SCIENCE CHINA
Information Sciences

July 2019, Vol. 62 070206:1–070206:17
https://doi.org/10.1007/s11432-018-9754-6

• RESEARCH PAPER •
Special Focus on Pigeon-Inspired Optimization

A self-organizing multimodal multi-objective pigeon-inspired optimization algorithm

Yi HU[1,2], Jie WANG[1], Jing LIANG[1*], Kunjie YU[1], Hui SONG[3], Qianqian GUO[1], Caitong YUE[1] & Yanli WANG[1]

[1]*School of Electrical Engineering, Zhengzhou University, Zhengzhou* 450001*, China;*
[2]*Industrial Technology Research Institute, Zhengzhou University, Zhengzhou* 450001*, China;*
[3]*Computer Science and Information Technology, School of Science, RMIT University, Melbourne* VIC 3000*, Australia*

Received 15 August 2018/Accepted 30 November 2018/Published online 31 May 2019

Abstract Multi-objective optimization algorithms have recently attracted much attention as they can solve problems involving two or more conflicting objectives effectively and efficiently. However, most existing studies focus on improving the performance of the solutions in the objective spaces. This paper proposes a novel multimodal multi-objective pigeon-inspired optimization (MMOPIO) algorithm where some mechanisms are designed for the distribution of the solutions in the decision spaces. First, MMOPIO employs an improved pigeon-inspired optimization (PIO) based on consolidation parameters for simplifying the structure of the standard PIO. Second, the self-organizing map (SOM) is combined with the improved PIO for better control of the decision spaces, and thus, contributes to building a good neighborhood relation for the improved PIO. Finally, the elite learning strategy and the special crowding distance calculation mechanisms are used to prevent premature convergence and obtain solutions with uniform distribution, respectively. We evaluate the performance of the proposed MMOPIO in comparison to five state-of-the-art multi-objective optimization algorithms on some test instances, and demonstrate the superiority of MMOPIO in solving multimodal multi-objective optimization problems.

Keywords multi-objective, multimodal, decision space, pigeon-inspired optimization, PIO, self-organizing map, SOM

Citation Hu Y, Wang J, Liang J, et al. A self-organizing multimodal multi-objective pigeon-inspired optimization algorithm. Sci China Inf Sci, 2019, 62(7): 070206, https://doi.org/10.1007/s11432-018-9754-6

1 Introduction

As effective tools for solving optimization problems [1, 2], evolutionary algorithms (EAs) have recently drawn considerable research interest. However, for most real-world optimization problems, there may exist two or more conflicting optimization objectives. This type of problem is usually defined as the multi-objective problem (MOP). Because single optimal solutions obtained by single-objective optimization algorithms cannot meet the need for trade-offs among different objectives, an increasing number of multi-objective optimization algorithms are being developed for solving MOPs. Some studies on MOPs are discussed below.

In [3], the long-term conflict avoidance problem was formulated as an MOP, which can be divided into several sub-problems by using a cooperative co-evolution algorithm. Moreover, the multi-objective evolutionary algorithm based on decomposition (MOEA/D) was introduced to solve each sub-problem in this paper. Qu et al. [4] introduced an efficient procedure for solving large-scale portfolio optimization

* Corresponding author (email: liangjing@zzu.edu.cn)

problems. The MOEAs and preselection methods were used to reduce the complexity and improve the effectiveness. Most existing MOEAs show poor versatility on problems with different shapes of Pareto fronts. To address this shortcoming, Tian et al. [5] proposed an MOEA based on an enhanced inverted generational distance indicator.

In [6], a novel constraint-handling method based on fast sorting differential evolution (DE) was presented for tackling constraint MOPs. MOPs with interval parameters are ubiquitous and increase the intrinsic complexity. A novel EA that interacts with a decisionmaker (DM) was proposed in [7]. This algorithm can obtain the most preferred solution during the optimization process. Rong et al. [8] demonstrated a multidirectional prediction strategy for solving dynamic MOPs, which used multiple directions determined by multiple representative solutions coming from the previous environments to predict the new locations of Pareto-optimal sets (PSs) in the decision space.

Zhang et al. [9] presented a competitive mechanism-based multi-objective particle swarm optimizer. A competition mechanism-based learning strategy was designed to guide the search of particle swarm optimization (PSO) to enhance its robustness in handling MOPs.

MOPs with more than three objectives are usually called many-objective optimization problems (MaOPs). In [10], a novel many-objective EA using a one-by-one selection strategy was proposed to help balance the convergence and diversity of MaOPs in the high-dimensional objective space. Moreover, a reference point-based evolutionary algorithm was demonstrated by Liu et al. [11] to help maintain an extensive and uniform distribution among MaOP solutions. A set-based GA [12] was used to solve MaOPs with interval parameters, which introduced set-based Pareto dominance relations and an evolutionary scheme to improve the performance of the proposed algorithm.

A space comprising of all possible solutions to an MOP is called a decision space. Moreover, the space consisting of all possible values of the MOPs objective functions is known as the objective space. The best trade-off solution set in the decision space is called the PS and the corresponding values in the objective space are known as the Pareto front (PF). There exist multiple PSs in some practical problems [13]. Problems that have more than one solution corresponding to the same point in the objective space are defined as multimodal multi-objective optimization problems (MMOPs) [14]. Finding all PSs in one decision space can help the DM to obtain multiple solutions and find more robust solutions. Multimodal multi-objective optimization can also improve the diversity of the population and reveal the potential properties of the problems. Aiming at multimodal single-objective optimization, many approaches have been proposed, such as the niching methods [15–17]. However, few studies have been conducted on multimodal multi-objective optimization. To find methods with good performance in solving MMOPs, we try to introduce manipulations on the solution distribution into the multi-objective optimization algorithm.

The method of improving the point distribution in the decision space and objective space is one of the most efficient ways for solving MMOPs. The Omni-Optimizer proposed by Deb et al. [18] introduced the method of calculating crowding distance for the non-dominated sorting operation. A method known as decision-space-based niching NSGAII (DN-NSGA-II) was proposed by Liang et al. [14] for MMOPs. In DN-NSGA-II, the non-dominated solutions and less-crowded points are obtained from non-dominated sorting and the crowding distance calculation, respectively, to locate more PSs in the decision space. Yue et al. [19] presented a new multi-objective particle swarm optimization using ring topology and special crowding distance (MO-Ring-PSO-SCD) algorithm. The special crowding distance (SCD) calculation method was adopted in this algorithm to improve the point distribution in both decision and objective spaces. The proposed ring topology technique in the proposed MO-Ring-PSO-SCD can help generate stable niches. A self-organizing multi-objective particle swarm optimization algorithm for solving multimodal multi-objective problems (SMPSO-MM) in [20] combined SOM with PSO for MMOPs. SMPSO-MM can map the individuals in the population to a latent space for building a neighboring relation to improve the distribution features in the decision space and objective space.

Since the concept of bio-inspired computing was derived from the simulation of complex ecosystems in the natural world, the related bio-inspired algorithms are provided as a sound scientific basis for solving practical problems. Swarm intelligence algorithms, such as PSO [21–23] and ant colony algorithm

(ACO) [24], have been widely adopted for many real-world problems. A novel intelligent computation method based on bionics, known as pigeon-inspired optimization (PIO), was proposed by Duan et al. [25] in 2014. PIO has solved some difficult practical problems, such as biological object recognition [26], detection of protein complexes [27] and brushless direct current motor parameter design [28]. However, MOPs, let alone MMOPs, have seldom been involved with PIO.

There exist various EAs for MOPs and multimodal single-objective optimization problems. However, to the best of our knowledge, there are only a few methods for MMOPs. Moreover, this is the first time that PIO has been used for tacking MMOPs. Therefore, seeking the most preferred solutions of the DMs for an MMOP deserves a careful study.

This paper proposes a novel multimodal multi-objective pigeon-inspired optimization (MMOPIO) for MMOPs by employing the framework of SOM, which clusters similar solutions in the current population at each generation into the same neighborhood and helps maintain more PSs. The significant characteristics of the proposed MMOPIO are listed as below.

(1) A consolidation strategy is adopted to simplify the structure of the basic PIO and to perform a smooth transition from the map-and-compass operator to the landmark operator of PIO.

(2) At each generation, SOM is used to pick up neighboring relation information and help find the distribution of solutions in the current population.

(3) The elite learning strategy and the SCD are employed to achieve more PSs and improve the diversity of the solutions.

Following are the main contributions of our study: (1) A self-organizing MMOPIO algorithm is proposed, which is effective for maintaining better PSs because their local topological properties can be well-preserved. A consolidation parameter is adopted to simplify the structure and perform a smooth transition between the two operators of PIO. The elite learning strategy and the SCD are used in environmental selection for improving the diversity of the obtained PSs. (2) The effectiveness of the proposed MMOPIO is verified by comparing the performance of MMOPIO with the other four state-of-the-art multimodal multi-objective optimization algorithms and one novel multi-objective optimization algorithm on MMOP test functions. (3) The parameter analyses of the consolidation parameter and the population sizes of different comparison algorithms are provided in detail.

