Efficient Large-Scale Multi-Objective Optimization Based on A Competitive Swarm Optimizer

Ye Tian, Xiutao Zheng, Xingyi Zhang, *Senior Member, IEEE*, and Yaochu Jin, *Fellow, IEEE*,

***Abstract*—There exist many multi-objective optimization problems (MOPs) containing a large number of decision variables in real-world applications, which are known as large-scale MOPs. Due to the ineffectiveness of existing operators in finding optimal solutions in a huge decision space, some decision variable division based algorithms have been tailored for improving the search efficiency in solving large-scale MOPs. However, these algorithms will encounter difficulties when solving problems with complicated landscapes, as the decision variable division is likely to be inaccurate and time-consuming. In this paper, we propose a competitive swarm optimizer (CSO) based efficient search for solving large-scale MOPs. The proposed algorithm adopts a new particle updating strategy that suggests a two-stage strategy to update position, which can highly improve the search efficiency. Experimental results on large-scale benchmark MOPs and an application example demonstrate the superiority of the proposed algorithm over several state-of-the-art multi-objective evolutionary algorithms, including problem transformation based algorithm, decision variable clustering based algorithm, particle swarm optimization algorithm, and estimation of distribution algorithm.**

***Index Terms*—Evolutionary multi-objective optimization, large-scale multi-objective optimization problem, competitive swarm optimizer, particle swarm optimization**

I. Introduction

MULTI-objective optimization problems (MOPs) are commonly seen in real-world applications [1]–[5], which are characterized by multiple objectives conflicting with each other. Due to the conflicting nature of the objectives, a set of trade-off solutions rather than a single optimal solution are expected to be found for each MOP. While general MOPs have been extensively studied for many years [6], little work has been dedicated to solving the MOPs with a large number of decision variables, even though they also widely exist in real-world applications [7]–[9]. Although there is no formal definition, MOPs with more than 100 decision variables are known as large-scale MOPs [10], [11].

Generally speaking, large-scale MOPs are much more difficult to be solved than those with a few decision variables, since the search space is exponentially related to the number of decision variables, i.e., the curse of dimensionality, which makes it impossible for multi-objective evolutionary algorithms (MOEAs) to explore the search space efficiently. In this case, MOEAs are likely to converge prematurely to a local optimum or converge to a too large region [12]. As a result, it has been empirically verified that most existing MOEAs cannot solve large-scale MOPs well [10], [11]. In fact, most existing MOEAs improve the performance in solving MOPs by enhancing the environmental selection (i.e., the strategy to select solutions from the current population for the next generation) [13]–[15]; however, their search efficiency is not high enough to find optimal solutions in a large search space, hence they are incapable of solving large-scale MOPs no matter how effective the environmental selection is. To address this issue, some MOEAs have been tailored for improving the search efficiency in solving large-scale MOPs, by means of the division of decision variables. These MOEAs can be roughly grouped into the following three categories.

The first category of MOEAs for large-scale MOPs is based on decision variable grouping. Inspired by the evolutionary algorithms for solving large-scale single-objective optimization problems (SOPs) [16]–[18], this category of MOEAs divides the decision variables into several groups, then optimizes each group of decision variables successively. For example, the CCGDE3 [19] randomly divides the decision variables into several groups of equal size, then optimizes each group of decision variables by GDE3 [20]. In [21], a cooperative coevolution algorithm called NSCCGA was proposed, which relates each decision variable to a subpopulation, and optimizes each subpopulation by NSGA-II indepen-

Manuscript received –. This work was supported in part by the National Natural Science Foundation of China under Grant 61672033, 61822301, 61876123, U1804262, State Key Laboratory of Synthetical Automation for Process Industries under Grant PAL-N201805 and Anhui Provincial Natural Science Foundation for Distinguished Young Scholars under Grant 1808085J06. *(Corresponding author: Xingyi Zhang)*

Y. Tian is with the Institutes of Physical Science and Information Technology, Anhui University, Hefei 230601, China. (email: field910921@gmail.com)

X. Zheng and X. Zhang are with the Institute of Bio-inspired Intelligence and Mining Knowledge, School of Computer Science and Technology, Anhui University, Hefei 230601, China. (email: zxt0086@outlook.com; xyzhanghust@gmail.com)

Y. Jin is with the Department of Computer Science, University of Surrey, Guildford, Surrey, GU2 7XH, U.K.. He is also with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China. (email: yaochu.jin@surrey.ac.uk)

dently.

The second category of MOEAs is based on decision variable analysis. Different from the MOEAs based on decision variable grouping, the decision variable analysis based MOEAs divide the decision variables into different types according to their contributions to the objective functions, and optimize each type of decision variables by different strategies. In [10], an algorithm called MOEA/DVA was developed based on control variable analysis, which classifies the decision variables into position variables, distance variables, and mixed variables, then divides all the distance variables into a number of independent subcomponents. Afterwards, MOEA/DVA optimizes each subcomponent of distance variables until the population converges, then optimizes all the decision variables for better diversity. Later, some MOEAs with similar ideas to MOEA/DVA have also been proposed [22], [23]. In [11], Zhang *et al.* suggested a decision variable clustering based MOEA for solving large-scale MOPs, called LMEA. In contrast to MOEA/DVA, LMEA classifies the decision variables into convergence-related variables and diversity-related variables, and iteratively optimizes each type of decision variables by different strategies. Experimental results demonstrate that LMEA is more accurate than MOEA/DVA in dividing the decision variables of large-scale MOPs.

More recently, a problem transformation based framework was proposed in [24], which can be regarded as the third category of MOEAs for large-scale MOPs. This framework divides the decision variables into a number of groups and optimizes a weight vector instead of the decision variables, hence the original large-scale MOP is transformed into a small-scale MOP, where the number of decision variables in the small-scale MOP equals to the number of groups.

In spite of various MOEAs for improving the search efficiency in solving large-scale MOPs, most of them are based on the division of decision variables, which will encounter difficulties in solving MOPs with complicated landscapes. To be specific, for the MOEAs based on decision variable grouping, two interacting variables may be divided into different groups, and thus the algorithms are likely to be trapped into local optimum when optimizing these two variables separately [16]. As for the MOEAs based on decision variable analysis, although they are capable of detecting the interactions between variables, this operation takes a large number of function evaluations, and the detecting result may be inaccurate when solving MOPs with complicated landscapes [11].

Nevertheless, few MOEAs for solving large-scale MOPs optimize all the decision variables together, which is mainly due to the low search efficiency of existing operators (e.g., simulated binary crossover [25], polynomial mutation [26], and differential evolution [27]). In this paper, we propose a competitive swarm optimizer (CSO) based efficient search for large-scale multi-objective optimization. Specifically, the main contributions of this paper are summarized as follows.

1) A particle updating strategy is proposed to improve the search efficiency for solving large-scale MOPs. In contrast to most existing algorithms that focus on the modification of updating velocity, the proposed strategy suggests a two-stage strategy to update position. To be specific, the proposed strategy first pre-updates the position of each particle according to its previous velocity, then updates the position of each pre-updated particle by learning from a leader. It is demonstrated that the proposed particle updating strategy can improve the search efficiency of algorithms.
2) A large-scale multi-objective CSO algorithm called LMOCSO is developed based on the proposed particle updating strategy. The proposed LMOCSO adopts a competitive mechanism to determine the particles to be updated, and uses the proposed updating strategy to update each particle. Afterwards, all the updated particles are combined with the original particles, and half the particles in the combined population survive to the next generation.
3) To verify the effectiveness of the proposed LMOCSO in solving large-scale MOPs, it is compared with four state-of-the-art MOEAs on the large-scale multi-objective test suite LSMOP [28]. Experimental results indicate that LMOCSO obtains significantly better results than the compared MOEAs on most test instances, and it has a good scalability with respect to the number of decision variables. Besides, the proposed LMOCSO is also verified on a large-scale MOP in application, i.e., the training of neural network [29].

The rest of this paper is organized as follows. In Section II, the particle updating strategies in existing MOEAs are briefly reviewed. Section III details the procedure of the proposed LMOCSO. Section IV presents the empirical results of LMOCSO and four state-of-the-art MOEAs on large-scale MOPs. Finally, the conclusions are drawn in Section V.

II. Related Work

A. Existing Particle Updating Strategies for Solving MOPs

As one of the most popular metaheuristics for solving optimization problems, particle swarm optimization (PSO) is a kind of swarm intelligence paradigm originally inspired by the behavior of bird flocking in nature [30]. Since the first multi-objective PSO algorithm was proposed [31], a number of PSO algorithms have been proposed for solving MOPs [32]–[42]. The most significant characteristic of PSO is that each particle (i.e., solution) has a position vector denoting the decision variables and a velocity vector, where the position is updated according to the velocity, and the velocity is updated according to the personal best position and the global best position.

Among existing multi-objective PSO algorithms, most of them focus on the modification of updating velocity

to enhance the performance in solving MOPs. Firstly, many algorithms use different strategies to determine the global best position for updating the velocity of each particle. For example, in MOPSO [31], SMPSO [33], and NMPSO [38], the global best position is the position of a particle randomly picked up from an archive. In MO_Ring_PSO_SCD [39], the global best position is the position of the best particle within the particle's neighborhood determined by a ring topology.

Secondly, some algorithms restrict the velocity after it is updated. For example, in OMOPSO [32] and dMOPSO [35], each velocity is reversed by being multiplied by -1 if the resulting position is out of the decision space. In SMPSO [33], each velocity is multiplied by a constriction factor after being updated, and then it is constrained according to the boundary values of decision variables. Moreover, some algorithms adapt the parameters in updating velocity, such as MO-TRIBES [34] and pccsAMOPSO [36].

In addition, some work adopts cooperative principles in multi-objective PSO algorithms. For instance, in CVEPSO [41], the vector-evaluated PSO [43] is equipped with cooperative principles [44], which successfully competes with state-of-the-art multi-objective PSO algorithms. In CCSMPSO [42], the SMPSO [33] is enhanced and parallelized by a cooperative coevolutionary paradigm, where both the effectiveness and efficiency are improved.

