

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_r^t + \theta_2 y_i^{t+1} \\ \theta_1 &= \frac{F^{t+1}}{f_r^t + F^{t+1}} \\ \theta_2 &= \frac{f_r^t}{f_r^t + F^{t+1}} \end{aligned} \tag{8}$$

where *r* is the random number between 1 and *N*, x_r^t is the *r*-th individual at iteration *t*, and f_r^t is its fitness value.

(2) M2-2: We randomly selected two individuals from the two previous iterations which is *t* and *t*-1. We can get the following update equation.

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_{r_1}^{t-1} + \theta_2 x_{r_2}^t + \theta_3 y_i^{t+1} \\ \theta_1 &= \frac{1}{2} \bullet \frac{f_{r_2}^t + F^{t+1}}{f_{r_1}^{t-1} + f_{r_2}^t + F^{t+1}} \\ \theta_2 &= \frac{1}{2} \bullet \frac{f_{r_1}^{t-1} + F^{t+1}}{f_{r_1}^{t-1} + f_{r_2}^t + F^{t+1}} \\ \theta_3 &= \frac{1}{2} \bullet \frac{f_{r_1}^{t-1} + f_{r_2}^t}{f_{r_1}^{t-1} + f_{r_2}^t + F^{t+1}} \end{aligned} \tag{9}$$

where r_1 and r_2 are two random numbers between 1 and *N*, $x_{r_1}^{t-1}$ and $x_{r_2}^t$ are individuals selected from iterations *t*-1 and *t*, respectively, $f_{r_1}^{t-1}$ and $f_{r_2}^t$ are their corresponding fitness values.

(3) M2-3: Similar to the above two models, three individuals were selected from *t*, *t*-1, *t*-2 iterations. We can get the following update equation.

$$\begin{aligned} x_i^{t+1} &= \theta_1 x_{r_1}^{t-2} + \theta_2 x_{r_2}^{t-1} + \theta_3 x_{r_3}^t + \theta_4 y_i^{t+1} \\ \theta_1 &= \frac{1}{3} \bullet \frac{f_{r_2}^{t-1} + f_{r_3}^t + F^{t+1}}{f_{r_1}^{t-2} + f_{r_2}^{t-1} + f_{r_3}^t + F^{t+1}} \\ \theta_2 &= \frac{1}{3} \bullet \frac{f_{r_1}^{t-2} + f_{r_3}^t + F^{t+1}}{f_{r_1}^{t-2} + f_{r_2}^{t-1} + f_{r_3}^t + F^{t+1}} \\ \theta_3 &= \frac{1}{3} \bullet \frac{f_{r_1}^{t-2} + f_{r_2}^{t-1} + F^{t+1}}{f_{r_1}^{t-2} + f_{r_2}^{t-1} + f_{r_3}^t + F^{t+1}} \\ \theta_4 &= \frac{1}{3} \bullet \frac{f_{r_1}^{t-2} + f_{r_2}^{t-1} + f_{r_3}^t}{f_{r_1}^{t-2} + f_{r_2}^{t-1} + f_{r_3}^t + F^{t+1}} \end{aligned} \tag{10}$$

where r_1, r_2, and r_3 are the random numbers between 1 and N, $x_{r_1}^{t-2}$, $x_{r_2}^{t-1}$, and $x_{r_3}^{t}$ are individuals selected from t-2, t-1, and t iterations, respectively, $f_{r_1}^{t-2}$, $f_{r_2}^{t-1}$, and $f_{r_3}^{t}$ are their fitness values.

4. MOEA/D-IFM

In this section, we incorporated the information feedback model into the MOEA/D. We first give the general framework of the proposed algorithm, and a detailed explanation of the algorithm was provided later.

4.1. General framework of MOEA/D-IFM

In the initialization phase, a population P is generated. The decomposition method used in the algorithm is Tchebycheff approach. In the standard MOEA/D algorithm, the number of weight vectors is the same with the population size, then the number of weight vectors is N. Each set of weight vectors transforms the multi-objective optimization problem into multiple single-objective optimization problems. Each weight vector corresponds to a single-objective optimization problem. The solutions on adjacent weight vectors are similar. Weight vectors have neighbors. Assume that the number of neighbors is T. The neighbors of each weight vector involves the T points around it. These T points are used to generate a new solution. After each generation of new population is produced, it is necessary to replace the solution in the neighborhood. First, we calculate Euclidean distance between the weight vectors and find the nearest T weight vectors for each weight vector form the neighbor $B=\{i_1, i_2, \ldots, i_T\}$ of this weight vector. External population is used to save non-dominated solutions during the search process.

Step 1) Initialization: Initialize parameters, generate the weight vectors $\lambda=(\lambda_1, \lambda_2, \ldots, \lambda_m)$. Calculate the neighbor $B=\{i_1, i_2, \ldots, i_T\}$. Generate an initial random population $P=\{x_1, x_2, \ldots, x_N\}$, take the minimum (maximum) value of each single objective value as a reference point $z^* = (z_1^*, z_2^*, \ldots, z_m^*)$.
Step 2) Search: For each solution x_i ($i=1, 2, \ldots, N$), randomly select two positions from B to generate a new solution y_i.
Step 2.1) Selecting individuals for updating from the saved information according to one of the six models.
Step 2.2) Update reference point $z^* = (z_1^*, z_2^*, \ldots, z_m^*)$ and B.
Step 3) Update EP. If the newly generated solution y is not dominated by any solution in EP, and remove solutions dominated by y.
Step 4) Stopping criterion: If the stopping condition is met, stop and output EP. Otherwise, go to Step 2).

4.2. Operators

Simulated binary crossover is a crossover operator commonly used in evolutionary algorithms. Where p_1 and p_2 are two parents, c_1 and c_2 are two children, β is spread factor, and u is a random number in the range [0,1]. β has two calculation equations that determine which equation will be used based on the value of the random number u.

$$\begin{aligned} c_1 &= \frac{1}{2} \times (p_2 + p_1) - \frac{1}{2} \times \beta(p_2 - p_1) \\ c_2 &= \frac{1}{2} \times (p_2 + p_1) + \frac{1}{2} \times \beta(p_2 - p_1) \\ \beta &= \begin{cases} (2 \times u)^{\frac{1}{\eta+1}} u \le 0.5 \\ \left(\frac{1}{2\times(1-u)}\right)^{\frac{1}{\eta+1}} u > 0.5 \end{cases} \end{aligned} \tag{11}$$

After the crossover operation, the individual undergoes a polynomial mutation to generate a new individual, u is a random number in the range [0, 1], η is a distribution index, u_k and l_k are the upper and lower bounds of the k-th decision variable.

$$\begin{aligned} c &= \begin{cases} \left[2 \times u + (1 - 2 \times u)(1-\delta_1)^{\eta+1}\right]^{\frac{1}{\eta+1}} - 1 u \le 0.5 \\ 1 - \left[2 \times (1-u) + 2 \times (u - 0.5)(1-\delta_2)^{\eta+1}\right]^{\frac{1}{\eta+1}} u > 0.5 \end{cases} \\ \delta_1 &= \frac{(p - l_k)}{u_k - l_k}, \delta_2 = \frac{u_k - p}{u_k - l_k} \end{aligned} \tag{12}$$

In Section 2, we introduced the basic MOEA/D algorithm and information feedback model. In this section, we will explain in detail how to integrate the information feedback model into the MOEA/D.