The remainder of this paper is organized as follows. Section 2 reviews the concept of MMOPs, and then, introduces the principles of the basic PIO and its deformation algorithm based on the consolidation parameter. Section 3 discusses the design of the proposed MMOPIO in detail. Section 4 presents the experimental studies and the results of the correlation analysis. Finally, Section 5 presents the conclusion of this paper.

2 Related work

2.1 Multimodal multi-objective problems

There always exist some different objectives in optimization problems. In MOPs, there are two or more conflicting objectives to be optimized, which can be defined as

$$\text{Min } F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})), \tag{1}$$

where $\boldsymbol{x}_i = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ expresses the decision vector (a vector in the n-dimensional decision space); $f(\boldsymbol{x}_i)$ is the corresponding objective value of $\boldsymbol{x}_i$ and is the objective vector (a vector in the m-dimensional objective space); m are the number of objectives to be optimized in MOPs. $\boldsymbol{x}$ and $\boldsymbol{y}$ represent two decision vectors. We can state that $\boldsymbol{y}$ is dominated by $\boldsymbol{x}$ ($\boldsymbol{x} \prec \boldsymbol{y}$) if the relation between these two vectors is as follows:

$$(\forall i \in \{1, 2, \ldots, m\} : f_i(\boldsymbol{x}) \leqslant f_i(y)) \wedge (\exists j \in \{1, 2, \ldots, m\} : f_j(\boldsymbol{x}) < f_j(\boldsymbol{y})). \tag{2}$$

In MOPs, we compare all obtained solutions according to the Pareto dominance relation expressed by (2). If a solution $\boldsymbol{x}$ is not dominated by any other solution in the decision space, we term this decision

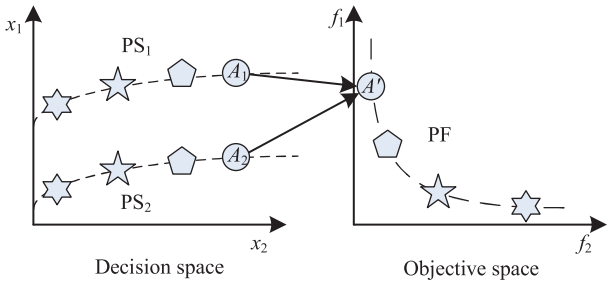

Figure 1 (Color online) Example of multimodal multi-objective problem.

vector $\boldsymbol{x}$ as a non-dominated solution, i.e., the PS is formed by a set of all non-dominated solutions in the MOPs decision space. The objective values $f(\boldsymbol{x}_i)$ obtained according to these non-dominated solutions comprise the corresponding PF. However, sometimes, the PF obtained in the MOPs may correspond to more than one PS. In this case, these MOPs are belong to MMOPs. As shown in Figure 1, the PF in the right coordinate system (objective space) corresponds to two different PSs (PS1 and PS2) in the left coordinate system (decision space), and the shape points (A_1 in the PS1 and A_2 in the PS2) in the left correspond to the same shape dot (A') in the right system simultaneously. With the conventional multi-objective optimization algorithms, even though we can find one of the PSs or part of the PSs and their corresponding PF accurately, some non-dominated solutions will be missing. Therefore, it is necessary to design more effective strategies to seek out all existing PSs and maintain them in the search process.

2.2 Framework of PIO

The PIO, proposed by Duan et al. [25] is a novel population-based optimization algorithm derived from the natural behavior of pigeons, which always find their destination accurately. The basic PIO mainly consists of two operators: the map-and-compass operator and the landmark operator. The details of these two processes are described below.

In the first stage, the map-and-compass operator helps guide the flight directions of virtual pigeons. This process can be described as follows:

$$\boldsymbol{v}_i(t) = \boldsymbol{v}_i(t-1) \cdot \mathrm{e}^{-R\times t} + \mathrm{rand} \cdot (\boldsymbol{x}_{g\mathrm{best}} - \boldsymbol{x}_i(t-1)), \quad i = 1, 2, \ldots, N, \tag{3}$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i(t-1) + \boldsymbol{v}_i(t), \tag{4}$$

where $R \in [0, 1]$ represents the map-and-compass operator parameter; t is the current iteration number; $\boldsymbol{x}_{g\mathrm{best}}$ denotes the best global position; N is the population size of the virtual pigeons; and $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$ indicate the positions and velocities, respectively, of the pigeons.

In the second stage, all pigeons obtained according to the first stage are sorted by calculating their corresponding fitness values. Moreover, only half of these pigeons are reserved for the next operation. This reservation mechanism is carried out in every generation. The realization of the landmark operator can be expressed as

$$\boldsymbol{x}_{\mathrm{center}}(t-1) = \frac{\sum_{i=1}^{N(t-1)} \boldsymbol{x}_i(t-1) \cdot F(\boldsymbol{x}_i(t-1))}{N(t-1) \cdot \sum_{i=1}^{N(t-1)} F(\boldsymbol{x}_i(t-1))}, \tag{5}$$

$$N(t) = \frac{N(t-1)}{2}, \tag{6}$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i(t-1) + \mathrm{rand} \cdot (\boldsymbol{x}_{\mathrm{center}}(t-1) - \boldsymbol{x}_i(t-1)), \tag{7}$$

where

$$F(\boldsymbol{x}_i(t-1)) = \begin{cases} \dfrac{1}{\mathrm{fitness}(\boldsymbol{x}_i(t-1)) + \varepsilon}, & \text{for minimization problem,} \\ \mathrm{fitness}(\boldsymbol{x}_i(t-1)), & \text{for maximization problem,} \end{cases} \tag{8}$$

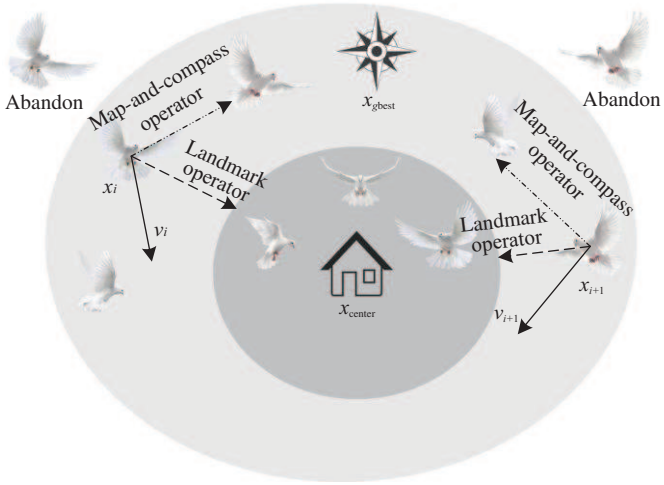

Figure 2 (Color online) Illustration of improved PIO.

$\boldsymbol{x}_{\text{center}}$ is defined as the supposed destination of the pigeons and $F(\boldsymbol{x}_i)$ presents the evaluation criterion of the reservation mechanism based on fitness values. For a maximization problem, half of the pigeons with larger fitness values will be reserved, and vice versa.

Inspired by the multi-objective pigeon-inspired optimization (MPIO) proposed by Qiu et al. [28], for simplicity, we combine the map-and-compass operator with the landmark operator by setting a consolidation parameter λ. The new implementation can be denoted as

$$\begin{aligned}\boldsymbol{v}_i(t) =& \boldsymbol{v}_i(t-1)\cdot \mathrm{e}^{-R\times t} + \mathrm{rand}_1\cdot\lambda\cdot(1-\mathrm{lg}_T^t)\cdot(\boldsymbol{x}_{g\text{best}}-\boldsymbol{x}_i(t-1))\\ &+\mathrm{rand}_2\cdot\lambda\cdot\mathrm{lg}_T^t\cdot(\boldsymbol{x}_{\text{center}}(t-1)-\boldsymbol{x}_i(t-1)),\end{aligned} \tag{9}$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i(t-1)+\boldsymbol{v}_i(t), \tag{10}$$

where T is the maximum iteration. As shown in Figure 2, the pigeons put more emphasis on coming close to $\boldsymbol{x}_{\text{center}}$ rather than $\boldsymbol{x}_{g\text{best}}$ as the number of iterations increase.

The map-and-compass operator is combined with the landmark operator by the parameter λ. Since different λ values affect the performance of PIO to a large extent, an experimental study is conducted to explore the impact of varying the consolidation parameter λ, as described in Subsection 4.3. The value of λ can balance the performance of the diversity and convergence of PIO, as proven in [28]. When the value of λ is small, the algorithm emphasizes the map-and-compass operator. Then, the diversity of the population is increased. Moreover, the convergence effect of the landmark operator increases along with the consolidation parameter λ.

2.3 Self-organizing map

SOM is an unsupervised clustering algorithm proposed by Kohonen [29]. Because of the merits of high-dimensional data visualization and simplicity [30,31], SOM has been widely used to deal with data mining problems [32]. Recently, an increasing number of researchers have been combining SOM with optimization algorithms to solve optimization problems [33–35]. Our proposed MMOPIO integrates SOM with PIO to solve MMOPs.