In recent years, CSO [45] has shown its competitiveness in solving large-scale SOPs, which is a novel swarm algorithm different from PSO. CSO randomly selects two particles each time, where the velocity of the particle with worse fitness value is updated according to the position of the particle with better fitness value. Specifically, the CSO based MOEA [40] updates a particle at the t-th generation by

$$\begin{aligned} \overrightarrow{\Delta}(t) &= \overrightarrow{x_w}(t) - \overrightarrow{x_l}(t) \\ \overrightarrow{v_l}(t+1) &= r_0\overrightarrow{v_l}(t) + r_1\overrightarrow{\Delta}(t), \\ \overrightarrow{x_l}(t+1) &= \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1) \end{aligned} \tag{1}$$

where $\overrightarrow{x_w}$ is the position of the leader (i.e., the particle with better fitness value), $\overrightarrow{x_l}$ and $\overrightarrow{v_l}$ are the position and velocity of the particle to be updated (i.e., the particle with worse fitness value), respectively, and r_0 and r_1 are uniformly randomly distributed values in $[0, 1]$.

B. Discussions

Although the original PSO fails on large-scale optimization problems due to the increasing roaming behavior (i.e., the tendency of particles to leave the search space early in the search process) in high-dimensional space [12], some adaptations of PSO [17], [18], [46] as well as CSO [45], [47] have been shown to be promising in tackling large-scale SOPs. Nevertheless, existing multi-objective PSO and CSO algorithms are effective in solving MOPs with no more than 50 decision variables

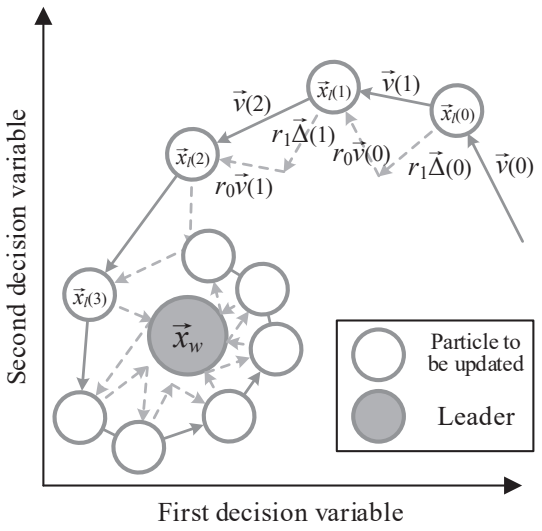


Fig. 1. Trajectory of a particle after learning from a fixed leader by the updating strategy of CSO given in Eq. (1). It can be found that the particle moves around the leader with a slow convergence speed.

[33], [40]. This is mainly because a set of optimal solutions rather than a single optimal solution are required for an MOP, but the search efficiency of PSO and CSO is not high enough to find a set of optimal solutions within a limited number of generations. To better illustrate this fact, Fig. 1 depicts the trajectory of one particle after learning from a fixed leader by CSO for several times. It is obvious from the figure that the particle moves around the leader with a slow convergence speed. Similarly, the trajectory of a particle in PSO may be an oscillatory graph bounded by exponential functions or a sinusoidal wave as reported in [48]. As a consequence, such an updating strategy cannot make the particles converge to optimal positions quickly, and thus may be ineffective in solving large-scale MOPs.

Taking a closer look at Fig. 1, it can be found that after the particle is updated by the difference between the positions of the leader and the particle $r_1\overrightarrow{\Delta}(t)$, it is further updated by its previous velocity $r_0\overrightarrow{v}(t)$. Therefore, the particle can directly move towards the leader only when $\overrightarrow{v}(t)$ and $\overrightarrow{\Delta}(t)$ have the same direction, but this scenario rarely occurs on large-scale MOPs, since the probability that two arbitrary vectors have similar directions decreases as the number of decision variables increases. For this reason, CSO will have a low search efficiency when it is employed to solve large-scale MOPs. To address this issue, the proposed LMOCSO uses a new particle updating strategy, which has higher search efficiency than the one shown in Fig. 1.

III. The Proposed LMOCSO

A. Particle Updating Strategy in LMOCSO

It is known that the position of a particle will be updated by its previous velocity $r_0\overrightarrow{v}(t)$ after it is updated

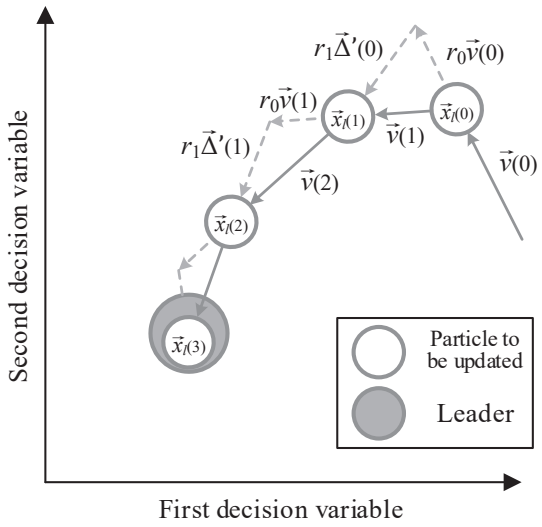


Fig. 2. Trajectory of a particle after learning from a fixed leader by the proposed updating strategy given in Eq. (6). It can be found that the particle moves towards the leader with a fast convergence speed.

Algorithm 1: $UpdatingParticle(P)$

Input: P (current population)
Output: P' (new population)

$Fitness \leftarrow$ Calculate the fitness of each particle in P by Eq. (7);
$P' \leftarrow \emptyset$;
while $|P| > 1$ **do**
  $\{\mathbf{p}, \mathbf{q}\} \leftarrow$ Randomly select two particles from P;
  $P \leftarrow P \setminus \{\mathbf{p}, \mathbf{q}\}$;
  if $Fitness(\mathbf{p}) < Fitness(\mathbf{q})$ **then**
    $\mathbf{x}_l \leftarrow \mathbf{p}$ // Particle to be updated
    $\mathbf{x}_w \leftarrow \mathbf{q}$ // Leader
  else
    $\mathbf{x}_l \leftarrow \mathbf{q}$ // Particle to be updated
    $\mathbf{x}_w \leftarrow \mathbf{p}$ // Leader
  Update $\mathbf{x}_l$ by learning from $\mathbf{x}_w$ by Eq. (6);
  Mutate $\mathbf{x}_l$ and $\mathbf{x}_w$ by polynomial mutation;
  $P' \leftarrow P' \cup \{\mathbf{x}_l, \mathbf{x}_w\}$;
return P';

by learning from a leader. Therefore, we can update the particle by its previous velocity $r_0\overrightarrow{v}(t)$ in advance, then update the particle by the difference between the new position of the particle and the leader $r_1\overrightarrow{\Delta}'(t)$, so that the particle can directly move towards the leader even if $\overrightarrow{v}(t)$ and $\overrightarrow{\Delta}'(t)$ have different directions. Fig. 2 plots the trajectory of one particle after learning from a fixed leader by the proposed particle updating strategy for several times. Obviously, the particle can move towards the leader with a fast convergence speed.

To be specific, in the proposed LMOCSO, two particles $\mathbf{x}_l$ and $\mathbf{x}_w$ are randomly picked up from the current population each time, and they are updated by their previous velocities:

$$\begin{aligned}\overrightarrow{x_l}'(t) &= \overrightarrow{x_l}(t) + r_0\overrightarrow{v_l}(t)\\ \overrightarrow{x_w}'(t) &= \overrightarrow{x_w}(t) + r_0\overrightarrow{v_w}(t)\end{aligned}. \tag{2}$$

Then, the particle with worse fitness value $\mathbf{x}_l$ is updated by learning from the better one $\mathbf{x}_w$ by:

$$\begin{aligned}\overrightarrow{\Delta}'(t) &= \overrightarrow{x_w}'(t) - \overrightarrow{x_l}'(t)\\ \overrightarrow{v_l}(t+1) &= r_0\overrightarrow{v_l}(t) + r_1\overrightarrow{\Delta}'(t)\\ \overrightarrow{x_l}(t+1) &= \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1)\end{aligned}. \tag{3}$$

It is noteworthy that after Eq. (2) is performed, the decision variables $\overrightarrow{x_l}$ and $\overrightarrow{x_w}$ are changed to $\overrightarrow{x_l}'$ and $\overrightarrow{x_w}'$, respectively, where $\overrightarrow{x_l}'$ may have better fitness value than $\overrightarrow{x_w}'$. Therefore, the objective values of $\overrightarrow{x_l}'$ and $\overrightarrow{x_w}'$ should be calculated to determine the leader and the particle to be updated. In other words, additional function evaluations are needed for updating each particle. To address this issue, we transform Eqs. (2) and (3) by the following way:

$$\begin{aligned}\overrightarrow{x_l}'(t+1) &= \overrightarrow{x_l}(t+1) + r_0\overrightarrow{v_l}(t+1)\\ &= \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1) + r_0\overrightarrow{v_l}(t+1)\\ &= \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1) + r_0\overrightarrow{v_l}(t+1)\\ &\quad + r_0\overrightarrow{v_l}(t) - r_0\overrightarrow{v_l}(t)\\ &= (\overrightarrow{x_l}(t) + r_0\overrightarrow{v_l}(t)) + \overrightarrow{v_l}(t+1)\\ &\quad + (r_0\overrightarrow{v_l}(t+1) - r_0\overrightarrow{v_l}(t))\\ &= \overrightarrow{x_l}'(t) + \overrightarrow{v_l}(t+1) + r_0(\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t))\end{aligned}. \tag{4}$$

Therefore, we have

$$\begin{aligned}\overrightarrow{v_l}(t+1) &= r_0\overrightarrow{v_l}(t) + r_1(\overrightarrow{x_w}'(t) - \overrightarrow{x_l}'(t))\\ \overrightarrow{x_l}'(t+1) &= \overrightarrow{x_l}'(t) + \overrightarrow{v_l}(t+1) + r_0(\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t))\end{aligned}, \tag{5}$$

where $\overrightarrow{x_l}$ is eliminated and Eq. (5) only contains $\overrightarrow{x_l}'$. In this case, we can obtain $\overrightarrow{x_l}'(t)$ by $\overrightarrow{x_l}'(t-1)$ without using $\overrightarrow{x_l}(t)$, and store $\overrightarrow{x_l}'(t)$ in the particle instead of $\overrightarrow{x_l}(t)$. Moreover, since the objective values of $\overrightarrow{x_l}'$ rather than $\overrightarrow{x_l}$ are needed in determining the leader and the particle to be updated, we can ignore $\overrightarrow{x_l}$ in the proposed algorithm, and replace all the $\overrightarrow{x_l}'$ in Eq. (5) by the symbol $\overrightarrow{x_l}$ for simplicity. To summarize, the particle updating strategy in the proposed LMOCSO is

$$\begin{aligned}\overrightarrow{v_l}(t+1) &= r_0\overrightarrow{v_l}(t) + r_1(\overrightarrow{x_w}(t) - \overrightarrow{x_l}(t))\\ \overrightarrow{x_l}(t+1) &= \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1) + r_0(\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t))\end{aligned}, \tag{6}$$

where r_0 and r_1 are uniformly randomly distributed values in $[0, 1]$ according to [45], [49].