4.3. Selection strategy

In this paper, the selection strategy we used was a neighbor selection strategy [42]. ME is a mutual evaluation matrix. ME is calculated by the relationship between each solution and other solutions. The specific calculation method is given as follows:

$$ME_{\mathrm{i}} = \min_{j \in I \setminus i} me_{ij}, \mathrm{I} = \{1, 2, ..., N^*\}$$
$$me_{ij} = \max_{p} f_p^j / f_p^i \tag{13}$$

where f_p^j and f_p^i are two different solutions to the p-th subproblem, and $i \neq j$. After calculating the ME value of each solution, we will retain $N/2$ solutions with larger ME values. It should be noted that each time the solution with the smallest ME value is removed, the ME value of the solution associated with it is recalculated. Then, a parallel distance d_{ij} between any two solutions is calculated, and the two solutions with the smallest distance are compared, and the one with the smaller ME value is removed. The parallel distance is calculated as follows:

$$d_{ij} = \left[\sum_{p=1}^{m} \left(\bar{f}_p^i - \bar{f}_p^j\right)^2 - \left(\sum_{p=1}^{m} (\bar{f}_p^i - \bar{f}_p^j) \right)^2 /m)^{1/2} \right] \tag{14}$$

$$\bar{f}_p = \frac{f_p - f_p^{\min}}{f_p^{\max} - f_p^{\min}}, p = 1, 2, ..., m. \tag{15}$$

where $\bar{f}_p^i$ and $\bar{f}_p^j$ are the normalized values of the different solutions of the p sub-problem, d_{ij} is calculated by Eq. (15), the solutions are all distributed within the range of [0, 1]. $f_p^{\max}$ and $f_p^{\min}$ are the maximum and minimum solutions of p-th sub-problem, respectively.

4.4. MOEA/D1-MOEA/D6

Because the information feedback model will select individuals from the previous iteration for the current update, information feedback models can only be used when the number of iterations is greater than two. First, we initialize the population and weight vectors. After the initialization is complete, we will calculate the Euclidean distance between the weight vectors. For each weight vector, we select the nearest T weight vectors according to the Euclidean distance and store the indexes of the T weight vectors in the neighbor $B = \{i_1, i_2, ..., i_T\}$. Each weight vector corresponds to a sub-objective. We randomly selected two locations from B to generate new individuals as parents. The crossover and mutation operations we use here are simulated binary crossover and polynomial mutation. Because the information feedback model needs to use the information in the previous iteration, the information before the current iteration is saved to the historical population.

We keep all the generations before the current iteration in the historical population, and then we select individuals from the historical population according to the model we choose. We detailed the six feedback models in Section 3. In this paper, we add six information feedback models which were introduced in Section 3 to the MOEA/D algorithm. These three models include M1-1, M1-2, and M1-3 with fixed selection. The improved algorithms for adding model M1-1, M1-2 and M1-3 are called MOEA/D1, MOEA/D2, and MOEA/D3. In addition, there are M2-1, M2-2, and M2-3 with random selection. Similarly, the three improved algorithms are called MOEA/D4, MOEA/D5, and MOEA/D6, respectively.

5. Experimentats

In this Section, we will test the performance of our proposed algorithm using the benchmark functions from CEC 2018 competition on many-objective optimization and test problems for large-scale many-objective optimization problems. The experiments are carried out in four aspects. On one hand, we compare the six improved algorithms with AMPDEA [6] and MOEA/D-PaS [32] on the many-objective benchmark functions from CEC 2018 competition. After the comparison, we choose the best one from the six improved algorithms and further compare it with other multi-objective algorithms on large-scale multi-objective optimization (LSMOP1-LSMOP9). MOEA/D-IFM is also compared with CVEA3 [42] on many-objective optimization and test problems from CEC 2018 competition. Finally, we use MOEA/D-IFM to solve a practical problem that is a multi-objective knapsack problem. The experimental values are the results of taking the average value after 20 independent runs.

5.1. Comparison of AMPDEA and MOEA/D-PaS with MOEA/D-IFM

5.1.1. Test instances

In this section, we used the benchmark functions from CEC 2018 competition on many-objective optimization to test the performance of eight comparative algorithms. The test functions are MaF1-MaF15 [5] originated from CEC 2018 competition on many-objective optimization. All test functions have 5,10, and 15 objectives, respectively.

Table 1
The IGD results of the eight algorithms on test problems.

M	AMPDEA	MOEAD1	MOEAD2	MOEAD3	MOEAD4	MOEAD5	MOEAD6	MOEADPaS
5	4	3	0	0	5	0	0	3
10	2	4	0	0	7	0	0	2
15	2	4	0	0	6	0	0	3

Table 2
The HV results of the seven algorithms on test problems.

M	AMPDEA	MOEAD1	MOEAD2	MOEAD3	MOEAD4	MOEAD5	MOEAD6	MOEADPaS
5	3	1	0	0	6	0	0	3
10	3	1	0	0	4	0	0	3
15	5	2	0	0	3	0	0	3

5.1.2. Metrics

We have two main purposes in solving multi-objective problems. One is the gap between the PF and the real PF, which is the accuracy of the algorithm. The second is the diversity of the solution distribution. Based on the above two points, we use IGD [46] and HV [16] as metrics of algorithm performance.

(1) Inverted generational distance (IGD) [46]: The IGD value is the mean of the sum of the distances calculated from the points in each PF (true) to the points in its nearest Approximation Front.

$$IGD(PF, A) = \frac{\sum_{i=1}^{|p|} d(PF, A)}{|P|} \tag{16}$$

where PF represents the true Pareto front of the solution, A represents the approximate Pareto front value, $d(PF, A)$ represents the minimum Euclidean distance between PF and A, $|P|$ is the number of individuals distributed on true Pareto front. If the performance is better, the distance between the value of the true Pareto front and the value of the approximate Pareto front will be smaller, and the value of the IGD will be smaller. Therefore, the smaller the value of the IGD, the convergence and distribution of multi-objective algorithm will be better.