SOM can map the input data (high-dimensional input space) to output data (low-dimensional representation space). The structure of a two-dimensional SOM is shown in Figure 3, where SOM comprises the input layer and the latent layer. There exists a certain number of neurons in each layer. We hypothesize that the input data are n-dimensional and the latent space includes $M = M_1 \times M_2$ neurons. $\boldsymbol{z}^u = (\boldsymbol{z}_1^u, \boldsymbol{z}_2^u)$ expresses the position of a neuron, and $\boldsymbol{w}^u = (\boldsymbol{w}_1^u, \boldsymbol{w}_2^u, \ldots, \boldsymbol{w}_n^u)$ represents the weight vector. The information of both the position and the weight vector is contained in each neuron ($u \in (1, 2, \ldots, M)$). At first, the training data are selected randomly from the input data, and then, the best neuron (u') is

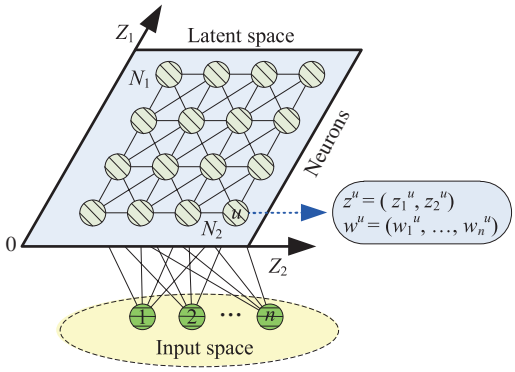

Figure 3 (Color online) Illustration of a two-dimensional SOM.

found by (14) according to the calculation of Euclidean distance. Afterward, the weight vector of u' and its neighboring neurons (SOM model) are updated iteratively according to (15) and (16). The testing data are mapped onto the obtained SOM model. Then, the best neuron and its neighboring neurons are selected, and finally, recorded.

3 Details of MMOPIO

3.1 Framework of MMOPIO

As a population-based algorithm, PIO has the ability and the advantage of solving various real-world problems [25–28]. Inspired by the self-organizing multi-objective evolutionary algorithm proposed by Zhang et al. [33], we propose an MMOPIO algorithm with SOM. In this subsection, the MMOPIO procedure is presented, followed by a detailed description of its underlying mechanism.

The framework of the proposed MMOPIO is shown in Algorithm 1, where **nbest** represents the neighboring leader pigeon that is found by SOM. We use the training data ($\boldsymbol{D}$) to update the SOM model. The best pigeon is found when we import the $\boldsymbol{P}$ (test data) into the updated SOM model. The best pigeons are retained in archive ($\boldsymbol{A}$). By using $\boldsymbol{A}$, the position of each pigeon can be improved generation by generation in a stable fashion.

Algorithm 1 Framework of MMOPIO

Require: N (pigeon size), $\boldsymbol{x}$ (current positions of pigeons),$\boldsymbol{v}$ (current velocities of pigeons), δ^e (standard deviation) R (the map-and-compass operator parameter), λ (the consolidation parameter), $\boldsymbol{A}$ (archive), $\boldsymbol{D}$ (training data) $\boldsymbol{P}$ (test data);

($\boldsymbol{x}$, $\boldsymbol{v}$, $\boldsymbol{A}$, $\boldsymbol{D}$) ← Initialize(N);
while termination criterion not fulfilled **do**
  nbest ← Self-OrganizingBasedLearning($\boldsymbol{x}, \boldsymbol{v}$);
  $\boldsymbol{x}$ ← Inversemap(**nbest**) , find the pigeon particle corresponding to **nbest** according to (14);
  Sorted_$\boldsymbol{x}$ ← non-dominated-SCD-sort($\boldsymbol{x}$);
  /*Select the elite pigeons*/
  for each $\boldsymbol{x}_i \in$ Sorted_$\boldsymbol{x}$ **do**
    ($\boldsymbol{x}_i, \boldsymbol{v}_i$) ← Update positions and velocities of pigeons by (9) and (10);
    $\boldsymbol{x}_i$ ← Generate offspring by the elite learning strategy according to (11);
    /*Update $\boldsymbol{A}$ and $\boldsymbol{D}$*/
    Temp_$\boldsymbol{A}$ ← $\boldsymbol{A} \cup \{\boldsymbol{x}_i\}$, i is the current iteration;
    Sorted_$\boldsymbol{A}$ ← non-dominated-SCD-sort(Temp_$\boldsymbol{A}$);
    $\boldsymbol{D}$ ← Sorted_$\boldsymbol{A} \setminus \boldsymbol{A}$;
  end for
end while

Ensure: $\boldsymbol{x}$ (final positions of pigeons).

The general procedure for implementing MMOPIO is as follows. First, the current positions ($\boldsymbol{x}$) and velocities ($\boldsymbol{v}$) of the pigeons, the training data ($\boldsymbol{D}$), and the archive ($\boldsymbol{A}$) are initialized. Then, $\boldsymbol{x}$ and $\boldsymbol{v}$

are imported into the Self-OrganizingBasedLearning operator to generate the neighboring leader pigeon (**nbest**), which is chosen from the neighborhood that guides the pigeons flying to promising positions. Moreover, we can find the updated $\boldsymbol{x}$ corresponding to **nbest** according to (14). After inverse mapping, the non-dominated-SCD-sort method [19] is used to sort $\boldsymbol{x}$ according to the dominance relation. Then, the steps of selecting the elite pigeons are executed in a loop. $\boldsymbol{x}$ and $\boldsymbol{v}$ are updated according to the improved (9) and (10), respectively. Then the candidate offspring are reproduced by the elite learning strategy, which can be expressed as

$$\boldsymbol{x}_i = \text{nbest}_i + \text{Gauss}(0, (\delta^e)^2) \cdot (\boldsymbol{b}_i - \boldsymbol{a}_i), \quad \text{if rand} < \delta^e, \tag{11}$$

where δ^e is the standard deviation and $\boldsymbol{a}$ and $\boldsymbol{b}$ are the minimum and maximum positions of the pigeons in MMOPIO. The Gaussian distribution mechanism (Gauss(0, $(\delta^e)^2$)) is used in the elite learning strategy to help improve the diversification in the early global search stage and accelerate the convergence in the subsequent local search stage. The mutation operation and boundary treatment are also implemented to avoid falling into local optima [20].

After selecting the elite pigeons, $\boldsymbol{A}$ is updated by removing the dominated solutions from $\boldsymbol{A} \cup \{\boldsymbol{x}_i\}$. Then, the new solutions generated by the non-dominated-SCD-sort algorithm and the current population are combined to accomplish the updating of $\boldsymbol{D}$. The above steps are repeated until the termination conditions are met.

3.2 Learning mechanism based on self-organizing strategy

Because SOM can represent data by mapping them from a high-dimensional space to a low-dimensional space and preserve the local topological properties at the same time [29], some researchers have combined SOM with an evolutionary multi-objective optimization algorithm to help generate better solutions. In [33], SOM is used to extract neighboring relation information, which can help guide recombination within the neighborhood to generate new solutions. In [20], SOM helps gather good and similar solutions together in MOPSO to build a neighboring relation in the decision space. The proposed SMPSO-MM improves the PS distribution on MMOPs. In this study, we use SOM to establish the neighboring relation of the current solutions and help improve the quality of the new solutions of PIO on MMOPs.

The neighboring leader pigeons play an essential role in the local search on MMOPs. SOM has the advantage of preserving the local topological properties of PSs well. In this study, SOM is used to cluster similar solutions into the same neighborhood and help find the distribution of the current population and $\boldsymbol{A}$. According to non-dominated-SCD-sort, the best solutions and their neighboring information stored in $\boldsymbol{A}$ are updated. The pigeons with larger SCD are preferred as the new offspring. The self-organizing-based learning mechanism is described by Algorithm 2.

Algorithm 2 Self-OrganizingBasedLearning($\boldsymbol{x}$, $\boldsymbol{v}$)

Require: $\boldsymbol{x}$ (current positions of pigeons), $\boldsymbol{v}$ (current velocities of pigeons), $\boldsymbol{w}^u$ (neuron weight vectors) η_0 (learning rate), σ_0 (learning radius), R (the map-and-compass operator parameter) λ (the consolidation parameter);

/*Update SOM model*/

$\boldsymbol{w}^u \leftarrow$ Initialize($\boldsymbol{x}$);

(η, σ) $\leftarrow$ Calculate values of learning rate and radius by (12) and (13);

Find the best neuron in $\boldsymbol{D}$ by (14);

Find the neighboring neurons of u' by (15);

Update the neuron weight vectors $\boldsymbol{w}^u$ according to (16);

/*Find the **nbest** pigeons*/

Map the pigeons of $\boldsymbol{P}$ to the updated SOM latent space and record the best neuron to the archive $\boldsymbol{AA}$. The best neuron is found according to (14);

Map the pigeons of $\boldsymbol{A}$ to the updated SOM latent space and record **nbest** to the archive $\boldsymbol{BA}$, **nbest**=$\boldsymbol{BA}$ $\{I_m^u\}$, I_m^u indicates the index of the m-th nearest neuron to neuron u in the latent space;

Ensure: **nbest** (the neighborhood pigeons).