The details of the particle updating strategy in the proposed LMOCSO are given in Algorithm 1. To begin with, the fitness of each particle in the current population P is calculated, which adopts the shift based density estimation (SDE) strategy [50]. The fitness of a particle $\mathbf{p}$

is defined as the minimum SDE based distance between the particle and others in the population, i.e.,

$$Fitness(\mathbf{p}) = \min_{\mathbf{q}\in P\setminus\{\mathbf{p}\}} \sqrt{\sum_{i=1}^{M}(\max\{0, f_i(\vec{q}) - f_i(\vec{p})\})^2}, \quad (7)$$

where $f_i(\vec{p})$ denotes the i-th objective value of $\mathbf{p}$ and M denotes the number of objectives. The SDE strategy can evaluate the quality of a solution in terms of both convergence and diversity, and has been widely used in many MOEAs [38], [51]. Therefore, we use the SDE based distance in the proposed LMOCSO to quantitatively measure the convergence and diversity of each particle in the population.

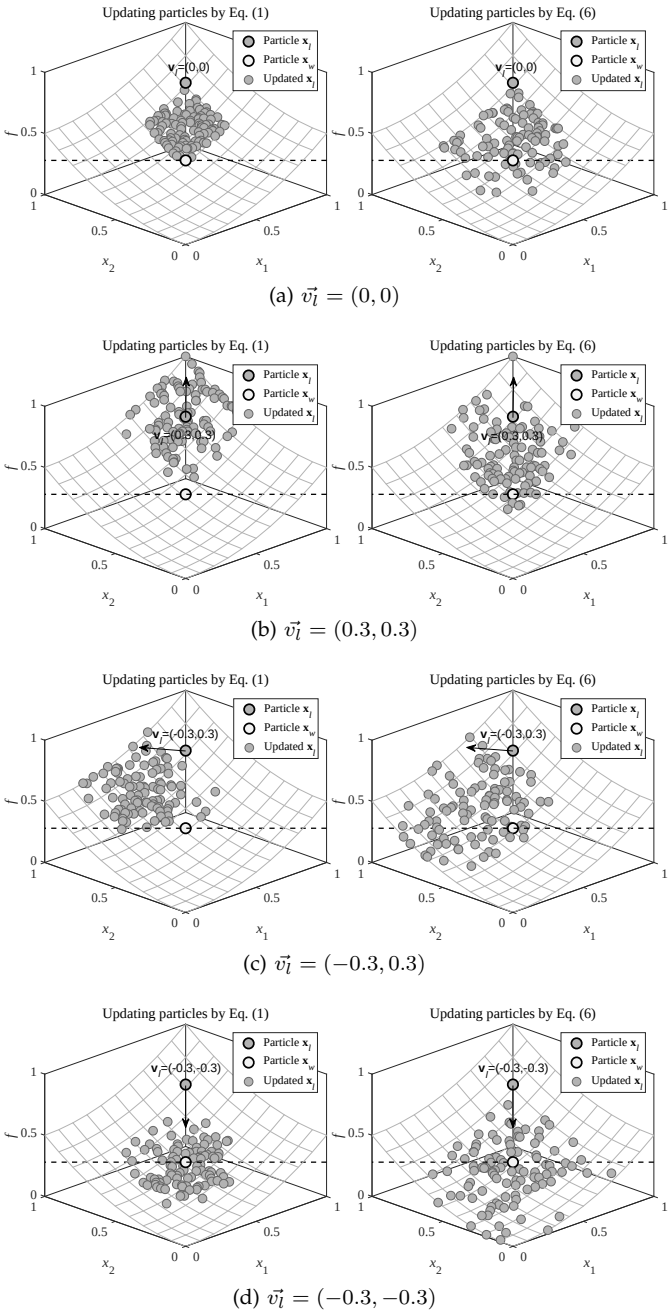
Afterwards, two particles are randomly picked up from the current population P, and the one with smaller fitness value $\mathbf{x}_l$ is updated by learning from the other one $\mathbf{x}_w$ by Eq. (6). Besides, to further improve the performance of LMOCSO in escaping from local optimum, both $\mathbf{x}_w$ and the updated $\mathbf{x}_l$ are slightly mutated by polynomial mutation [26]. Finally, the mutated $\mathbf{x}_w$ and $\mathbf{x}_l$ are put into the new population P', and this procedure repeats until all the particles in the current population P are visited.

B. Analysis

In comparison to Eq. (1), it can be found that Eq. (6) has an additional component $r_0(\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t))$. If we regard $\overrightarrow{x_w}(t) - \overrightarrow{x_l}(t)$ as 'first derivative' or 'velocity', then $\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t)$ can be regarded as 'second derivative' or 'acceleration'. As a matter of fact, both second derivative and acceleration can improve the search efficiency in solving optimization problems, where the second derivative has shown effectiveness in the Gauss-Newton algorithm [52], and the acceleration has also been verified to be useful in metaheuristics [53].

To investigate the effect of $\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t)$ in search dynamics, we assume a minimization problem $f(x_1, x_2) = (x_1^2 + x_2^2)/2$ and two particles $\overrightarrow{x_l} = (0.8, 0.8)$, $\overrightarrow{x_w} = (0.5, 0.5)$. Fig. 3 shows 100 particles obtained by updating $\mathbf{x}_l$ according to $\mathbf{x}_w$ by Eq. (1) and Eq. (6), in the case when $\overrightarrow{v_l}$ is set to $(0, 0)$, $(0.3, 0.3)$, $(-0.3, 0.3)$, and $(-0.3, -0.3)$, respectively. From the figure, two observations can be made. First, Eq. (6) has a higher search efficiency than Eq. (1), since the particles obtained by Eq. (6) spread more widely than those obtained by Eq. (1). Second, Eq. (1) can obtain particles better than the leader $\mathbf{x}_w$ only when the velocity $\overrightarrow{v_l}$ is set to the same direction to the difference between $\overrightarrow{x_l}$ and $\overrightarrow{x_w}$ (i.e., $\overrightarrow{v_l} = (-0.3, -0.3)$), whereas Eq. (6) can obtain particles better than the leader $\mathbf{x}_w$ regardless of the direction of $\overrightarrow{v_l}$.

To further demonstrate the advantages of $\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t)$ in Eq. (6), we extend the above example to four widely adopted benchmark SOPs [28], i.e., the Sphere function, the Schwefel's problem 2.21, the Rastrigin's function, and the Ackley's function. We randomly initialize two particles with 100 decision variables for each



(a) $\vec{v_l} = (0, 0)$

(b) $\vec{v_l} = (0.3, 0.3)$

(c) $\vec{v_l} = (-0.3, 0.3)$

(d) $\vec{v_l} = (-0.3, -0.3)$

Fig. 3. 100 particles obtained by updating $\mathbf{x}_l$ according to $\mathbf{x}_w$ by Eq. (1) and Eq. (6) with different settings of $\vec{v_l}$.

SOP, and generate 100 particles by updating the particle with a worse objective value according to the better one by Eq. (1) and Eq. (6). Then, we quantify the diversity of the 100 particles in decision space by the diversity measure in [45]. The statistical results are listed in Table I, where each test is run for 30 times and the mean of the metric values is recorded. It is obvious that the particles obtained by Eq. (6) have significantly better diversity (i.e., a larger metric value) than those obtained by Eq. (1) on the four SOPs.

To confirm whether the enhanced diversity can improve the search performance, the original CSO with Eq. (1) and the CSO with Eq. (6) are compared on the

TABLE I
DIVERSITY OF 100 PARTICLES OBTAINED BY UPDATING A PARTICLE ACCORDING TO A LEADER BY EQ. (1) AND EQ. (6) ON FOUR SOPS.

Method	Sphere function	Schwefel's problem 2.21	Rastrigin's function	Ackley's function
Eq. (1)	0.8060	0.7626	0.8724	0.9378
Eq. (6)	0.9942	0.9369	1.0672	1.1526

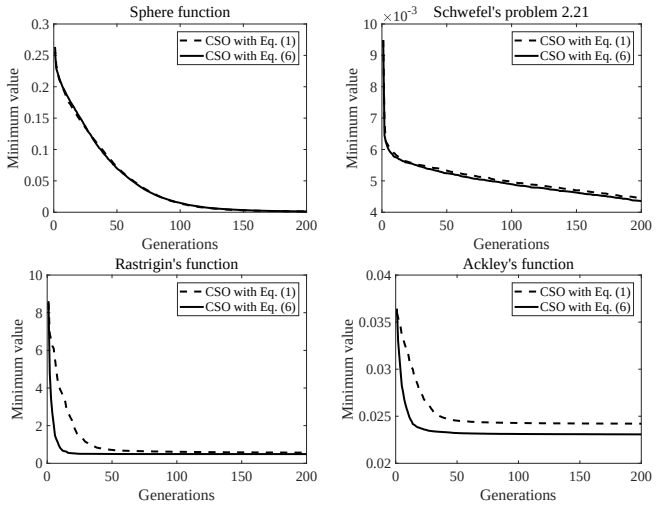


Fig. 4. Convergence profiles of CSO with Eq. (1) and CSO with Eq. (6) on four SOPs.

above four SOPs. The convergence profiles of the two algorithms averaged over 30 runs are plotted in Fig. 4, where the number of decision variables is set to 100, the population size is set to 100, and the number of generations is set to 200. It can be seen from Fig. 4 that the CSO with Eq. (1) has similar performance to the CSO with Eq. (6) on the Sphere function and the Schwefel's problem 2.21, whereas the CSO with Eq. (6) converges faster than the CSO with Eq. (1) on the Rastrigin's function and the Ackley's function. As a consequence, the advantages of $\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t)$ in Eq. (6) can be confirmed.