(2) HyperVolume (HV) [16]: The value of HV represents the volume of the region dominated by the obtained Pareto front. HV can simultaneously evaluate the convergence and distribution of the solution set. The larger the value of HV, the better the quality of the solution set. For the same solution set, the selected reference points are different, then the value of HV is different.

5.1.3. Parameter settings

In this section, we compare six MOEA/D algorithms incorporated into the information feedback models with AMPDEA [6] and MOEA/D-PaS [32]. The six information feedback models are M1-1, M1-2, M1-3, M2-1, M2-2, and M2-3. There are many ways to generate weights for MOEA/D. In this section, we use PBI method to generate weight vectors. In order to test the fairness of the algorithm performance, the eight algorithms have the same population size of 100 and maximum number of iterations of 10,000. IGD and HV values are the results of taking the average value after 20 runs. The probability of crossover rate $Pc = 1$, the distribution index of simulated binary crossover $disC = 20$, the expectation of number of bits mutation rate $proM = 1$, the distribution index of polynomial mutation $disM = 20$.

5.1.4. Experimental Results

For the experimental results of five-objective, ten-objective, and fifteen-objective problems, we only give the number of times when each algorithm reached the optimal. The results of the IGD of the eight algorithms on the five-objective, ten-objective, and fifteen-objective problems are shown in Table 1. The HV results of the eight algorithms are shown in Table 2. MOEA/D1-MOEA/D6 are six MOEA/D algorithms incorporated into models M2-1, M2-2, M2-3, M1-1, M1-2, and M1-4, respectively. M indicates the objective number of test problems, and N indicates the size of the population.

On the five-objective test functions, AMPEAD can obtain the best average IGD value on the MaF1, MaF3, MaF4, and MaF12 test functions, and MOEA/D-PAS can obtain the best average IGD value on the test functions MaF5, MaF11, and MaF14. On the remaining eight test functions, our proposed algorithm MOEA/D-IFM can get the smallest IGD value. This advantage is more obvious on the ten-objective test functions. Only AMPDEA has the smallest average IGD value on the two test functions MaF3 and MaF6. Similarly, only MOEA/D-PaS obtained the best average IGD value on the two test functions MAF1 and MAF4. MOEA/D-IFM can get the best IGD values on 11 out of 15 test functions. For the results of the fifteen-objective test functions, MOEA/D-IFM is also the best in all comparative algorithms, and 10 out of 15 test functions are optimal.

From the perspective of the value of HV, the performance of our proposed MOEA/D-IFM algorithm is also excellent. Table 2 is the results of the HV on five-objective, ten-objective, fifteen-objective, and twenty-objective problems, respectively. It can be seen that HV values of MODE/D-IFM are significantly larger than AMPDEA and MOEA/D-PaS for most test problems. On the all five-objective test functions, there are two test functions where all algorithms cannot get the HV value.

Table 3
The IGD results of the seven algorithms on test problems.

M	MOEAD4	NSGAIII	RVEA	KnEA	MOEADPaS	MOEADD	CMOPSO
5	7	0	0	0	0	2	0
10	6	0	0	0	0	3	0
15	7	0	0	1	0	1	0
20	7	0	0	1	0	0	1

Table 4
The HV results of the seven algorithms on test problems.

M	MOEAD4	NSGAIII	RVEA	KnEA	MOEADPaS	MOEADD	CMOPSO
5	5	0	0	1	0	1	1
10	4	0	0	0	0	0	2
15	5	0	0	0	0	0	2
20	3	0	0	0	0	0	2

In addition, MOEA/D-IFM can get the best HV values on 7 out of 13 test functions. Both AMPDEA and MOEA/D-PaS have 3 optimal HV values. For the results of ten-objective test functions, MOEA/D-IFM can get the best HV values on five test problems (MaF5, MaF7, MaF8, MaF10, and MaF12), and AMPDEA performs best on three test function (MaF2, MaF6, and MaF11). However, for the fifteen-objective test functions, both AMPDEA and MOEA/D-IFM can get the optimal HV values on 5 test functions, and MPEA/D-PaS has the optimal performance on four test functions.

From the perspective of both IGD and HV, the performance of our proposed improved algorithm is better than AMPDEA and MOEA/D-PaS. From the final comprehensive results, MOEA/D4 performs best. The information feedback model used in MOEA/D4 is M1-1. M1-1 selects the historical information from the previous generation at the same position with the current individual being updated. In order to more clearly observe the comparative results, we give a comparison figure of IGD values obtained by eight algorithms on 15 test functions, as shown in Fig. 1. From Fig. 1, we can observe that the MOEA/D4 algorithm can get optimal results in most cases, and its convergence speed is the fastest.

5.2. The best one to compare with other multi-objective algorithms

We can see that MOEA/D4 performs best after comparing with the above eight algorithms, so we further compare MOEA/D4 with other six multi-objective optimization algorithms, whichi are RVEA [3], KnEA [47], MOEA/D-PaS [32], NSGA-III [8],CMOPSO [48], and MOEA/DD [15]. In this section, the evaluation criteria are the same with Section 5.1. The test function we used are large-scale multi-objective and many-objective optimization (LSMOP1-LSMOP9) problems [4]. All test functions have 5,10,15, and 20 objectives, respectively.

5.2.1. Comparison with six multi-objective algorithms

Among the six algorithms used for comparing with MOEA/D4, REVA [14] is a secularization approach which can adopt itself to balance convergence and diversity of the solutions in the high-dimensional objective space. KnEA [11] is knee point-driven MOEA, where knee points of the non-dominated fronts in the current population are preferred in selection. NSGA-III [8] is a reference-point-based many-objective evolutionary algorithm. NSGA-III is proposed based on the framework of NSGA-II. MOEA/D-PaS [32] provides a simple yet effective method called Pareto adaptive scalarizing (PaS) approximation method, and then combines this method into MOEA/D to form MOEA/D-PaS. CMOPSO [48] is a multi-objective particle swarm optimization algorithm based on competition mechanism. The particle update in the population is related to its competing particles. MOEAD/D [15] is an evolutionary many-objective optimization algorithm based on dominance and decomposition. MOEAD/D can combine dominance- and decomposition-based approaches to balance the convergence and diversity of the evolutionary process.