There are two important parts in the mechanism: SOM model updating and the selection of **nbest** pigeons. In the updating part of the SOM model, the neuron weight vectors ($\boldsymbol{w}^u$) are initialized first.

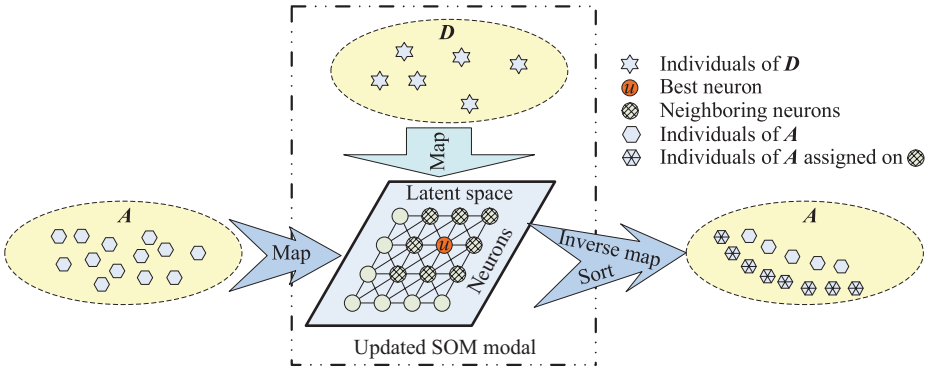

Figure 4 (Color online) The process of finding **nbest**.

Then, the learning rate (η_0) and the learning radius (σ_0) are updated [36] according to the current iteration (t) and the maximum iteration (T):

$$\eta{=}\eta_0 \cdot \left(1 - \frac{t}{T}\right), \tag{12}$$

$$\sigma{=}\sigma_0 \cdot \left(1 - \frac{t}{T}\right) \Big/ t. \tag{13}$$

Afterward, the best neuron in $\boldsymbol{D}$ and its neighboring neurons are sought by

$$u' = \arg \min_{1 \leqslant u \leqslant M} \|\boldsymbol{x} - \boldsymbol{w}^u\|_2, \tag{14}$$

$$U = \left\{1 \leqslant u \leqslant M \wedge \left\|\boldsymbol{z}^u - \boldsymbol{z}^{u'}\right\|_2 < \sigma\right\}. \tag{15}$$

The new values of $\boldsymbol{w}^u$ are finally obtained by

$$\boldsymbol{w}^u = \boldsymbol{w}^u + \eta \cdot \exp\left(\left\|\boldsymbol{z}^u - \boldsymbol{z}^{u'}\right\|_2\right)(\boldsymbol{x} - \boldsymbol{w}^u). \tag{16}$$

In the **nbest** pigeon selection part, once the SOM model is updated, the **nbest** pigeons can be selected and recorded by the following operations. The pigeons in the test data ($\boldsymbol{P}$) are mapped to the new SOM latent space and u' is first logged to $\boldsymbol{AA}$. Then, the pigeons in $\boldsymbol{A}$ are mapped to the updated SOM latent space and the best neuron is recorded to $\boldsymbol{BA}$. Afterward, the non-dominated-SCD-sort operation proposed by Yue et al. [19] is adopted for $\boldsymbol{BA}$ and the **nbest** pigeons are selected via the sorted $\boldsymbol{BA}$.

The process of finding the nbest pigeons is illustrated in Figure 4. The SOM model is updated by the newly generated non-dominated solutions that are sorted in $\boldsymbol{D}$. The updated SOM model can reflect the dynamic changes in the pigeons in the decision space and improve the diversification of the obtained solutions. According to the built neighboring relation, the best neuron will be found and recorded when the pigeons in the test data ($\boldsymbol{P}$) are mapped onto the updated SOM latent space. The pigeon particle in the input space corresponding to **nbest** is calculated by (14) and the solutions of the neighboring neurons of **nbest** are calculated by (15).

The **nbest** pigeons consist of the solutions of the neighboring neurons of the best neuron clustered by SOM. By this mechanism, the neuron weight vectors $\boldsymbol{w}^u$ are updated when the solutions are sorted by non-dominated-SCD-sort in each generation. In the decision space, the **nbest** pigeons are close to other non-dominated solutions assigned to the neighboring pigeons. Different leader pigeons can improve the diversity of the algorithm and maintain more multimodal solutions.

4 Experimental studies

4.1 Test functions and performance indicators

Eleven multimodal multi-objective test functions are applied in the experimental studies. These test functions contain MMF1, MMF2 (designed by Liang et al. [14]), six test instances (MMF3–MMF8) proposed

by Yue et al. [19], two complex benchmarks named SYM-PART-simple and SYM-PART-rotated [37], and Omni-test function [18] with n=3.

Pareto set proximity (PSP) is a new indicator proposed by Yue et al. [19]. A PSP value can reflect both the convergence of the obtained PS and the degree of similarity between the obtained PS and the true PS. PSP can be defined as

$$\mathrm{PSP} = \frac{\mathrm{CR}}{\mathrm{IGDX}}, \tag{17}$$

where CR is the cover rate of the solutions modified from the maximum spread (MS) [38], and IGDX [39] presents the inverted generational distance in the decision space. CR can be expressed by

$$\mathrm{CR} = \left(\prod_{i=1}^{n} \delta_i\right)^{1/2n}, \tag{18}$$

where

$$\delta_i = \begin{cases} 1, & \boldsymbol{Q}_i^{\max} = \boldsymbol{Q}_i^{\min}, \\ 0, & \boldsymbol{q}_i^{\min} \geqslant \boldsymbol{Q}_i^{\max} \parallel \boldsymbol{q}_i^{\max} \leqslant \boldsymbol{Q}_i^{\min}, \\ \left(\dfrac{\min(\boldsymbol{q}_i^{\max}, \boldsymbol{Q}_i^{\max}) - \max(\boldsymbol{q}_i^{\min}, \boldsymbol{Q}_i^{\min})}{\boldsymbol{Q}_i^{\max} - \boldsymbol{Q}_i^{\min}}\right), & \text{otherwise.} \end{cases} \tag{19}$$

The dimensionality of the decision space is denoted by n, where i= (1, 2, ..., n), $\boldsymbol{q}_i^{\min}$, and $\boldsymbol{Q}_i^{\min}$ are the minimum of the achieved PS and the true PS in i-dimensional space, respectively, and $\boldsymbol{q}_i^{\max}$ and $\boldsymbol{Q}_i^{\max}$ are the maximum of the obtained PS and the true PS in i-dimensional space, respectively. IGDX can be written as

$$\mathrm{IGDX}(\boldsymbol{O}, \boldsymbol{P}^*) = \frac{\sum_{q \in \boldsymbol{P}^*} d(\boldsymbol{q}, \boldsymbol{O})}{|\boldsymbol{P}^*|}, \tag{20}$$

where $\boldsymbol{P}^*$ indicates a set whose solutions are distributed uniformly in the true PS. All achieved solutions in the decision space form a set denoted as $\boldsymbol{O}$. $\boldsymbol{d}(\boldsymbol{q}, \boldsymbol{O})$ reflects the Euclidean distance between $\boldsymbol{q}$ and the solutions in $\boldsymbol{O}$.

Since PSP is a metric that can be used to measure the performances of algorithms in the decision space, another indicator named IGDf is employed to demonstrate the optimization effects of the proposed MMOPIO in the objective space. The calculation formula of IGDf is similar to (20), where $\boldsymbol{P}^*$ can be expressed by a set formed by the points in the true PF and the solutions in $\boldsymbol{O}$ can be replaced by points distributed in the obtained PF in the objective space. A larger PSP and smaller IGDf values mean that the proposed MMOPIO works well in both the decision and objective spaces.

4.2 Experimental setup

Since solving MMOPs is still in the early stages of development, there are no more training and testing data. Only very few test functions can be found in the current literature. Eleven representative benchmark test functions are chosen to measure the effectiveness of the proposed MMOPIO and the other five comparison algorithms. Most of these test functions were proposed in the past three years. The dimension of the decision space of MMF1–MMF8, SYM-PART-simple, and SYM-PART-rotated is 2. The dimension of the decision space of Omni-test is set to 3 for a more accessible visualization. The features of the eleven test functions are described in detail in [19].