C. Procedure of LMOCSO

LMOCSO has a quite simple procedure, the details of which are presented in Algorithm 2. It begins with the initialization of a random population P and a set of uniformly distributed reference vectors R. In each generation of LMOCSO, the particles in the current population P are updated by the proposed particle updating strategy, then the environmental selection is carried out on the combination of P and the updated population P'. Note that since LMOCSO adopts a competitive mechanism in updating particles, it does not need to store or update personal-best particles or global-best particles as required in PSO algorithms.

LMOCSO adopts the environmental selection strategy of RVEA [54] to select the particles for the next generation from $P \cup P'$. This environmental selection strategy first associates each particle in $P \cup P'$ to its closest reference vector in R according to the angles between particles and reference vectors, then selects one particle with the best angle-penalized distance (APD) among all the particles associated with the same reference vector. The APD value of particle $\mathbf{p}$ with respect to reference vector $\vec{r}$ can be mathematically defined as

$$APD(\mathbf{p},\vec{r}) = (1+M\cdot(\frac{t}{t_{max}})^{\alpha}\cdot\frac{\langle\vec{f}(\vec{p}),\vec{r}\rangle}{\min_{\vec{s}\in R,\vec{s}\neq\vec{r}}\langle\vec{s},\vec{r}\rangle})\cdot\|\vec{f}(\vec{p})\|, \tag{8}$$

where $\vec{f}(\vec{p})$ denotes the objective vector of $\mathbf{p}$, M denotes the number of objectives, t denotes the generation index, t_{max} denotes the maximal number of generations, α is a penalty parameter, and $\langle\vec{s},\vec{r}\rangle$ denotes the angle between vectors $\vec{s}$ and $\vec{r}$. Such a reference vector guided selection strategy takes both convergence and diversity into account, and a set of well-converged particles with the same uniform distribution to the reference vectors are expected to be obtained. As a result, LMOCSO uses the new particle updating strategy to improve the convergence, and adopts the environmental selection in RVEA to maintain the diversity.

Algorithm 2: Procedure of LMOCSO

Input: N (population size)
Output: P (final population)
$P \leftarrow RandomInitialization(N)$;
$R \leftarrow UniformReferenceVector(N)$;
while *termination criterion not fulfilled* **do**
  $P' \leftarrow UpdatingParticle(P)$;
  $P \leftarrow RVEA\_EnvironmentalSelection(P \cup P', R)$;
return P;

IV. EXPERIMENTAL STUDIES

In this section, several experiments are conducted to investigate the performance of LMOCSO in solving large-scale MOPs. First, the effectiveness of the particle updating strategy in LMOCSO is empirically verified. Then, the proposed LMOCSO is compared to several state-of-the-art MOEAs on large-scale benchmark MOPs. Afterwards, the scalability of LMOCSO is studied by being tested on MOPs with the number of decision variables ranging from 100 to 5000. Finally, the proposed LMOCSO is applied to train a neural network with thousands of weights. All the experiments are performed on PlatEMO [55].

Four state-of-the-art MOEAs are involved in the experiments, namely, WOF-NSGA-II [24], LMEA [11], MMOPSO [37], and IM-MOEA [56]. WOF-NSGA-II is a decision variable grouping and problem transformation based MOEA for solving large-scale MOPs. LMEA is a decision variable clustering based MOEA tailored for large-scale MOPs, which has been verified to be capable

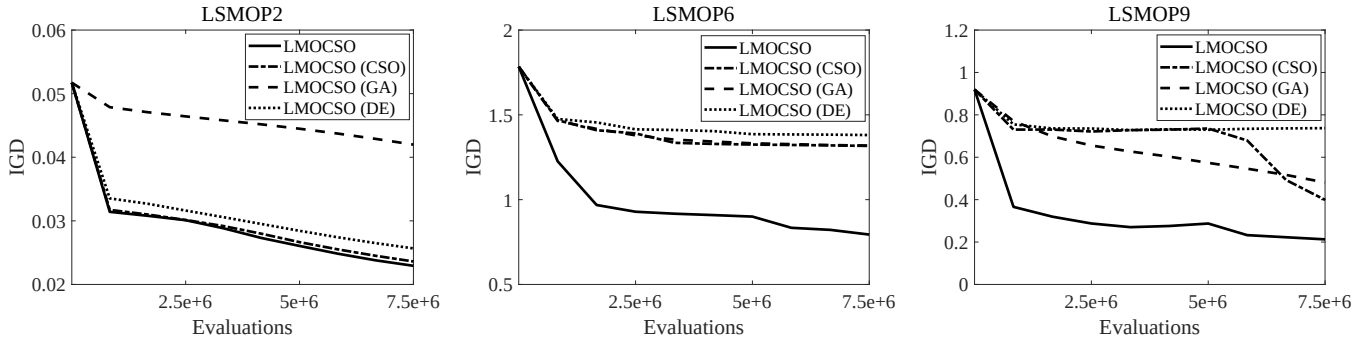


Fig. 5. Convergence profiles of IGD values obtained by LMOCSO and its three variants on 3-objective LSMOP2, LSMOP6, and LSMOP9 with 500 decision variables.

of solving MOPs with up to 5000 decision variables. MMOPSO is a multi-objective PSO algorithm, it uses multiple search strategies for updating particles in the population, and adopts genetic operators in updating the archive. IM-MOEA is an estimation of distribution algorithm that employs a Gaussian process based inverse modeling, which has shown promising potential in solving large-scale MOPs.

The large-scale multi-objective test suite [28] is adopted as the test problems, i.e., LSMOP1–LSMOP9. The LSMOP test suite is designed based on large-scale SOPs [57], by following four basic principles as suggested in [58] and [59] for guaranteeing the extensibility and generality of the problems. Among the nine LSMOP problems, there exist linear variable linkage (LSMOP1–LSMOP4) and nonlinear variable linkage (LSMOP5–LSMOP9) on the Pareto set, as well as linear Pareto front (LSMOP1–LSMOP4), concave Pareto front (LSMOP5–LSMOP8), and disconnected Pareto front (LSMOP9).

A. Experimental Settings

Algorithms: For fair comparisons, all parameters of the compared algorithms are tuned for a relatively good performance. To be specific, the penalty parameter α of APD in LMOCSO is set to 2. The number of evaluations for each optimization of original problem t_1, the number of evaluations for each optimization of transformed problem t_2, the number of chosen solutions q, the number of groups γ, and the ratio of evaluations for optimization of both original and transformed problems δ in WOF-NSGA-II are set to 1000, 1000, 2, 3 and 0.7, respectively. The number of selected solutions for decision variable clustering $nSel$, the number of perturbations on each solution for decision variable clustering $nPer$, and the number of selected solutions for decision variable interaction analysis $nCor$ in LMEA are set to 5, 5 and 6, respectively. The number of reference vectors K and the model group size L in IM-MOEA are set to 10 and 3, respectively. The simulated binary crossover and polynomial mutation are adopted to generate offsprings in WOF-NSGA-II, LMEA, and the archive of MMOPSO; the probability of crossover is set to 1, the probability of mutation is set to $1/D$ (D denotes the number of decision variables), and the distribution index of them is set to 20.

Population size: The population size is set to the same in all the compared MOEAs, which is 300 for bi-objective MOPs and 496 for three-objective MOPs. In addition, the Das and Dennis's systematic approach [60] is adopted to generate the same number of uniformly distributed reference vectors to be used in MMOPSO and LMOCSO.

Problems: For LSMOP1–LSMOP9, the number of subcomponents in each variable group n_k is set to 5, the number of objectives M is set to 2 and 3, and the number of decision variables D is varied from 100 to 5000.

Termination criterion: The number of evaluations is adopted as the termination criterion for all compared MOEAs, which is set to $15000 \times D$.

Performance metric: The inverted generational distance (IGD) [61] and hypervolume (HV) [62] are adopted to assess the performance of MOEAs in the experiment. For calculating IGD, roughly 10000 reference points are sampled on each Pareto front of LSMOP1–LSMOP9 by the methods suggested in [63]. All the tests are run for 30 times independently, and the mean and standard deviation of the IGD values are recorded. Besides, the Wilcoxon rank sum test with a significance level of 0.05 is adopted to perform statistical analysis on the experimental results, where the symbols '$+$', '$-$' and '$\approx$' indicate that the result by another MOEA is significantly better, significantly worse and statistically similar to that obtained by LMOCSO, respectively.

B. Effectiveness of the Particle Updating Strategy in LMOCSO

To verify the effectiveness of the particle updating strategy in LMOCSO, it is compared with three of its variants, in which the proposed particle updating strategy is replaced by the operators in original CSO (i.e., replacing Eq. (6) by Eq. (1)), genetic algorithm (GA, i.e., simulated binary crossover [25] and polynomial mutation [26]), and differential evolution (DE) [64], respectively. Fig. 5 plots the convergence profiles of IGD values obtained by LMOCSO and its three variants on 3-objective LSMOP2, LSMOP6, and LSMOP9 with 500 decision variables, averaged over 30 runs.

TABLE II
IGD VALUES OF WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, AND LMOCSO ON 2-OBJECTIVE LSMOP1–LSMOP9, WHERE THE BEST RESULT ON EACH TEST INSTANCE IS SHOWN IN A GRAY BACKGROUND.