5.2.2. Experimental results

For the experimental results, we only give the statistical data. Table 3 records the results of IGD for each algorithm on the five-objective, ten-objective, fifteen-objective, and twenty-objective test problems, respectively. As can be seen from Table 3, MOEA/D4 can get the best IGD values on most test functions. It can be seen that the performance of MOEA/D4 on large-scale multi-objective problems is significant. Table 4 records the HV values of each algorithm on the 5-objective, 10-objective, 15-objective, and 20-objective test problems. It is clear from Table 4 that MOEA/D4 is significantly better than other algorithms. For the ten, fifteen, and twenty objectives, MOEA/D4 also showed its superiority. The best results can be obtained on most functions in the experiment. Through the comparison of the experimental results, we can find that the improved MOEA/D-IFM algorithm can obtain a better solution in most cases.

In addition, Fig. 2 shows the IGD values obtained by seven multi-objective algorithms. The difference between the comparative algorithms can be more clearly. Experiments show that MOEA/D4 improved by the information feedback model M-1 can perform optimally on the test problems of various objectives (Fig. 3).

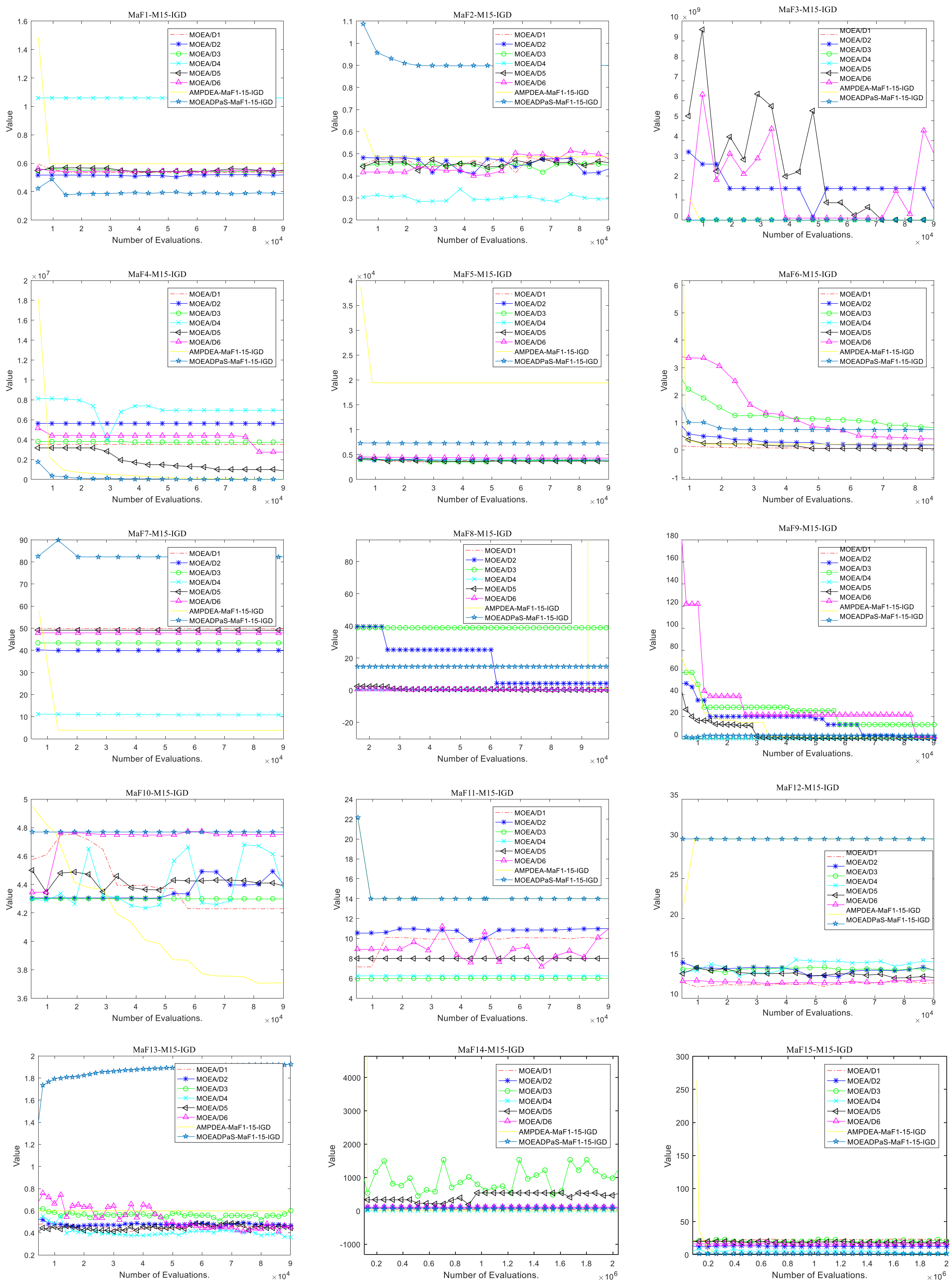

Fig. 1. Evolution of the mean of IGD values obtained by MOEA/D-PaS, AMPDEA, MOEA/D1, MOEA/D2, MOEA/D3, MOEA/D4, MOEA/D5, and MOEA/D6 for MaF1-15.