The proposed MMOPIO is compared with the other five algorithms, including Omni-Optimizer [18], MPIO [28], DN-NSGA-II [14], MO-Ring-PSO-SCD [19], and SMPSO-MM [20]. Since the population size setting affects the performance of an EA in most cases, an experimental study is conducted to explore the influence of varying the population size. Details are presented in Subsection 4.4. The results show that most of these six comparison algorithms perform better when the population size of each algorithm is set

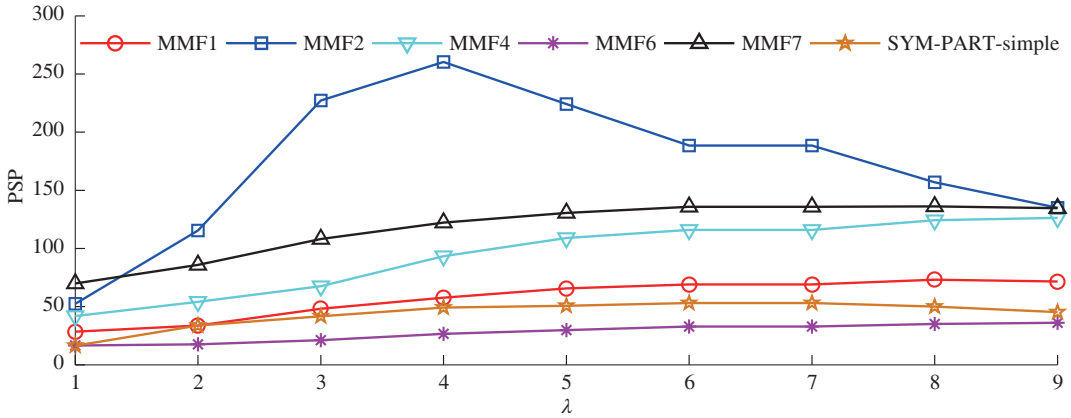

Figure 5 (Color online) PSP values obtained by MMOPIO with different λ values.

to 700. For a fair comparison, we set the population size of each comparison algorithm to 700 and the maximal number of function evaluations of each comparison algorithm to 60000 according to [20]. The other parameter settings of these five comparison methods are the same as the parameters recommended in the original papers.

All experiments on the eleven benchmark test instances are conducted 30 times independently. The experiments are implemented with MATLAB 2014a on the 64-bit Microsoft Windows 7 operating system running on a PC with Intel(R) Core(TM) i5-6500 3.20 GHz CPU and 8 GB RAM.

4.3 Comparison of different algorithms

The consolidation parameter performs a smooth transition between the two stages of PIO [28]. Experiments with different λ values are conducted to study their effects on the performance of the proposed MMOPIO. The results are as follows.

Figure 5 indicates the PSP values obtained by our proposed method with different λ values (from 1 to 9) averaging over 30 runs. The test functions are MMF1, MMF2, MMF4, MMF6, MMF7, and SYM-PART-simple. MMF1, MMF2, MMF4, and MMF7 are representative MMOPs that do not overlap in every dimension. MMF6 and SYM-PART-simple have four and nine PSs, respectively, which overlap in every dimension. It can be observed that the performance of MMOPIO is relatively sensitive to the value of λ on MMOPs, especially in the case of functions without overlapping. For most of these test functions, the proposed MMOPIO achieves the best performance when the λ value is set to 7. However, MMOPIO maintains a relatively robust performance on MMF6 and SYM-PART-simple with different λ values. The sensitivity analysis of the parameter λ suggests a size of 7 for the consolidation parameter in the proposed MMOPIO on MMOPs.

To study the performance of the proposed MMOPIO, we compare MMOPIO with five other algorithms, most of which are multimodal multi-objective optimization methods with good performance. The exception is MPIO, which is only a novel multi-objective optimization algorithm without any application for solving multimodal problems.

The PSs and PFs of MM4 obtained by six different algorithms are shown in Figures 6 and 7, respectively. It is obvious that the PSs obtained by MO-Ring-PSO-SCD, SMPSO-MM, and MMOPIO are better than those obtained by the other methods. Moreover, the results of Omni-Optimizer and DN-NSGA-II are worse than those of the three methods mentioned above but better than those of MPIO.

For MO-Ring-PSO-SCD, SMPSO-MM, and MMOPIO, the solutions are distributed more evenly in the decision space because of the application of the non-dominated-SCD-sort method [19]. The SCD is calculated after the non-dominated sorting, which involves calculations in both the decision space and objective space to improve the distribution of the solutions in PSs and PFs simultaneously. On the other hand, MO-Ring-PSO-SCD adds a ring topology mechanism, whereas SMPSO-MM and MMOPIO select a self-organizing method to generate offspring with stable niches in the decision space. Otherwise, the

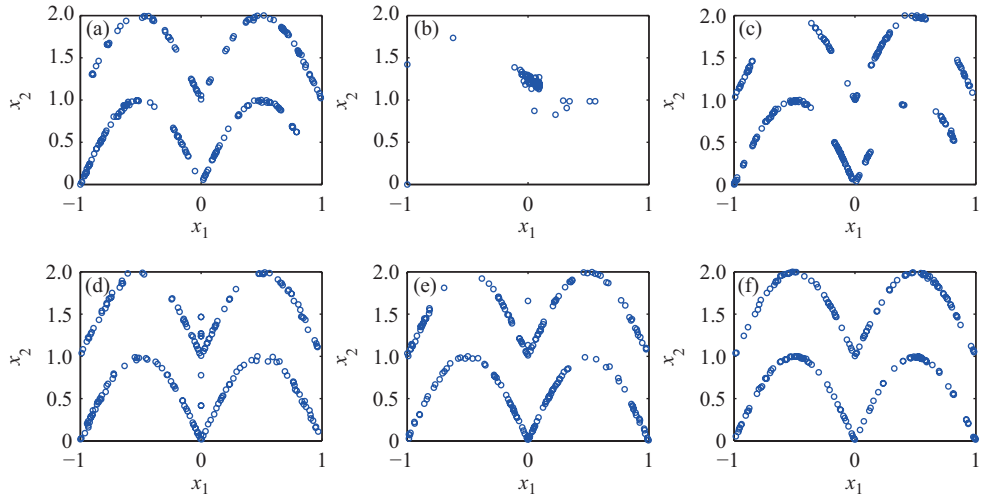

Figure 6 (Color online) PSs of MM4 obtained by different algorithms. (a) PS of Omni-Optimizer; (b) PS of MPIO; (c) PS of DN-NSGA-II; (d) PS of MO-Ring-PSO-SCD; (e) PS of SMPSO-MM; (f) PS of MMOPIO.

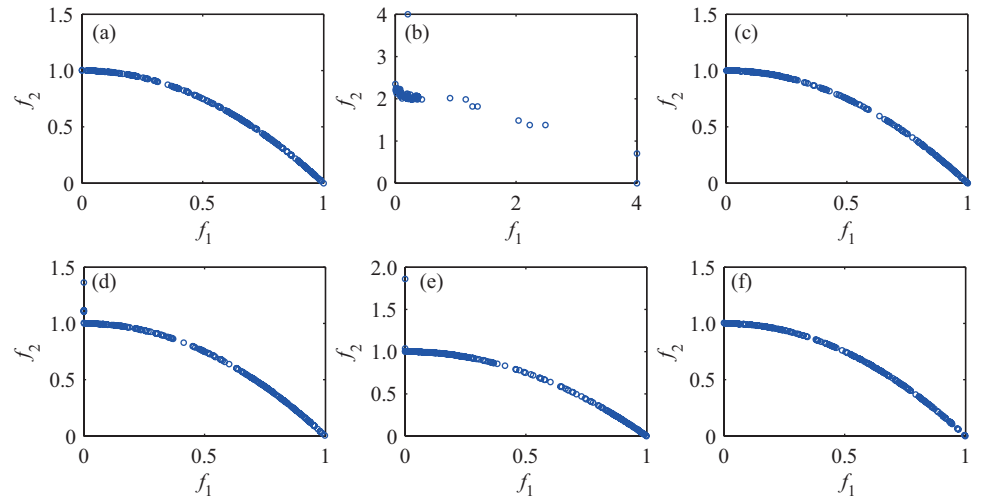

Figure 7 (Color online) PFs of MM4 obtained by different algorithms. (a) PF of Omni-Optimizer; (b) PF of MPIO; (c) PF of DN-NSGA-II; (d) PF of MO-Ring-PSO-SCD; (e) PF of SMPSO-MM; (f) PF of MMOPIO.

remaining algorithms (Omni-Optimizer and MPIO) only calculate a normal crowding distance in the objective space or do not have more effective niching methods, so they cannot obtain better PSs even if the corresponding PF is good.

As described by Yue et al. [19], the PSP indicator can reflect the performance of an algorithm dealing with MMOPs. The obtained solutions would have a better convergence capacity when the PSP value increases, meaning that the algorithm with a larger PSP value has a higher degree of similarity between the obtained PS and the true PS.