Problem	Dec.	WOF-NSGA-II	LMEA	MMOPSO	IM-MOEA	LMOCSO
LSMOP1	100	5.5582e-1 (1.80e-1) −	1.3448e-2 (3.53e-3) −	1.5594e-2 (5.77e-3) −	1.4631e-2 (1.20e-2) −	1.1983e-3 (2.57e-6)
	200	5.0097e-1 (1.57e-1) −	1.4375e-2 (3.75e-3) −	4.1558e-3 (8.41e-4) −	6.9477e-3 (3.85e-4) −	1.2227e-3 (3.59e-6)
	500	3.3081e-1 (1.60e-1) −	2.1068e-2 (7.96e-3) −	8.8283e-3 (5.57e-4) −	1.1642e-2 (2.23e-4) −	1.7642e-3 (7.19e-5)
LSMOP2	100	4.2245e-2 (1.17e-2) −	8.5442e-2 (6.66e-2) −	2.6094e-2 (3.18e-3) −	9.2006e-3 (4.28e-4) −	5.6153e-3 (2.03e-3)
	200	2.5562e-2 (7.85e-3) −	9.1816e-2 (3.69e-2) −	3.9824e-2 (1.30e-2) −	7.6005e-3 (2.51e-4) −	2.9438e-3 (2.16e-4)
	500	1.2078e-2 (5.54e-3) −	4.8756e-2 (1.85e-2) −	3.5184e-2 (1.02e-3) −	6.7890e-3 (7.14e-5) −	2.4341e-3 (1.55e-4)
LSMOP3	100	6.0887e-1 (1.63e-1) −	9.0652e-1 (2.51e-1) −	2.6939e+0 (1.40e+0) −	5.2798e-1 (6.06e-2) +	5.7684e-1 (2.46e-1)
	200	6.0860e-1 (1.62e-1) +	8.3645e-1 (2.26e-1) −	2.6966e+0 (1.20e+0) −	5.2893e-1 (3.86e-2) +	6.1973e-1 (1.75e-1)
	500	6.1289e-1 (2.50e-1) ≈	1.0303e+0 (4.16e-1) −	1.9144e+0 (3.14e-1) −	5.3077e-1 (3.90e-2) +	7.0696e-1 (2.52e-4)
LSMOP4	100	7.6558e-2 (2.09e-2) −	8.1623e-2 (2.18e-2) −	3.6941e-2 (6.67e-3) −	1.2485e-2 (4.34e-4) −	2.2257e-3 (5.05e-4)
	200	2.9916e-2 (8.20e-3) −	4.8666e-2 (1.28e-2) −	3.1740e-2 (1.08e-2) −	8.5148e-3 (3.63e-4) −	1.5547e-3 (9.20e-5)
	500	1.8301e-2 (7.51e-3) −	2.7228e-2 (1.02e-2) −	2.8345e-2 (3.19e-3) −	6.9118e-3 (2.90e-4) −	1.3840e-3 (9.35e-5)
LSMOP5	100	2.9193e-1 (1.02e-1) −	4.4736e-1 (2.53e-1) −	5.0477e-3 (7.66e-4) −	4.2308e-2 (1.12e-2) −	1.3684e-3 (1.61e-5)
	200	1.8013e-1 (1.30e-1) −	4.0115e-1 (1.95e-1) −	4.4235e-3 (5.74e-4) −	3.7999e-2 (1.12e-2) −	1.5162e-3 (1.93e-5)
	500	9.5035e-2 (6.14e-2) −	3.3899e-1 (1.79e-1) −	4.9614e-3 (1.30e-3) −	3.7528e-2 (9.01e-3) −	3.3919e-3 (1.03e-4)
LSMOP6	100	4.7622e-1 (1.77e-1) +	6.8908e-1 (2.43e-1) ≈	4.5427e-1 (2.49e-2) +	3.1482e-1 (6.71e-2) +	7.4224e-1 (8.95e-5)
	200	5.2814e-1 (1.87e-1) −	6.2166e-1 (2.06e-1) −	3.9931e-1 (3.53e-2) −	3.2237e-1 (6.03e-2) −	2.8641e-1 (5.10e-2)
	500	3.1331e-1 (1.27e-1) +	5.2873e-1 (2.85e-1) ≈	3.0790e-1 (6.26e-2) +	2.6772e-1 (6.98e-2) +	5.4755e-1 (4.23e-2)
LSMOP7	100	7.5753e-1 (2.07e-1) ≈	1.2656e+0 (3.28e-1) −	1.7250e+0 (4.43e-1) −	9.2268e-1 (1.25e-1) −	9.0244e-1 (3.86e-1)
	200	1.0481e+0 (3.23e-1) −	1.2666e+0 (3.83e-1) −	1.9288e+0 (7.02e-1) −	1.0595e+0 (9.00e-2) −	7.1905e-1 (5.88e-2)
	500	9.6110e-1 (4.35e-1) ≈	1.0598e+0 (5.46e-1) ≈	1.3515e+0 (2.17e-1) −	1.4175e+0 (1.12e-1) −	7.9114e-1 (5.10e-1)
LSMOP8	100	2.3081e-1 (1.06e-1) −	8.4453e-2 (2.58e-2) −	4.9862e-2 (4.00e-3) −	2.5794e-2 (5.74e-3) −	5.9939e-3 (3.35e-3)
	200	1.2096e-1 (8.02e-2) −	5.6156e-2 (1.57e-2) −	4.6374e-2 (9.20e-4) −	3.3421e-2 (4.71e-3) −	7.4335e-3 (9.36e-4)
	500	4.3341e-2 (2.07e-2) −	3.5016e-2 (1.31e-2) −	2.6209e-2 (1.68e-3) −	2.3163e-2 (4.34e-3) −	1.8031e-2 (3.87e-3)
LSMOP9	100	7.5941e-1 (2.03e-1) −	4.9554e-1 (1.32e-1) −	8.1840e-2 (5.43e-2) +	6.2293e-1 (3.46e-1) −	1.9221e-1 (2.37e-1)
	200	7.5941e-1 (2.03e-1) −	4.7881e-1 (1.24e-1) −	3.1129e-1 (4.46e-2) −	5.6584e-1 (3.64e-1) −	1.1393e-1 (2.22e-1)
	500	7.0877e-1 (2.86e-1) −	4.2956e-1 (1.61e-1) −	2.0055e-1 (6.68e-3) −	4.1775e-1 (2.95e-1) −	8.8080e-3 (8.63e-3)
+/ − / ≈		3/21/3	0/24/3	3/24/0	5/22/0	

'+', '−' and '≈' indicate that the result is significantly better, significantly worse and statistically similar to that of LMOCSO, respectively.

As shown in Fig. 5, for LSMOP2 with a simple landscape posing little challenge to algorithms in finding well-converged solutions, LMOCSO and the variants based on original CSO and differential evolution can drive the population to converge to the Pareto front quickly, and they finally obtain similar IGD values. However, for LSMOP6 and LSMOP9 with complex landscape or Pareto front which are much more difficult to be solved than LSMOP2, LMOCSO converges significantly faster than the three variants. As a result, the advantages of the particle updating strategy in the proposed LMOCSO for solving large-scale MOPs can be verified.

C. Comparisons Between LMOCSO and Existing MOEAs

Table II lists the IGD values of the five compared MOEAs on 2-objective LSMOP1–LSMOP9 with 100, 200 and 500 decision variables. It is obvious that the proposed LMOCSO exhibits better overall performance than the other four compared MOEAs. Specifically, LMOCSO performs the best on 20 out of the 27 test instances, which is followed by IM-MOEA gaining 5 best results, WOF-NSGA-II gaining 1 best result, and MMOPSO gaining 1 best result. In terms of the Wilcoxon rank sum test, the proportion of test instances where LMOCSO performs significantly better than WOF-NSGA-II, LMEA, MMOPSO, and IM-MOEA is 21/27, 24/27, 24/27 and 22/27, respectively.

For further observation, Fig. 6 depicts the non-dominated solutions with the median IGD values obtained by the five compared MOEAs on 2-objective LSMOP4, LSMOP8, and LSMOP9 with 100 decision variables. For WOF-NSGA-II and LMEA based on genetic operators, it can be seen from the figure that they cannot obtain a set of well-converged solutions for the

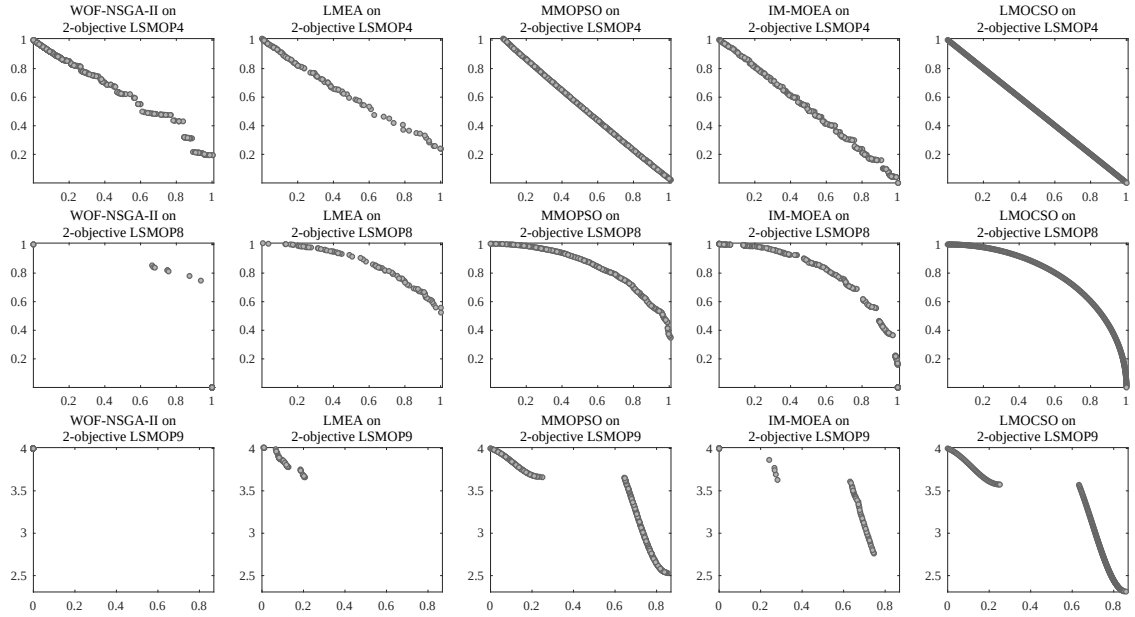


Fig. 6. Non-dominated solutions with the median IGD values obtained by WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, and LMOCSO on 2-objective LSMOP4, LSMOP8, and LSMOP9 with 100 decision variables.