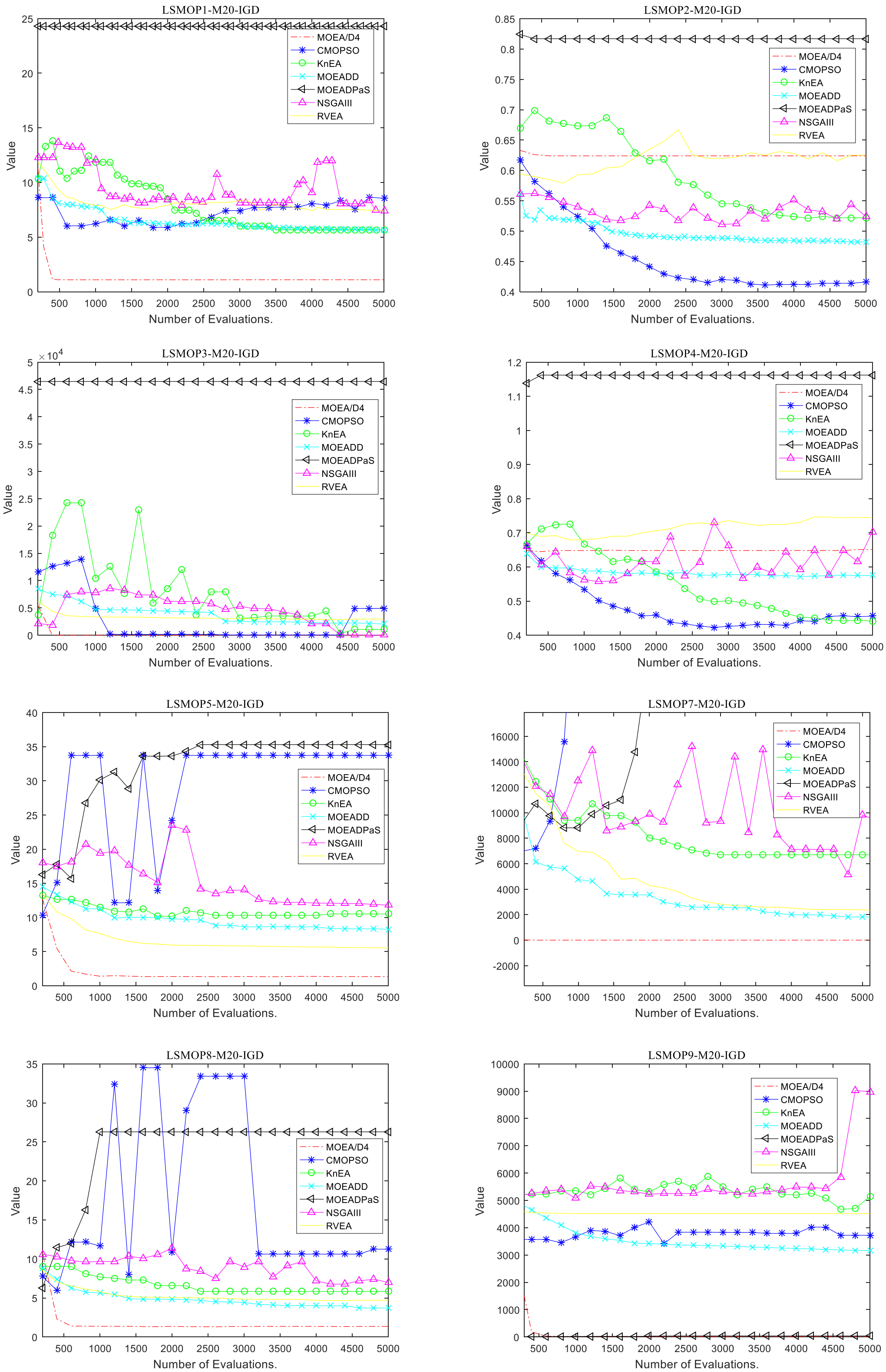

Fig. 2. Evolution of the mean IGD values obtained by MOEA/D4, KnEA, MOEA/DD, NSGA-III, RVEA, and MOEA/D-PaS for LSMOP1 on twenty-objective test problems.

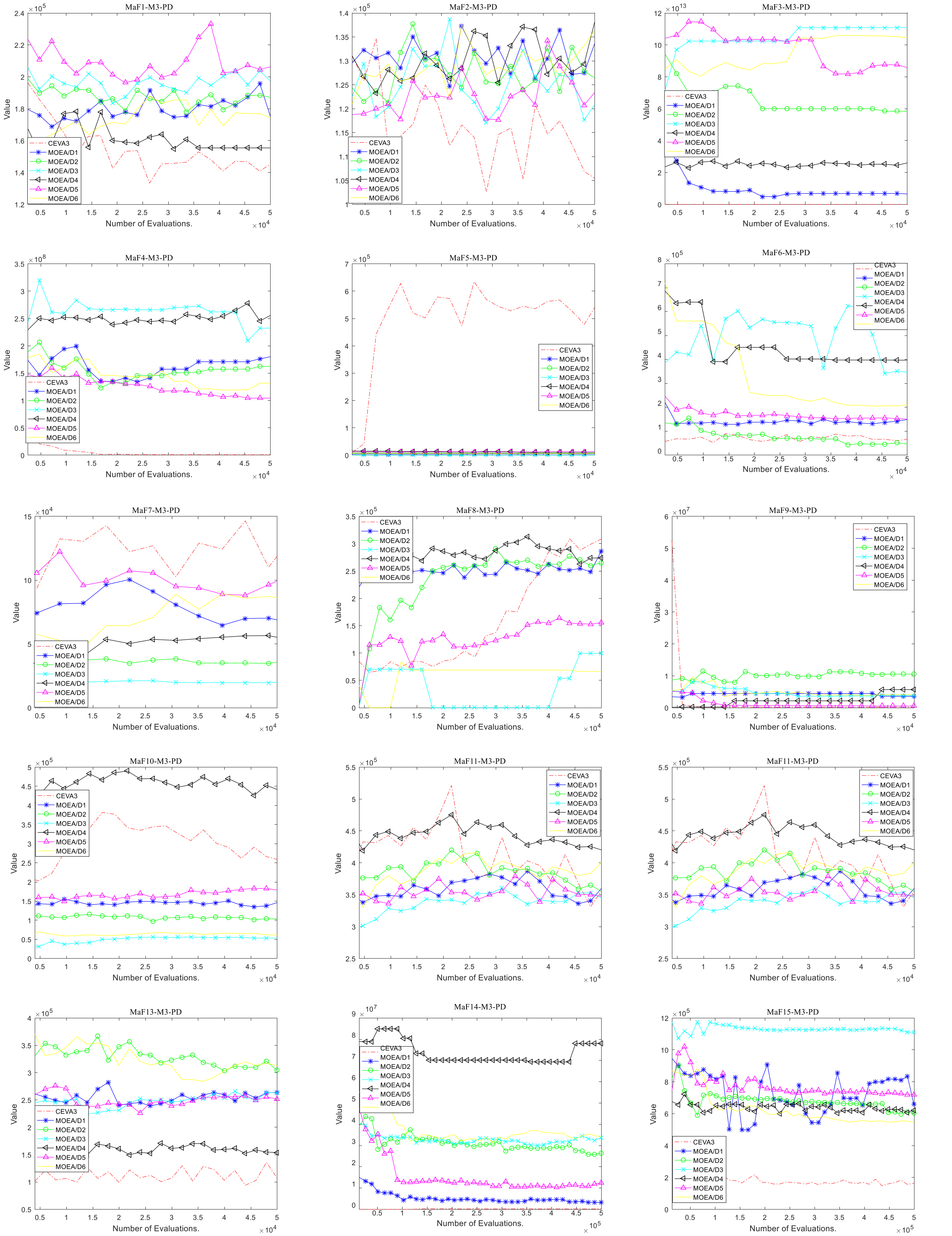

Fig. 3. Evolution of the mean of PD values obtained by CVEA3, MOEA/D1, MOEA/D2, MOEA/D3, MOEA/D4, MOEA/D5, and MOEA/D6 for MaF1-15.

Table 5
The Coverage results of the seven algorithms on test problems.