According to Figure 8, eleven multimodal multi-objective benchmarks problems are tested by Omni-Optimizer, MPIO, DN-NSGA-II, MO-Ring-PSO-SCD, SMPSO-MM, and MMOPIO. The results show that the mean PSP values obtained by MMOPIO are highest for most of these eleven test functions, especially for the three complex benchmarks (SYM-PART-simple, SYM-PART-rotated, and Omni-test), meaning that the solutions obtained by MMOPIO are closer to the true PSs and have much better convergence. SMPSO-MM ranks second, followed by MO-Ring-PSO-SCD. The performances of Omni-Optimizer and DN-NSGA-II are similar for most of these eleven test functions. The worst performance is exhibited by MPIO. When it comes to MMF5 and MMF6, SMPSO-MM has a much larger mean PSP value than any other comparision algorithms. Moreover, the mean PSP values of SMPSO-MM are

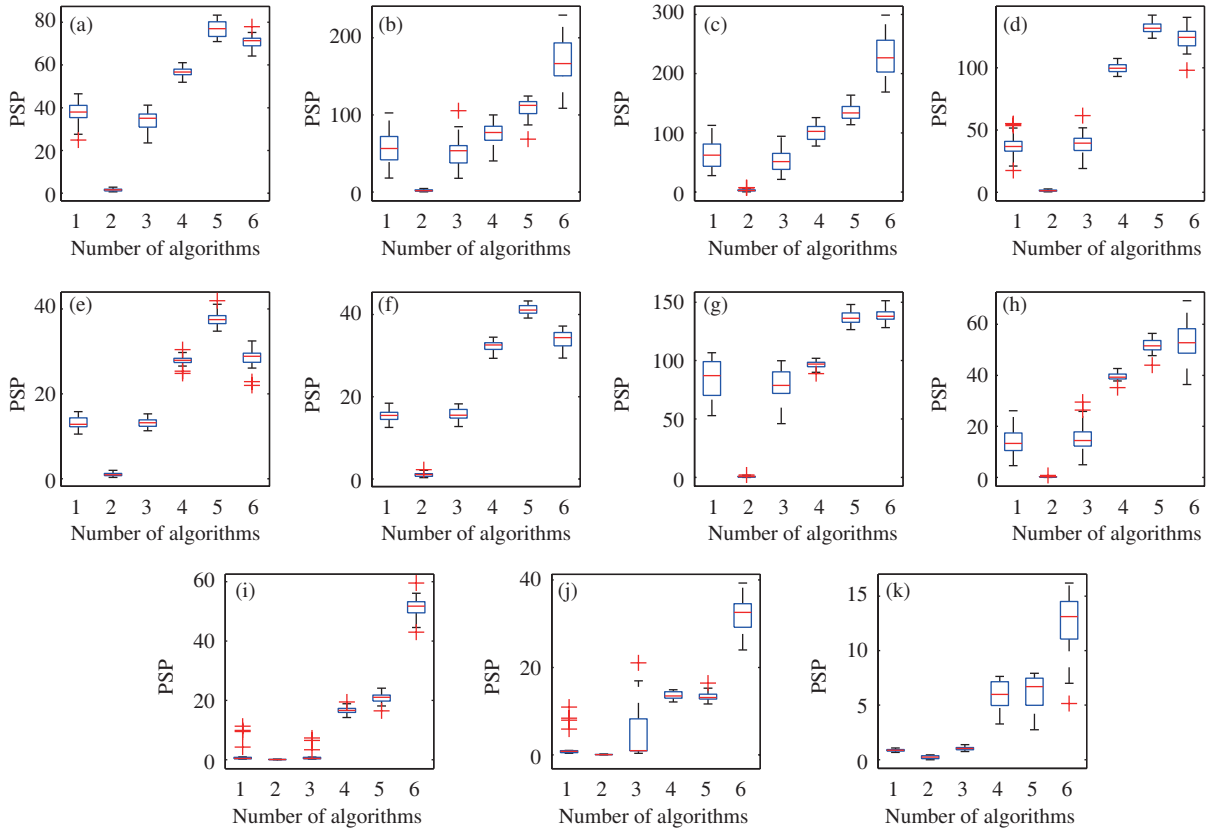

Figure 8 (Color online) PSP value box-plots of 11 test functions under six different algorithms. In each plot, the horizontal axis numbers indicate different algorithms: 1=Omni-Optimizer, 2=MPIO, 3=DN-NSGA-II, 4=MO-Ring-PSO-SCD, 5=SMPSO-MM, and 6=MMOPIO. (a) MMF1; (b) MMF2; (c) MMF3; (d) MMF4; (e) MMF5; (f) MMF6; (g) MMF7; (h) MMF8; (i) SYM-PART-simple; (j) SYM-PART-rotated; (k) Omni-test.

similar to those of MMOPIO on MMF1, MMF4, MMF7, and MMF8. Note that MMF1, MMF4, MMF5, MMF7, and MMF8 are functions without overlaps in every dimension, except MMF6, meaning that these functions are relatively simple MMOPs. As can be verified from the results, the performance of MMOPIO increases with the complexity of the test function.

The means and standard deviations of the PSP and IGDf indicator values achieved by the six comparison algorithms for each test function are shown in Tables 1 and 2, respectively. On each test function, 30 independent runs are conducted. The underlined PSP and IGDf values indicate excellent performance. '⊛', '⊖' and '◎' mean that the other algorithms perform better, worse, or similarly, respectively, as compared with the proposed MMOPIO according to Wilcoxon's rank sum test.

As shown in Table 1, the methods of Omni-Optimizer, MPIO, DN-NSGA-II, MO-Ring-PSO-SCD, SMPSO-MM, and MMOPIO achieve the 0, 0, 0, 0, 4, and 7 best mean PSP values respectively, indicating that the proposed MMOPIO performs better than the five compared state-of-the-art multi-objective algorithms in solving MMOPs in the decision space except for SMPSO-MM, which ranks first on MMF1, MMF4, MMF5, and MMF6.

The IGDf values denote the degree of proximity of the obtained PF to the true PF; smaller PSP values are desirable. As indicated in Table 2, MMOPIO obtains six best mean IGDf values and shows that the achieved PFs of these six different test instances by this method are much closer to the true PFs. Omni-Optimizer obtained the four best mean IGDf values, which indicate that this algorithm performed worse than MMOPIO in the objective space but much better than the other compared algorithms.

Tables 1 and 2 show that the proposed MMOPIO could obtain higher PSP values but relatively low IGDf values at the same time for solving MMOPs. The proposed MMOPIO is superior in both the PSP and IGDf values, indicating that it can achieve a better balance in both the decision and objective spaces.

Table 1 PSP values achieved by different algorithms

Mean (Std dev)	Omni-Optimizer	MPIO	DN-NSGA-II	MO-Ring-PSO-SCD	SMPSO-MM	MMOPIO
MMF1	36.93(5.51) ⊖	1.51(0.69) ⊖	33.61(5.07) ⊖	56.67(2.38) ⊖	76.63(3.21) ⊛	70.69(2.9)
MMF2	56.17(25.93) ⊖	1.89(1.02) ⊖	49.68(18.52) ⊖	74.99(13.72) ⊖	107.9(12.4) ⊖	170.35(32.95)
MMF3	64.36(22.41) ⊖	2.98(1.72) ⊖	52.06(19.85) ⊖	99.17(14.21) ⊖	136.43(12.83)⊖	225.01(33.17)
MMF4	38.22(8.78) ⊖	1.31(0.61) ⊖	37.43(8.64) ⊖	99.55(3.83) ⊖	133.03(4.8) ⊛	122.14(9.62)
MMF5	13.15(1.34) ⊖	0.97(0.37) ⊖	13.18(1.32) ⊖	28.15(1.21) ◎	37.53(1.58) ⊛	28.33(2.22)
MMF6	15.74(1.43) ⊖	1.11(0.54) ⊖	15.55(1.29) ⊖	32.09(1.21) ⊛	41.4(1.38) ⊛	33.97(2.05)
MMF7	82.64(16.29) ⊖	0.7(0.54) ⊖	77.74(14.29) ⊖	96.69(3.74) ⊖	135.99(4.65) ◎	137.54(5.93)
MMF8	14.16(5.19) ⊖	0.32(0.19) ⊖	15.86(5.4) ⊖	39.58(1.39) ⊖	51(3.06) ⊖	54.67(7.93)
SYM-PART-simple	1.85(5.01) ⊖	0.08(0.05) ⊖	0.81(1.34) ⊖	16.41(1.18) ⊖	20.51(1.74) ⊖	51.4(4.02)
SYM-PART-rotated	2.64(4.22) ⊖	0.07(0.04) ⊖	4.49(5.49) ⊖	13.76(0.91) ⊖	13.26(1.06) ⊖	32.04(3.31)
Omni-test	0.87(0.13) ⊖	0.25(0.15) ⊖	1.01(0.18) ⊖	5.9(1.26) ⊖	6.29(1.31) ⊖	12.41(2.74)
⊛/ ⊖ /◎	0/11/0	0/11/0	0/11/0	1/9/1	4/6/1	