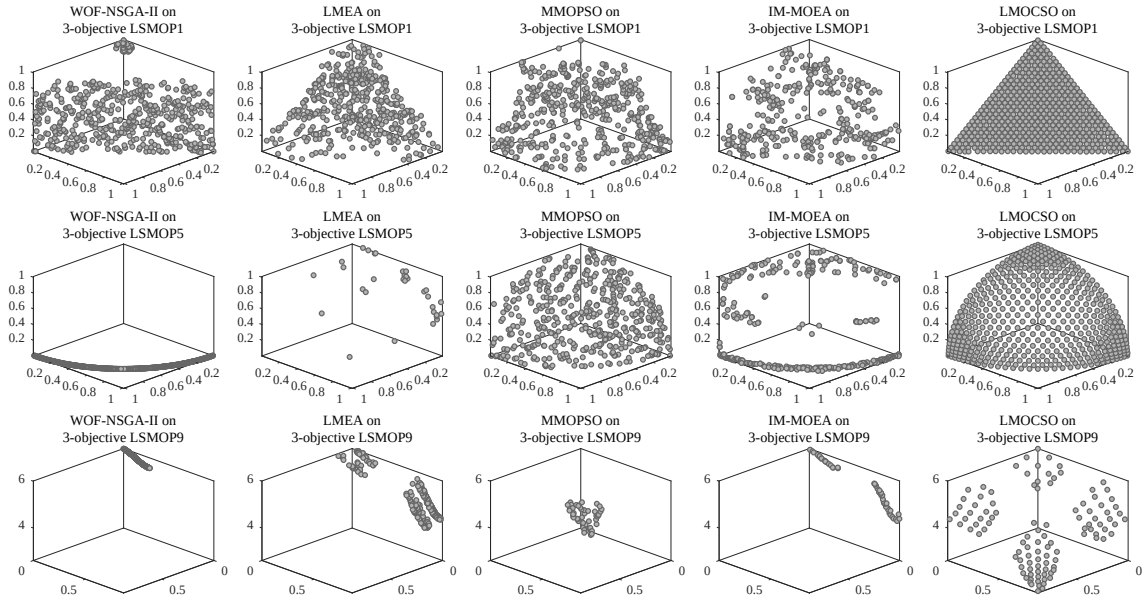


Fig. 7. Non-dominated solutions with the median IGD values obtained by WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, and LMOCSO on 3-objective LSMOP1, LSMOP5, and LSMOP9 with 100 decision variables.

three test problems. For IM-MOEA based on estimation of distribution algorithm, it can obtain a set of well-converged solutions for LSMOP4 and LSMOP8, but the diversity of the obtained solution sets are not satisfactory. For MMOPSO and LMOCSO based on swarm algorithm, both of them can obtain a set of solutions with good convergence and diversity for the three test problems, and the convergence of the solution sets obtained by LMOCSO is slightly better than those obtained by MMOPSO. Therefore, it can be concluded that swarm algorithms are promising in solving large-scale MOPs, and the particle updating strategy in the proposed LMOCSO can further accelerate the convergence of the algorithm.

Table III presents the IGD values of the five compared MOEAs on 3-objective LSMOP1–LSMOP9 with 100, 200 and 500 decision variables. Similar to the statistical results on 2-objective MOPs given in Table II, LMOCSO shows the best performance among the five compared algorithms, which performs the best on 20 out of the 27 test instances. Besides, the proportion of test

TABLE III
IGD VALUES OF WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, AND LMOCSO ON 3-OBJECTIVE LSMOP1–LSMOP9, WHERE THE BEST RESULT ON EACH TEST INSTANCE IS SHOWN IN A GRAY BACKGROUND.

Problem	Dec.	WOF-NSGA-II	LMEA	MMOPSO	IM-MOEA	LMOCSO
LSMOP1	100	2.0453e-1 (9.45e-2) −	4.0561e-2 (1.07e-2) −	1.2051e-1 (1.21e-2) −	1.2285e-1 (1.03e-2) −	1.8811e-2 (5.36e-4)
	200	1.6640e-1 (5.55e-2) −	4.0034e-2 (1.04e-2) −	1.3429e-1 (1.79e-2) −	1.4062e-1 (1.21e-2) −	1.9830e-2 (4.73e-4)
	500	1.5269e-1 (7.14e-2) −	3.8564e-2 (1.46e-2) −	1.7002e-1 (6.08e-3) −	1.5396e-1 (7.37e-3) −	2.6577e-2 (2.80e-3)
LSMOP2	100	1.5546e-1 (4.17e-2) −	5.2952e-2 (1.88e-2) −	1.4552e-1 (4.59e-3) −	1.0122e-1 (2.47e-3) −	3.8228e-2 (1.83e-3)
	200	5.4219e-2 (1.45e-2) −	8.0715e-2 (2.89e-2) −	1.0452e-1 (6.81e-3) −	7.7645e-2 (1.31e-3) −	2.5961e-2 (2.34e-3)
	500	5.0683e-2 (2.05e-2) −	4.0526e-2 (1.60e-2) −	5.6123e-2 (9.38e-4) −	4.9683e-2 (8.78e-4) −	2.0137e-2 (4.00e-4)
LSMOP3	100	3.7564e-1 (1.13e-1) ≈	7.9854e-1 (2.10e-1) −	4.0034e-1 (5.84e-2) −	5.1488e-1 (1.84e-2) −	3.7798e-1 (3.75e-2)
	200	4.1649e-1 (1.21e-1) ≈	8.3810e-1 (2.21e-1) −	3.3427e-1 (9.88e-2) +	5.6006e-1 (1.90e-2) −	4.2794e-1 (3.89e-2)
	500	3.9684e-1 (1.61e-1) +	7.9351e-1 (3.01e-1) −	5.4873e-1 (4.50e-2) −	5.9126e-1 (2.54e-2) −	4.8198e-1 (1.99e-2)
LSMOP4	100	1.5805e-1 (4.67e-2) −	1.0764e-1 (2.79e-2) −	1.4131e-1 (5.12e-3) −	1.9488e-1 (9.95e-3) −	5.0407e-2 (7.91e-3)
	200	1.8453e-1 (4.94e-2) −	7.0722e-2 (1.84e-2) −	1.6720e-1 (1.88e-2) −	1.4104e-1 (5.83e-3) −	3.8933e-2 (2.45e-3)
	500	1.2013e-1 (4.88e-2) −	3.9792e-2 (1.49e-2) −	1.5466e-1 (9.24e-3) −	9.0180e-2 (3.82e-3) −	2.6484e-2 (1.16e-3)
LSMOP5	100	3.6064e-1 (9.71e-2) −	1.6323e+0 (1.69e+0) −	5.7271e-2 (8.74e-3) −	1.5364e-1 (2.76e-2) −	2.2401e-2 (6.66e-5)
	200	2.8339e-1 (9.76e-2) −	2.2109e+0 (2.82e+0) −	8.0568e-2 (1.05e-2) −	1.5209e-1 (2.09e-2) −	2.2801e-2 (6.32e-5)
	500	2.6876e-1 (1.15e-1) −	6.1402e+0 (4.56e+0) −	2.3278e-1 (1.25e-2) −	1.7477e-1 (9.03e-3) −	2.6079e-2 (2.79e-4)
LSMOP6	100	8.5054e-1 (2.27e-1) −	1.0915e+0 (3.18e-1) −	9.8570e-1 (1.42e-1) −	6.3174e-1 (7.81e-2) +	7.3607e-1 (1.34e-1)
	200	9.8667e-1 (2.63e-1) ≈	1.4212e+0 (4.24e-1) −	1.2448e+0 (1.53e-1) −	7.8131e-1 (5.05e-2) +	8.9335e-1 (3.05e-1)
	500	1.0633e+0 (4.30e-1) −	1.3302e+0 (5.36e-1) −	1.0677e+0 (2.40e-1) −	1.3058e+0 (3.95e-2) −	7.4746e-1 (3.72e-1)
LSMOP7	100	7.6341e-1 (2.11e-1) −	1.9379e+0 (7.10e-1) −	6.7572e-1 (1.70e-1) −	6.0272e-1 (3.26e-2) −	4.8582e-1 (1.60e-1)
	200	7.7781e-1 (2.09e-1) +	1.4201e+0 (4.27e-1) −	7.4328e-1 (1.49e-1) +	5.8149e-1 (2.15e-2) +	8.0447e-1 (2.29e-1)
	500	7.3933e-1 (2.99e-1) +	7.9726e-1 (2.99e-1) +	8.2797e-1 (9.35e-2) +	5.7436e-1 (1.93e-2) +	8.6096e-1 (1.79e-1)
LSMOP8	100	3.3774e-1 (9.09e-2) −	1.4946e-1 (6.27e-2) −	1.5848e-1 (1.21e-2) −	1.5073e-1 (6.27e-3) −	4.0666e-2 (4.31e-3)
	200	2.4093e-1 (1.00e-1) −	1.3231e-1 (3.68e-2) −	1.1885e-1 (2.48e-3) −	1.4124e-1 (6.48e-3) −	4.8776e-2 (4.22e-3)
	500	1.8257e-1 (7.39e-2) −	8.1827e-2 (3.09e-2) −	7.5988e-2 (1.40e-3) −	1.1378e-1 (6.21e-3) −	4.7086e-2 (2.21e-3)
LSMOP9	100	1.0913e+0 (3.00e-1) −	4.6233e-1 (1.23e-1) −	1.0224e+0 (1.18e-1) −	5.8487e-1 (1.93e-3) −	7.1758e-2 (6.56e-2)
	200	1.0735e+0 (2.86e-1) −	4.1952e-1 (1.09e-1) −	6.5856e-1 (2.97e-2) −	5.7846e-1 (3.55e-3) −	5.8015e-2 (1.80e-3)
	500	9.8137e-1 (4.72e-1) −	3.8381e-1 (1.45e-1) −	3.8890e-1 (1.50e-2) −	5.6482e-1 (1.21e-2) −	2.1268e-1 (3.07e-2)
+/ − / ≈		3/21/3	1/26/0	3/24/0	4/23/0	

'+', '−' and '≈' indicate that the result is significantly better, significantly worse and statistically similar to that of LMOCSO, respectively.

instances where LMOCSO performs significantly better than WOF-NSGA-II, LMEA, MMOPSO, and IM-MOEA is 21/27, 26/27, 24/27 and 23/27, respectively. Fig. 7 plots the non-dominated solutions with the median IGD values obtained by the five compared MOEAs on 3-objective LSMOP1, LSMOP5, and LSMOP9 with 100 decision variables. It can be found from the figure that LMOCSO can obtain a solution set with good convergence and diversity on all the three MOPs, whereas the solution sets obtained by the other four algorithms are not satisfactory. Although the solution set obtained by LMEA on LSMOP1 and the solution set obtained by MMOPSO on LSMOP5 seem to converge to the Pareto front, the diversity of these two solution sets is significantly worse than those obtained by LMOCSO. As a consequence, the above experimental results demonstrate the effectiveness of the proposed LMOCSO in solving large-scale MOPs.