M	CVEA3	MOEAD1	MOEAD2	MOEAD3	MOEAD4	MOEAD5	MOEAD6
3	5	2	0	0	6	0	1
10	6	2	0	0	5	0	0
15	5	1	1	0	4	1	0

Table 6
The PD results of the seven algorithms on test problems.

M	CVEA3	MOEAD1	MOEAD2	MOEAD3	MOEAD4	MOEAD5	MOEAD6
3	3	1	0	1	4	5	1
10	4	0	1	0	5	3	2
15	7	1	0	0	4	1	2

5.3. Comparison of CEVA with MOEA/D-IFM

In this section, we used the benchmark functions from CEC 2018 competition on many-objective optimization to test the performance of seven comparative algorithms. All test functions have 3,10, and 15 objectives, respectively.

5.3.1. Metrics

(1) Set Coverage (C-metric) [45]: A and B are two approximations of a MOP, $C(A, B)$ is defined as the percentage of the solutions B in which are dominated by at least one solution in A. $C(A, B) = 0.1$ means that 10% of the solutions in B are dominated by at least one solution in A. $C(A, B) = 0$ means that no solutions in B are dominated by solutions in A. $C(A, B)$ is not necessarily equal to 1 - $C(B, A)$.

$$C(A, B) = \frac{|\{u \in B | \exists v \in A : v \text{dominates} u\}|}{|B|} \tag{17}$$

(2) Pure Diversity (PD) [30]: PD is a recently proposed metric to evaluate the diversity of multi-objective algorithms. X is a solution set and s is a solution in X. $(s, X-s)$ is the difference between solution s and other solutions in solution set X.

$$\begin{aligned} \mathrm{PD}(X) &= \max_{s_i \in X}(\mathrm{PD}(x - s_i) + d(s_i, X - s_i)) \\ d(s, X) &= \min_{s_i \in X}(\mathrm{dsissmilarity}(s, s_i)) \end{aligned} \tag{18}$$

5.3.2. Experimental Results

Tables 5-6 are the experimental statistical data. As we can see from Tables 5-6, MOEA/D-IFM can obtain best results in most of the test instances. MOEA/D-IFM obtains significantly better Coverage than CVEA3. For three-objective, ten-objective, and fifteen-objective test problems, the proportion of MOEA/D-IFM obtains best results are 9/15, 7/15, and 7/15, respectively. For MaF10 test problem, the performance of MOEA/D-IFM and CVEA3 are equal. For the PD metric, MOEA/D-IFM outperforms CVEA3 on 12 out of 15 on three-objective test problems. For ten-objective test problems and fifteen-objective test problems, MOEA/D-IFM can obtain 11 and 8 best results on all 15 test problems. It is clear from Tables 5-6 that MOEA/D-IFM shows better performance than CVEA3 in most test instances.

5.4. MOEA/D-IFM implementations for the multi-objective knapsack problem (MOKP)

5.4.1. Problem Description

For a MOKP problem with n items and m backpacks, we can mathematically express it as:

$$\begin{aligned} & x = (x_1, x_2, \ldots, x_m) \in \{0, 1\}^m \\ & \sum_{j=1}^{m} w_{i,j} \cdot x_j \le c_i \forall i \in \{1, 2, \ldots, n\} \\ & \text{maximize} f_i(x) = \sum_{j=1}^{m} p_{i,j} \cdot x_j \end{aligned} \tag{19}$$

where $p_{i,j}$ is the profit of item j in backpack i, $w_{i,j}$ is weight of item j in knapsack i, c_i is capacity of knapsack i. Our task is to put these j items into i backpacks to maximize the profits, and the items placed in each backpack cannot exceed the capacity of the backpack.

Table 7
The IGD results of the nine algorithms on the MOKP.

Problem	M	N	MOEAD1	MOEAD2	MOEAD3	MOEAD4	MOEAD5	MOEAD6	MOEADD	MOEAD	NSGAIII
MOKP	2	250	1.1235e+4 (1.83e+2) +	1.1065e+4 (1.57e+2) +	1.1175e+4 (2.74e+2) +	1.1063e+4 (1.71e+2) +	1.1183e+4 (1.55e+2) +	1.1097e+4 (2.00e+2) +	1.2533e+4 (1.43e+2) +	1.2815e+4 (6.61e+1) -	1.2664e+4 (1.03e+2)
MOKP	2	500	2.2328e+4 (2.68e+2) +	2.2117e+4 (3.24e+2) +	2.2323e+4 (3.09e+2) +	2.2123e+4 (3.24e+2) +	2.2212e+4 (3.00e+2) +	2.2114e+4 (2.22e+2) +	2.5220e+4 (3.57e+2) +	2.6107e+4 (1.73e+2) -	2.5576e+4 (1.72e+2)
MOKP	2	750	3.2477e+4 (3.21e+2) +	3.2111e+4 (3.58e+2) +	3.2414e+4 (3.57e+2) +	3.2176e+4 (2.77e+2) +	3.2124e+4 (2.99e+2) +	3.2092e+4 (4.24e+2) +	3.6513e+4 (6.76e+2) +	3.7940e+4 (2.60e+2) -	3.7031e+4 (2.23e+2)
MOKP	3	250	1.3133e+4 (2.52e+2) +	1.3031e+4 (1.37e+2) +	1.2974e+4 (1.90e+2) +	1.2889e+4 (1.73e+2) +	1.3053e+4 (2.06e+2) +	1.2965e+4 (1.49e+2) +	1.4691e+4 (1.33e+2) =	1.4979e+4 (7.96e+1) -	1.4648e+4 (1.15e+2)
MOKP	3	500	2.5776e+4 (3.29e+2) +	2.5349e+4 (3.49e+2) +	2.5566e+4 (2.98e+2) +	2.5387e+4 (2.68e+2) +	2.5422e+4 (2.92e+2) +	2.5437e+4 (2.89e+2) +	2.8392e+4 (1.96e+2) +	2.9512e+4 (1.61e+2) -	2.8589e+4 (2.46e+2)
MOKP	3	750	3.8291e+4 (3.19e+2) +	3.7744e+4 (4.24e+2) +	3.8172e+4 (4.73e+2) +	3.7703e+4 (3.16e+2) +	3.7949e+4 (4.12e+2) +	3.7859e+4 (3.52e+2) +	4.1845e+4 (2.19e+2) +	4.3911e+4 (2.72e+2) -	4.2353e+4 (1.78e+2)
MOKP	4	250	1.4854e+4 (2.30e+2) +	1.4697e+4 (1.86e+2) +	1.4657e+4 (1.94e+2) +	1.4617e+4 (1.77e+2) +	1.4781e+4 (1.75e+2) +	1.4564e+4 (2.12e+2) +	1.6323e+4 (2.97e+2) =	1.6913e+4 (1.07e+2) -	1.6384e+4 (1.60e+2)
MOKP	4	500	2.9240e+4 (3.81e+2) +	2.8902e+4 (1.98e+2) +	2.9033e+4 (3.44e+2) +	2.8850e+4 (3.12e+2) +	2.8929e+4 (3.12e+2) +	2.8978e+4 (2.51e+2) +	3.2255e+4 (3.77e+2) =	3.3518e+4 (1.66e+2) -	3.2284e+4 (2.00e+2)
MOKP	4	750	4.2873e+4 (4.07e+2) +	4.2513e+4 (5.24e+2) +	4.2736e+4 (4.30e+2) +	4.2388e+4 (3.86e+2) +	4.2586e+4 (2.87e+2) +	4.2484e+4 (3.57e+2) +	4.6735e+4 (8.74e+2) =	4.9185e+4 (1.79e+2) -	4.6999e+4 (5.79e+2)