Table 2 IDGf values achieved by different algorithms

Mean (Std dev)	Omni-Optimizer	MPIO	DN-NSGA-II	MO-Ring-PSO-SCD	SMPSO-MM	MMOPIO
MMF1	8.15E-04 ⊖	2.18E-01 ⊖	1.02E-03 ⊖	1.18E-03 ⊖	7.82E-04 ◎	7.64E-04
	(5.62E-05)	(1.16E-01)	(7.37E-05)	(6.40E-05)	(3.89E-05)	(3.99E-05)
MMF2	1.20E-03 ⊛	4.93E-01 ⊖	1.55E-03 ⊛	7.02E-03 ⊖	4.91E-03 ⊖	2.19E-03
	(1.11E-03)	(2.68E-01)	(6.74E-04)	(8.19E-04)	(4.77E-04)	(2.83E-04)
MMF3	8.21E-04 ⊛	4.54E-01 ⊖	1.74E-03 ⊛	5.32E-03 ⊖	3.94E-03 ⊖	1.84E-03
	(1.63E-04)	(3.21E-01)	(1.79E-03)	(5.86E-04)	(3.01E-04)	(1.78E-04)
MMF4	7.79E-04 ⊖	6.75E-01 ⊖	8.93E-04 ⊖	1.06E-03 ⊖	6.96E-04 ⊖	6.77E-04
	(6.52E-05)	(4.37E-01)	(6.87E-05)	(7.30E-05)	(3.49E-05)	(3.79E-05)
MMF5	7.70E-04 ⊖	1.16E-01 ⊖	9.29E-04 ⊖	1.14E-03 ⊖	7.62E-04 ⊖	7.25E-04
	(2.97E-05)	(4.36E-02)	(4.19E-05)	(4.10E-05)	(2.18E-05)	(2.88E-05)
MMF6	7.69E-04 ⊖	2.39E-01 ⊖	9.37E-04 ⊖	1.08E-03 ⊖	7.05E-04 ⊖	6.94E-04
	(3.14E-05)	(2.40E-01)	(5.86E-05)	(3.75E-05)	(2.50E-05)	(2.83E-05)
MMF7	8.55E-04 ⊖	1.59E+00 ⊖	1.14E-03 ⊖	1.06E-03 ⊖	6.71E-04 ◎	6.58E-04
	(3.21E-05)	(2.38E-01)	(6.32E-05)	(4.76E-05)	(2.11E-05)	(2.39E-05)
MMF8	8.37E-04 ⊖	5.20E-01 ⊖	1.03E-03 ⊖	1.64E-03 ⊖	1.20E-03 ⊖	7.79E-04
	(3.20E-05)	(4.50E-01)	(6.65E-05)	(8.41E-05)	(5.83E-05)	(4.49E-05)
SYM-PART-simple	3.45E-03 ◎	9.23E-01 ⊖	3.61E-03 ◎	1.34E-02 ⊖	1.07E-02 ⊖	3.49E-03
	(4.07E-04)	(3.45E-01)	(4.59E-04)	(1.52E-03)	(1.59E-03)	(4.76E-04)
SYM-PART-rotated	3.72E-03 ⊛	1.02E+00 ⊖	4.32E-03 ◎	1.64E-02 ⊖	1.65E-02 ⊖	4.78E-03
	(4.78E-04)	(3.63E-01)	(6.70E-04)	(2.17E-03)	(2.21E-03)	(8.12E-04)
Omni-test	9.98E-01 ⊖	2.89E+00 ⊖	9.97E-01 ⊖	9.76E-01 ⊛	9.77E-01 ⊛	9.96E-01
	(2.13E-04)	(6.13E-01)	(3.03E-04)	(3.03E-03)	(2.50E-03)	(3.58E-04)
⊛/ ⊖ /◎	3/7/1	0/11/0	2/7/2	1/10/0	1/8/2	

4.4 Effects of population sizes

Population size carries great importance for most of the optimization algorithms. Although a larger population size can improve the performance of an algorithm in most cases, the high computational cost should be considered. To study the effects of different population sizes on the performances of the six algorithms, experiments with different population sizes are conducted. Figure 9 shows the achieved PSP values of different algorithms with different population sizes. In each subfigure, the horizontal axis represents the respective population sizes of 100, 300, 500, 700, and 900 of the comparison methods. The vertical axis denotes the average PSP values of the experiments, which were repeated 30 times with the six algorithms.

Figures 9(a)–(f) present the PSP values on MMF1, MMF3, MMF7, MMF8, SYM-PART-simple, and Omni-test. These values are obtained by different algorithms with varying population size. These six test functions have multimodal multi-objective properties. The numbers of PSs of different test functions

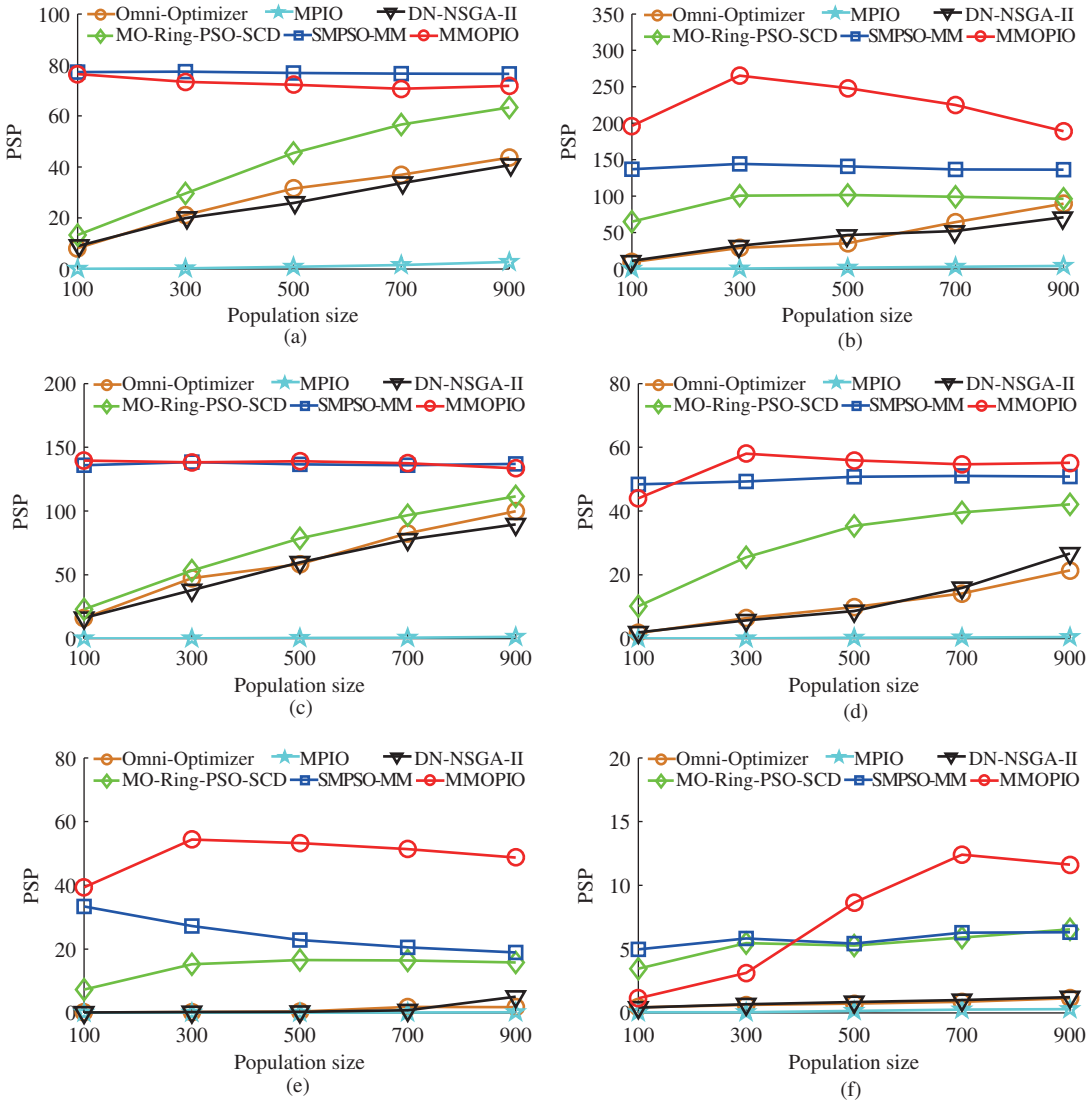

Figure 9 (Color online) Achieved PSP values of different algorithms with different population sizes. (a) PSP values on MMF1; (b) PSP values on MMF3; (c) PSP values on MMF7; (d) PSP values on MMF8; (e) PSP values on SYM-PART-simple; (f) PSP values on Omni-test.

are different. MMF1, MMF3, and MMF7 have 2 PSs each. Moreover, the numbers of PSs of MM8, SYM-PART-simple, and Omni-test are 4, 9, and 27, respectively. Note that the PS number represents the degree of complexity of a function to a certain extent. The larger the PS number, the more complex is the test function.