D. Scalability of LMOCSO with Respect to the Number of Decision Variables

To further investigate the scalability of the proposed LMOCSO with respect to the number of decision variables, we examine the performance of LMOCSO on the problems with a larger number of decision variables. Fig. 8 shows the IGD values of LMOCSO and the other compared MOEAs on 3-objective LSMOP2, LSMOP6, and LSMOP9 averaged over 30 runs, where the number of decision variables D is set to 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000 and 5000, and the number of evaluations is set to $15000 \times D$ for all the MOEAs.

In general, LMOCSO exhibits similar performance on the same MOP with different numbers of decision variables, and it outperforms the compared MOEAs on most

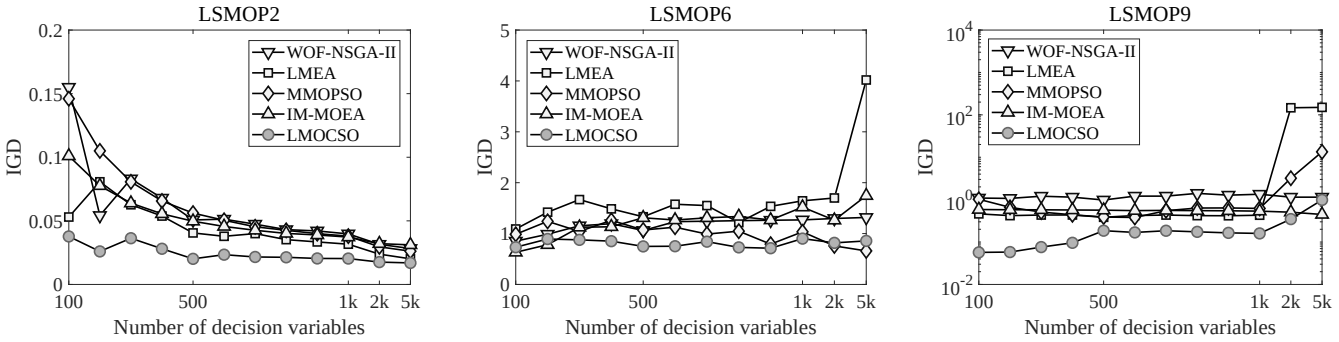


Fig. 8. IGD values of WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, and LMOCSO on 3-objective LSMOP2, LSMOP6, and LSMOP9 with the number of decision variables ranging from 100 to 5000.

TABLE IV
HV VALUES OF WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, AND LMOCSO FOR TRAINING NEURAL NETWORK, WHERE THE BEST RESULT ON EACH TEST INSTANCE IS SHOWN IN A GRAY BACKGROUND.

Dataset (No. of samples/ No. of features)	Dec.	WOF-NSGA-II	LMEA	MMOPSO	IM-MOEA	LMOCSO
Breast Cancer Diagnostic (569/30)	641	7.9557e-1 (3.47e-2)−	4.9957e-1 (1.54e-2)−	8.4338e-1 (1.18e-2)−	6.3032e-1 (1.68e-2)−	8.7127e-1 (9.44e-3)
Connectionist Bench Sonar (208/60)	1241	6.9876e-1 (8.03e-2)−	3.9154e-1 (1.27e-2)−	8.2863e-1 (1.30e-2)≈	5.8986e-1 (1.05e-2)−	8.3507e-1 (1.38e-2)
Hill Valley (606/100)	2041	4.6217e-1 (4.52e-2)−	3.1932e-1 (3.77e-2)−	5.6524e-1 (4.86e-2)≈	3.7711e-1 (2.87e-2)−	5.5515e-1 (1.40e-2)
Musk1 (476/166)	3361	6.7944e-1 (3.56e-2)−	3.6495e-1 (5.30e-3)−	8.0484e-1 (1.31e-2)−	5.4451e-1 (1.14e-2)−	8.2814e-1 (3.02e-3)
Madelon (2600/500)	10041	5.3560e-1 (3.48e-2)−	3.1590e-1 (4.54e-3)−	5.7023e-1 (2.76e-2)−	3.6803e-1 (1.17e-2)−	6.1149e-1 (2.46e-2)
+/ − / ≈		0/5/0	0/5/0	0/3/2	0/5/0	

'+', '−' and '≈' indicate that the result is significantly better, significantly worse and statistically similar to that of LMOCSO, respectively.

test instances. For LSMOP2 with a simple landscape, the IGD values obtained by LMOCSO are better than those obtained by the compared MOEAs. For LSMOP6 with a highly multi-modal landscape, the performance of LMOCSO is overall better than the compared MOEAs. As for LSMOP9 with a disconnected Pareto front, the IGD values obtained by LMOCSO are relatively small when the number of decision variables varies from 100 to 1000, but the performance of LMOCSO deteriorates on LSMOP9 with 2000 and 5000 decision variables. Therefore, it can be confirmed that the proposed LMOCSO has a good scalability with respect to the number of decision variables on most test instances, in the case when the number of evaluations is linearly related to the number of decision variables.

E. Applying LMOCSO to Neural Network Training

Finally, the proposed LMOCSO and four compared MOEAs are adopted to optimize the weights of a feedforward neural network with one hidden layer containing 20 neurons. The goal is to minimize both the complexity and classification error rate of the neural network [29], i.e.,

$$\begin{aligned} &\text{Minimize } f_1(\vec{x}) = \frac{1}{D}\sum_{i=1}^{D}|x_i| \\ &f_2(\vec{x}) = ErrorRate(\vec{x}) \end{aligned}, \tag{9}$$

where $\vec{x} = (x_1, x_2, \ldots, x_D)$ is the decision vector denoting all the weights of the neural network. That is, all the weights in the neural network are encoded in the solution, and the number of decision variables D equals to the number of weights. The decision space and objective space of the problem are $[-1, 1]^D$ and $[0, 1]^2$, respectively. As suggested in [65], each solution is fine tuned by gradient descent after being generated for better approximation performance.

Table IV lists the HV values obtained by the five compared MOEAs on five datasets chosen from the UCI machine learning repository [66], where each MOEA is

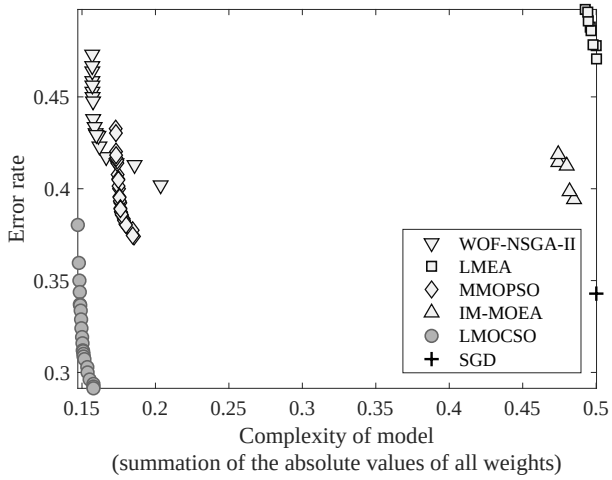


Fig. 9. Non-dominated solutions with the median HV values obtained by WOF-NSGA-II, LMEA, MMOPSO, IM-MOEA, LMOCSO, and SGD on the Madelon dataset.

run for 100 generations with a population size of 50, and the reference point for HV calculation is set to $(1.1, 1.1)$. The number of samples (i.e., training data) and the number of features (i.e., inputs) of each dataset are also presented in the leftmost column of Table IV. As can be seen from the table, LMOCSO exhibits better overall performance than the compared MOEAs in neural network training, which obtains the best HV value on four out of five datasets. Fig. 9 shows the non-dominated solutions with the median HV values obtained by the five compared MOEAs on the Madelon dataset. Besides, the result obtained by stochastic gradient descent (SGD) [67] is also shown in the figure, where the learning rate is set to 1, the momentum is set to 0.9, the mini-batch size is set to 20, and the number of epochs is set to 5000. It can be found that the models obtained by LMOCSO have significantly smaller complexity and error rate than those obtained by the compared MOEAs and SGD. As a consequence, it can be confirmed that the proposed LMOCSO is also effective in solving large-scale MOPs in applications.

V. Conclusions

This paper has proposed a CSO based efficient search for solving large-scale MOPs, called LMOCSO. In the proposed LMOCSO, a competitive mechanism is adopted to determine the particles to be updated, and a new particle updating strategy is developed to improve the search efficiency. Different from existing algorithms that focus on the modification of updating velocity, the proposed particle updating strategy suggests a two-stage strategy to update position. The experimental results have demonstrated that the proposed LMOCSO can better solve large-scale MOPs than several state-of-the-art MOEAs.

According to the experimental results given in Fig. 8, it can be found that the performance of the proposed LMOCSO needs to be further improved on large-scale MOPs with multi-modal landscapes (i.e., LSMOP3, LSMOP6, and LSMOP7). For example, the decision variable analysis method [10] or the decision variable clustering method [11] can be employed to reduce the search space, so that LMOCSO can escape from local optimum more easily. In addition, the environmental selection of the proposed LMOCSO is based on a set of reference vectors, which are overspecialized for the MOPs with regular Pareto fronts [68]. Hence it is desirable to adopt other environmental selection strategies for solving MOPs with irregular Pareto fronts [69].

References

[1] H. Ishibuchi and T. Murata, "Multiobjective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics-Part C*, vol. 28, no. 3, pp. 392–403, 1998.

[2] J. Handl, D. B. Kell, and J. Knowles, "Multiobjective optimization in bioinformatics and computational biology," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 2, pp. 279–292, 2007.

[3] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, "A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 321–344, 2013.

[4] J. B. Kollat, P. M. Reed, and R. Maxwell, "Many-objective groundwater monitoring network design using bias-aware ensemble kalman filtering, evolutionary optimization, and visual analytics," *Water Resources Research*, vol. 47, no. 2, 2011.

[5] J. W. Kruisselbrink, M. T. Emmerich, T. Bäck, A. Bender, A. P. IJzerman, and E. van der Horst, "Combining aggregation with Pareto optimization: A case study in evolutionary molecular design," in *Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization*, 2009, pp. 453–467.

[6] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011.