5.4.2. Parameter settings

In this section, we use nine MOKP test instances. The number of backpacks m is set to 2, 3, and 4, respectively, and the number of items n is 250, 500, and 750, respectively. The weight and the profit are random values generated between [10,100]. The population size is 100 and the maximum number of iterations is 10,000. The experiment was run 20 times and the average values were taken.

5.4.3. Experimental results

It can be clearly seen from Table 7 that the average and standard deviation of IGD obtained by MOEA/D-IFM are superior to other comparative multi-objective algorithms. The experimental results obtained by each algorithm can be shown in detail. The **bold** font in Table 7 indicates that it is the best performer among all comparative algorithms. MOEA/D4 can get the optimal IGD value on the test of 250 items in two backpacks, 250 items in 3 backpacks, 750 items in 3 backpacks, 500 items in 4 backpacks, and 750 items in 4 backpacks. MOEA/D6 achieves the minimum IGD value for 500 items in 2 backpacks, 750 items in 2 backpacks, and 250 items in 4 backpacks. On all 9 test instances, our proposed MOEA/D-IFM is better than other comparative algorithms. This further proves that MOEA/D-IFM has certain advantages in solving MOKP.

6. Conclusions

In the previous multi-objective optimization algorithms, we rarely or even did not use the population information before the current iteration. In the process of evolution, we were likely to abandon some useful information. This paper proposed a new multi-objective optimization algorithm based on MOEA/D with information feedback models. The feedback model can make up for the shortcomings that had wasted a lot of useful information. The information feedback models preserve historical information and selects individuals in a random or fixed manner. We verified the performance of the MOEA/D-IFM algorithm on multiple sets of multi-objective test problems. We compared the six improved MOEA/D-IFM algorithms with AMPDEA, MOEA/D-PaS, and CVEA3 on many-objective optimization problems from CEC 2018 competition. Among six MOEA/D-IFM algorithms, MOEA/D4 had the best performance. Then the best MOEA/D4 algorithm was further compared with KnEA, MOEA/D-PaS, NSGA-III, RVEA, and MOEA/DD. The practical problem of a set of multi-objective backpacks is also used to verify the performance of the algorithm. The experimental results showed that the MOEA/D-IFM algorithm incorported into the information feedback model is superior to the comparative state-of-the-art multi-objective algorithms.

In this paper, we have improved the MOEA/D algorithm by using only six simplest information feedback models. In future work, we will try to incorporate other information feedback models into MOEA/D. These improved algorithms will be used to solve engineering problems, such as test-sheet composition [10], path planning [28], big data optimization [40,41], fault diagnosis [39], and other multi-objective engineering problems. In future work, we will compare MOEA/D with other multi-objective algorithms, such as a new resource allocation strategy based on the relationship between sub-problems for MOEA/D [31].

Declaration of Competing Interest

None

CRediT authorship contribution statement

Yin Zhang: Conceptualization, Methodology, Software. **Gai-Ge Wang:** Supervision, Writing - original draft. **Keqin Li:** Visualization, Investigation. **Wei-Chang Yeh:** Data curation, Writing - review & editing. **Muwei Jian:** Software, Validation. **Junyu Dong:** Data curation, Writing - review & editing.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 41576011, No. U1706218, No. 41706010, No. 41927805, and No. 61503165). All the authors thanks all the reviewers for their comments.

References

[1] R. Akbari, R. Hedayatzadeh, K. Ziarati, B. Hassanizadeh, A multi-objective artificial bee colony algorithm, Swarm Evol. Comput. 2 (2012) 39–52.
[2] B. Chen, W. Zeng, Y. Lin, D. Zhang, A new local search-based multiobjective optimization algorithm, IEEE Trans. Evol. Comput. 19 (1) (2015) 50–73.
[3] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, A reference vector guided evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 20 (5) (2016) 773–791.
[4] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, Test problems for large-scale multiobjective and many-objective optimization, IEEE Trans. Cybern. 47 (12) (2017) 4108–4121.
[5] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, X. Yao, A benchmark test suite for evolutionary many-objective optimization, Complex Intell. Syst. 3 (1) (2017) 67–81.
[6] R. Cheng, M. Li, Y. Tian, X. Xiang, X. Zhang, S. Yang, Y. Jin, X. Yao. Competition on many objective optimization at 2018 IEEE Congress on Evolutionary Computation http://www.cs.bham.ac.uk/~chengr/CEC_Comp_on_MaOO/2018/webpage.html, 2018.
[7] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[8] K. Deb, 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. 18 (4) (2014) 577–601.
[9] M. Dorigo, G.D. Caro, Ant colony optimization: a new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation (CEC 1999), 1999, pp. 1470–1477.
[10] H. Duan, W. Zhao, G. Wang, X. Feng, Test-sheet composition using analytic hierarchy process and hybrid metaheuristic algorithm TS/BBO, Math. Probl. Eng. 2012 (2012) 1–22.
[11] R. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks: IEEE, 1995, pp. 1942–1948.
[12] A.H. Gandomi, A.H. Alavi, Krill herd: a new bio-inspired optimization algorithm, Commun. Nonlin. Sci. Numer. Simulat. 17 (12) (2012) 4831–4845.
[13] H. Jain, K. Deb, An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: handling constraints and extending to an adaptive approach, IEEE Trans. Evol. Comput. 18 (4) (2014) 602–622.
[14] L. Ke, Q. Zhang, R. Battiti, MOEA/D-ACO: a multiobjective evolutionary algorithm using decomposition and antcolony, IEEE Trans. Cybern. 43 (6) (2013) 1845–1859.
[15] K. Li, K. Deb, Q. Zhang, S. Kwong, An evolutionary many-objective optimization algorithm based on dominance and decomposition, IEEE Trans. Evol. Comput. 19 (5) (2015) 694–716.
[16] Z. Liang, R. Song, Q. Lin, Z. Du, J. Chen, Z. Ming, J. Yu, A double-module immune algorithm for multi-objective optimization problems, Appl. Soft Comput. 35 (2015) 161–174.
[17] X. Ma, F. Liu, Y. Qi, M. Gong, M. Yin, L. Li, L. Jiao, J. Wu, MOEA/D with opposition-based learning for multiobjective optimization problem, Neurocomputing 146 (2014) 48–64.
[18] X. Ma, Y. Qi, L. Li, F. Liu, L. Jiao, J. Wu, MOEA/D with uniform decomposition measurement for many-objective problems, Soft Comput. 18 (12) (2014) 2541–2564.
[19] A. Mohammadi, H. Asadi, S. Mohamed, K. Nelson, S. Nahavandi, OpenGA, a C++ genetic algorithm library, in: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2017), 2017, pp. 2051–2056.
[20] A. Mohammadi, H. Asadi, S. Mohamed, K. Nelson, S. Nahavandi, Multiobjective and interactive genetic algorithms for weight tuning of a model predictive control-based motion cueing algorithm, IEEE Trans. Cybern. 49 (9) (2019) 3471–3481.
[21] Y. Qi, J. Yu, X. Li, Y. Quan, Q. Miao, Enhancing robustness of the inverted PBI scalarizing method in MOEA/D, Appl. Soft Comput. 71 (2018) 1117–1132.
[22] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[23] N. Srinivas, K. Deb, Muiltiobjective optimization using nondominated sorting in genetic algorithms, Evol. Comput. 2 (3) (1994) 221–248.
[24] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[25] Y.-Y. Tan, Y.-C. Jiao, H. Li, X.-K. Wang, A modification to MOEA/D-DE for multiobjective optimization problems with complicated Pareto sets, Inf. Sci. 213 (2012) 14–38.
[26] G.-G. Wang, L. Guo, A.H. Gandomi, G.-S. Hao, H. Wang, Chaotic krill herd algorithm, Inf. Sci. 274 (2014) 17–34.
[27] G.-G. Wang, S. Deb, Monarch butterfly optimization, Neural Comput. Appl. 31 (7) (2019) 1995–2014.
[28] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, M. Shao, Path planning for uninhabited combat aerial vehicle using hybrid meta-heuristic DE/BBO algorithm, Adv. Sci., Eng. Med. 4 (6) (2012) 550–564.
[29] G. Wang, Y. Tan, Improving metaheuristic algorithms with information feedback models, IEEE Trans. Cybern. 49 (2) (2019) 542–555.
[30] H. Wang, Y. Jin, X. Yao, Diversity assessment in many-objective optimization, IEEE Trans. Cybern. 47 (6) (2016) 1–13.
[31] P. Wang, W. Zhu, H. Liu, B. Liao, L. Cai, X. Wei, S. Ren, J. Yang, A new resource allocation strategy based on the relationship between subproblems for MOEA/D, Inf. Sci. 501 (2019) 337–362.
[32] R. Wang, Q. Zhang, T. Zhang, Decomposition-based algorithms using pareto adaptive scalarizing methods, IEEE Trans. Evol. Comput. 20 (6) (2016) 821–837.
[33] R. Wang, J. Xiong, H. Ishibuchi, G. Wu, T. Zhang, On the effect of reference point in MOEA/D for multi-objective optimization, Appl. Soft Comput. 58 (2017) 25–34.
[34] Z. Wang, Q. Zhang, A. Zhou, M. Gong, L. Jiao, Adaptive replacement strategies for MOEA/D, IEEE Trans. Cybern. 46 (2) (2016) 474–486.
[35] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.
[36] Y. Xiang, Y. Zhou, M. Li, Z. Chen, A vector angle-based evolutionary algorithm for unconstrained many-objective optimization, IEEE Trans. Evol. Comput. 21 (1) (2017) 131–152.
[37] X. Yang, D. Suash, Cuckoo search via lévy flights, in: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC 2009), 2009, pp. 210–214.
[38] X.S. Yang, Bat algorithm: a novel approach for global engineering optimization, Eng. Comput. 29 (5) (2012) 464–483.
[39] J.-H. Yi, J. Wang, G.-G. Wang, Improved probabilistic neural networks with self-adaptive strategies for transformer fault diagnosis problem, Adv. Mech. Eng. 8 (1) (2016) 1–13.

[40] J.-H. Yi, S. Deb, J. Dong, A.H. Alavi, G.-G. Wang, An improved NSGA-III algorithm with adaptive mutation operator for Big Data optimization problems, Future Gener. Comput. Syst. 88 (2018) 571–585.
[41] J.-H. Yi, L.-N. Xing, G.-G. Wang, J. Dong, A.V. Vasilakos, A.H. Alavi, L. Wang, Behavior of crossover operators in NSGA-III for large-scale optimization problems, Inf. Sci. 509 (2020) 470–487.
[42] J. Yuan, H. Liu, F. Gu, A cost value based evolutionary many-objective optimization algorithm with neighbor selection strategy, in: 2018 IEEE Congress on Evolutionary Computation (CEC 2018), 2018, pp. 1–8.
[43] C. Zhang, K.C. Tan, L.H. Lee, L. Gao, Adjust weight vectors in MOEA/D for bi-objective optimization problems with discontinuous Pareto fronts, Soft Comput. 22 (12) (2018) 3997–4012.
[44] H. Zhang, A. Zhou, S. Song, Q. Zhang, X. Gao, A self-organizing multiobjective evolutionary algorithm, IEEE Trans. Evol. Comput. 20 (5) (2016) 792–806.
[45] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[46] Q. Zhang, A. Zhou, Y. Jin, a regularity model-based multiobjective estimation of distribution algorithm, IEEE Trans. Evol. Comput. 12 (1) (2008) 41–63.
[47] X. Zhang, Y. Tian, Y. Jin, A knee point-driven evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 19 (6) (2015) 761–776.
[48] X. Zhang, X. Zheng, R. Cheng, J. Qiu, Y. Jin, A competitive mechanism based multi-objective particle swarm optimizer with fast convergence, Inf. Sci. 427 (2018) 63–76.
[49] W. Zheng, Y. Tan, L. Meng, H. Zhang, An improved MOEA/D design for many-objective optimization problems, Appl. Intell. 48 (2018) 3839–3861.
[50] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength pareto evolutionary algorithm, 2001 TIK-report.