[7] S. K. Goh, K. C. Tan, A. Al-Mamun, and H. A. Abbass, "Evolutionary big optimization (BigOpt) of signals," in *Proceedings of the 2015 IEEE Congress on Evolutionary Computation*, 2015, pp. 3332–3339.

[8] Z. P. Stanko, T. Nishikawa, and S. R. Paulinski, "Large-scale multi-objective optimization for the management of seawater intrusion," in *Proceedings of the 2015 American Geophysical Union Fall Meeting*, 2015.

[9] K. Deb and C. Myburgh, "A population-based fast algorithm for a billion-dimensional resource allocation problem with integer variables," *European Journal of Operational Research*, vol. 261, no. 2, pp. 460–474, 2017.

[10] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, 2016.

[11] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, 2018.

[12] E. T. Oldewage, A. P. Engelbrecht, and C. W. Cleghorn, "The merits of velocity clamping particle swarm optimisation in high dimensional spaces," in *Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence*, 2017.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[14] Q. Zhang and H. Li, "MOEA/D: A multi-objective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[15] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, "An indicator based multi-objective evolutionary algorithm with reference point adaptation for better versatility," *IEEE Transactions on Evolutionary Computation*, 2017, in press.

[16] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[17] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Proceedings of the 2009 IEEE Congress on Evolutionary Computation*, 2009, pp. 1546–1553.

[18] ——, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[19] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computatio*, 2013, pp. 2758–2765.

[20] S. Kukkonen and J. Lampinen, "GDE3: the third evolution step of generalized differential evolution," in *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, 2005, pp. 443–450 Vol.1.

[21] A. W. Iorio and X. Li, "A cooperative coevolutionary multiobjective algorithm using non-dominated sorting," in *Proceedings of the 2004 Genetic and Evolutionary Computation Conference*, 2004, pp. 537–548.

[22] A. Song, Q. Yang, W. N. Chen, and J. Zhang, "A random-based dynamic grouping strategy for large scale multi-objective optimization," in *Proceedings of the 2016 IEEE Congress on Evolutionary Computation*, 2016, pp. 468–475.

[23] B. Cao, J. Zhao, Z. Lv, and X. Liu, "A distributed parallel cooperative coevolutionary multi-objective evolutionary algorithm for large-scale optimization," *IEEE Transactions on Industrial Informatics*, 2017, in press.

[24] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multi-objective optimization based on problem transformation," *IEEE Transactions on Evolutionary Computation*, 2017, in press.

[25] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 4, pp. 115–148, 1995.

[26] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and Informatics*, vol. 26, no. 4, pp. 30–45, 1996.

[27] Y. Wang, H. Liu, H. Long, Z. Zhang, and S. Yang, "Differential evolution with a new encoding mechanism for optimizing wind farm layout," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1040–1054, 2018.

[28] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4108–4121, 2017.

[29] Y. Jin, T. Okabe, and B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms," in *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, 2004, pp. 1–8.

[30] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.

[31] C. C. Coello and M. Lecsuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, 2002, pp. 1051–1056.

[32] M. Sierra and C. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance," in *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, 2005, pp. 505–519.

[33] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. C. Coello, F. Luna, and E. Alba, "SMPSO: A new PSO-based metaheuristic for multi-objective optimization," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making*, 2009, pp. 66–73.

[34] Y. Cooren, M. Clerc, and P. Siarry, "MO-TRIBES, an adaptive multiobjective particle swarm optimization algorithm," *Computational Optimization & Applications*, vol. 49, no. 2, pp. 379–400, 2011.

[35] S. Zapotecas Martínez and C. A. Coello Coello, "A multi-objective particle swarm optimizer based on decomposition," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, 2011, pp. 69–76.

[36] W. Hu and G. G. Yen, "Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 1–18, 2015.

[37] Q. Lin, J. Li, Z. Du, J. Chen, and Z. Ming, "A novel multi-objective particle swarm optimization with multiple search strategies," *European Journal of Operational Research*, vol. 247, no. 3, pp. 732–744, 2015.

[38] 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 Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 32–46, 2018.

[39] C. Yue, B. Qu, and J. Liang, "A multi-objective particle swarm optimizer using ring topology for solving multimodal multi-objective problems," *IEEE Transactions on Evolutionary Computation*, 2017, in press.

[40] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, "A competitive mechanism based multi-objective particle swarm optimizer with fast convergence," *Information Sciences*, vol. 427, pp. 63–76, 2018.

[41] J. Maltese, B. Ombuki-Berman, and A. Engelbrecht, "Co-operative vector-evaluated particle swarm optimization for multi-objective optimization," in *Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence*, 2015, pp. 1294–1301.

[42] A. Atashpendar, B. Dorronsoro, G. Danoy, and P. Bouvry, "A scalable parallel cooperative coevolutionary PSO algorithm for multi-objective optimization," *Journal of Parallel & Distributed Computing*, vol. 112, no. 2, pp. 111–125, 2017.

[43] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimization method in multiobjective problems," in *Proceedings of the 2002 ACM Symposium on Applied Computing*, 2002, pp. 603–607.

[44] F. Van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[45] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization." *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[46] S. Z. Zhao, J. J. Liang, P. N. Suganthan, and M. F. Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization," in *Proceedings of the 2010 IEEE Congress on Evolutionary Computation*, 2010, pp. 3845–3852.

[47] P. Mohapatra, K. N. Das, and S. Roy, "A modified competitive swarm optimizer for large scale optimization problems," *Applied Soft Computing*, vol. 59, pp. 340–362, 2017.

[48] E. Ozcan and C. K. Mohan, "Analysis of a simple particle swarm optimization system," in *Intelligent Engineering Systems Through Artificial Neural Networks*, 1998, pp. 253–258.

[49] E. T. Oldewage, A. P. Engelbrecht, and C. W. Cleghorn, "The importance of component-wise stochasticity in particle swarm optimization," in *Proceedings of the 2018 International Conference on Swarm Intelligence*, 2018, pp. 264–276.

[50] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 348–365, 2014.

[51] B. Li, K. Tang, J. Li, and X. Yao, "Stochastic ranking algorithm for many-objective optimization based on multiple indicators," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 924–938, 2016.

[52] A. Björck, *Numerical methods for least squares problems*. SIAM, Philadelphia, 1996.

[53] Z. Beheshti and S. M. Hj. Shamsuddin, "CAPSO: Centripetal accelerated particle swarm optimization," *Information Sciences*, vol. 258, no. 3, pp. 54–79, 2014.

[54] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, 2016.

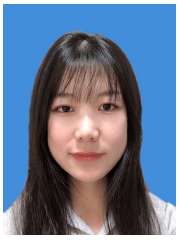
[55] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[56] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, "A multiobjective evolutionary algorithm using gaussian process-based inverse

modeling," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, 2015.
[57] M. N. Omidvar, X. Li, and K. Tang, "Designing benchmark problems for large-scale continuous optimization," *Information Sciences*, vol. 316, pp. 419–436, 2015.
[58] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, 2005, pp. 105–145.
[59] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit." *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.
[60] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *Siam Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.
[61] A. Zhou, Y. Jin, Q. Zhang, and B. Sendhoff, "Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion," in *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, 2006, pp. 892–899.
[62] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.
[63] Y. Tian, X. Xiang, X. Zhang, R. Cheng, and Y. Jin, "Sampling reference points on the Pareto fronts of benchmark multi-objective optimization problems," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation*, 2018, in press.
[64] K. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: A practical approach to global optimization*. Springer Science & Business Media, 2006.
[65] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 3, pp. 397–415, 2008.
[66] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[67] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[68] H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, "Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 169–190, 2017.
[69] Y. Hua, Y. Jin, and K. Hao, "A clustering-based adaptive evolutionary algorithm for multiobjective optimization with irregular pareto fronts," *IEEE Transactions on Cybernetics*, 2018, in press.

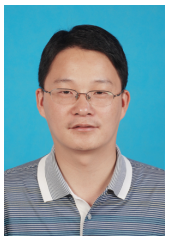
Ye Tian received the B.Sc., M.Sc., and Ph.D. degrees from Anhui University, Hefei, China, in 2012, 2015, and 2018, respectively.

He is currently a Lecturer with the Institutes of Physical Science and Information Technology, Anhui University, Hefei, China. His current research interests include multi-objective optimization methods and their application. He is the recipient of the 2018 IEEE Transactions on Evolutionary Computation Outstanding Paper Award.



Xiutao Zheng received the B.Sc. degree from Hebei University of Economics and Business, Shijiazhuang, China, in 2015, where she is currently pursing the M.Sc. degree with the School of Computer Science and Technology, Anhui University, Hefei, China.

Her current research interests include particle swarm optimization and evolutionary multi-objective optimization.



Xingyi Zhang received the B.Sc. degree from Fuyang Normal College, Fuyang, China, in 2003, and the M.Sc. and Ph.D. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2006 and 2009, respectively.

He is currently a Professor with the School of Computer Science and Technology, Anhui University, Hefei, China. His current research interests include unconventional models and algorithms of computation, evolutionary multi-objective optimization, and membrane computing. He is the recipient of the 2018 IEEE Transactions on Evolutionary Computation Outstanding Paper Award.

Yaochu Jin (M'98-SM'02-F'16) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Bochum, Germany, in 2001.

He is currently a Distinguished Chair Professor in Computational Intelligence, Department of Computer Science, University of Surrey, Guildford, U.K., where he heads the Nature Inspired Computing and Engineering Group. He was a Finland Distinguished Professor and a Changjiang Distinguished Visiting Professor appointed. He has (co)authored over 300 peer-reviewed journal and conference papers and been granted eight patents on evolutionary optimization.

He is the Editor-in-Chief of the IEEE Transactions on Cognitive and Developmental Systems and Complex & Intelligent Systems. He is also an Associate Editor or Editorial Board Member of the IEEE Transactions on Evolutionary Computation, IEEE Transactions on Cybernetics, IEEE Transactions on Nanobioscience, Evolutionary Computation, BioSystems, Soft Computing, and Natural Computing. He is an IEEE Distinguished Lecturer (2017-2019). He is the recipient of the 2014 and 2016 IEEE Computational Intelligence Magazine Outstanding Paper Award, the 2018 IEEE Transactions on Evolutionary Computation Outstanding Paper Award, and the Best Paper Award of the 2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. He is a Fellow of IEEE.