Figure 9(a), (c), and (d) shows that the performance of the proposed MMOPIO is much better than those of the other comparison algorithms, except for SMPSO-MM, on MMF1, MMF7, and MMF8 with different population sizes. MMOPIO and SMPSO-MM have similar exploratory behaviors on MMF1 and MMF7 for both algorithms to use SOM for achieving better solutions. The superior performances of MMOPIO and SMPSO-MM demonstrate that SOM is effective in tackling MMOPs. As shown in these three subfigures, the performances of MMOPIO and SMPSO-MM maintain a relatively stable trend with different population sizes. However, the other four comparison algorithms, except for MPIO, perform much better when their population sizes increase. The experimental results demonstrate that the population size setting is sensitive for MO-Ring-PSO-SCD, DN-NSGA-II, and Omni-Optimizer on relatively simple MMOPs. However, MMOPIO and SMPSO-MM can achieve a proper balance with varying population sizes when the number of PSs of the test function is low.

Figure 9(b) and (e) shows that MMOPIO performs better than the other five algorithms on MMF3 and SYM-PART-simple. The most considerable PSP value is achieved with the population size set to 300. The performance of MMOPIO slowly weakens as the population size increases. This phenomenon may occur because the maximal number of function evaluations is equal to the product of the population size and the number of iterations, which will decrease when the population size increases. In this condition, MMOPIO may not possess the ability of convergence. Figure 9(f) shows that the PSP value of MMOPIO first increases with the population size, and then decreases when the population size arrives at 700 for Omni-test. MMOPIO performs much better than the other compared algorithms after the population size has been set to more than 500. Note that MMF3, SYM-PART-simple, and Omni-test are different because they are complicated functions that overlap in the decision space. As shown in Figure 9(b), (e), and (f), the largest PSP value of MO-Ring-PSO-SCD is achieved when the population size increases to 300, after which it will achieve a balance. The PSP values of DN-NSGA-II and Omni-Optimizer increase slightly when the population sizes increase. MPIO achieves the lowest PSP values with varying population sizes on these three complex test functions. SMPSO-MM achieves a balanced performance on MMF3 and Omni-test. The largest value of PSP on SYM-PART-simple is achieved when the population size is 100, and then, the value decreases slowly when the population size increases.

As shown in Figure 9(a)–(f), the six comparison algorithms can be ranked according to their performance: MMOPIO, SMPSO-MM, MO-Ring-PSO-SCD, DN-NSGA-II, Omni-Optimizer, and MPIO. Comparing the performance of MPIO with MMOPIO on these six different test functions, we can conclude that the SOM strategy proposed in this paper can significantly improve the ability of MPIO in solving MMOPs. The experimental results show that most of these six comparison algorithms achieve the largest PSP values on different test functions when the population size is 700 or 900. Taking computational cost into consideration, we set the population size to 700 in the proposed MMOPIO to solve MMOPs.

5 Conclusion

For solving MMOPs, we propose a novel multimodal multi-objective pigeon-inspired optimization algorithm. A newly invented but effective PIO, together with the SOM strategy, was conducted to address the MMOPs, which was the first time for PIO to be adopted for solving MMOPs.

In MMOPIO, the framework of a basic PIO is simplified by introducing a consolidation parameter. SOM and the SCD calculation mechanism mainly realize the manipulations to improve the solution distribution. The elite learning strategy, together with the mutation operation, is conducive to balancing the convergence and diversity.

The experimental results demonstrated that the proposed MMOPIO outperforms the other five comparison algorithms and shows promising versatility for different MMOPs, especially for the test function MMF7 with irregular PSs and three complicated test instances, which were SYM-PART-simple, SYM-PART-rotated, and Omni-test (n=3).

In the future, it would be worthwhile to explore the potentialities of the improved MMOPIO in solving more complex MMOPs, such as MMOPs with multiple local PSs and with irregular PFs, and even real-world problems. Moreover, further investigation into applying more effective manipulation strategies on solution distribution in other meta-heuristic algorithms is desirable.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61473266, 61673404, 61305080, 61806179, 61876169, U1304602), China Postdoctoral Science Foundation (Grant Nos. 2014M552013, 2017M622373), Project Supported by the Research Award Fund for Outstanding Young Teachers in Henan Provincial Institutions of Higher Education of China (Grant No. 2014GGJS-004), Program for Science and Technology Innovation Talents in Universities of Henan Province in China (Grant Nos. 16HASTIT041, 16HASTIT033), and Scientific and Technological Project of Henan Province (Grant No. 152102210153).

References

1 Ali M Z, Awad N H, Duwairi R M. Multi-objective differential evolution algorithm with a new improved mutation strategy. Int J Artif Intell, 2016, 14: 23–41
2 Gong D M, Qin N N, Sun X Y. Evolutionary algorithms for optimization problems with uncertainties and hybrid indices. Inf Sci, 2011, 181: 4124–4138
3 Guan X M, Zhang X J, Lv R L, et al. A large-scale multi-objective flights conflict avoidance approach supporting 4D trajectory operation. Sci China Inf Sci, 2017, 60: 112202
4 Qu B Y, Zhou Q, Xiao J M, et al. Large-scale portfolio optimization using multiobjective evolutionary algorithms and preselection methods. Math Problems Eng, 2017, 2017: 1–14
5 Tian Y, Cheng R, Zhang X, et al. An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. IEEE Trans Evol Comput, 2018, 22: 609–622
6 Liang J J, Zheng B, Qu B Y, et al. Multi-objective differential evolution algorithm based on fast sorting and a novel constraints handling technique. In: Proceedings of IEEE Congress on Evolutionary Computation, Beijing, 2014. 445–450
7 Gong D W, Sun J, Ji X. Evolutionary algorithms with preference polyhedron for interval multi-objective optimization problems. Inf Sci, 2013, 233: 141–161
8 Rong M, Gong D W, Zhang Y, et al. Multidirectional prediction approach for dynamic multiobjective optimization problems. IEEE Trans Cybern, 2018. doi: 10.1109/TCYB.2018.2842158
9 Zhang X, Zheng X, Cheng R, et al. A competitive mechanism based multi-objective particle swarm optimizer with fast convergence. Inf Sci, 2018, 427: 63–76
10 Liu Y P, Gong D W, Sun J, et al. A many-objective evolutionary algorithm using a one-by-one selection strategy. IEEE Trans Cybern, 2017, 47: 2689–2702
11 Liu Y P, Gong D W, Sun X, et al. Many-objective evolutionary optimization based on reference points. Appl Soft Comput, 2017, 50: 344–355
12 Gong D W, Sun J, Miao Z. A set-based genetic algorithm for interval many-objective optimization problems. IEEE Trans Evol Comput, 2018, 22: 47–60
13 Preuss M, Kausch C, Bouvy C, et al. Decision space diversity can be essential for solving multiobjective real-world problems. In: Proceedings of the 19th International Conference on Multiple Criteria Decision Making, Auckland, 2010. 367–377
14 Liang J J, Yue C T, Qu B Y. Multimodal multi-objective optimization: a preliminary study. In: Proceedings of IEEE Congress on Evolutionary Computation, Vancouver, 2016. 2454–2461
15 Liang J J, Qu B Y, Mao X B, et al. Differential evolution based on fitness Euclidean-distance ratio for multimodal optimization. Neurocomputing, 2014, 137: 252–260
16 Qu B Y, Suganthan P N, Liang J J. Differential evolution with neighborhood mutation for multimodal optimization. IEEE Trans Evol Comput, 2012, 16: 601–614
17 Liang J J, Ma S T, Qu B Y, et al. Strategy adaptative memetic crowding differential evolution for multimodal optimization. In: Proceedings of IEEE Congress on Evolutionary Computation, Brisbane, 2012. 1–7
18 Deb K, Tiwari S. Omni-optimizer: a procedure for single and multi-objective optimization. In: Proceedings of International Conference on Evolutionary Multi-Criterion Optimization, Guanajuato, 2005. 47–61
19 Yue C T, Qu B Y, Liang J J. A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems. IEEE Trans Evol Comput, 2018, 22: 805–817
20 Liang J, Guo Q Q, Yue C T, et al. A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi-objective problems. In: Proceedings of International Conference on Swarm Intelligence, Shanghai, 2018. 550–560
21 Liang J J, Chan C C, Huang V L, et al. Improving the performance of a FBG sensor network using a novel dynamic multi-swarm particle swarm optimizer. In: Proceedings of SPIE - The International Society for Optical Engineering, Boston, 2005. 373–378
22 Liang J J, Pan Q K, Chen T J, et al. Solving the blocking flow shop scheduling problem by a dynamic multi-swarm particle swarm optimizer. Int J Adv Manuf Technol, 2011, 55: 755–762
23 Liang J J, Song H, Qu B Y, et al. Comparison of three different curves used in path planning problems based on particle swarm optimizer. Math Problems Eng, 2014, 2014: 1–15
24 Yang Q, Chen W N, Yu Z, et al. Adaptive multimodal continuous ant colony optimization. IEEE Trans Evol Comput, 2017, 21: 191–205
25 Duan H, Qiao P. Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. Int J Intel Comp Cyber, 2014, 7: 24–37
26 Xin L, Xian N. Biological object recognition approach using space variant resolution and pigeon-inspired optimization for UAV. Sci China Technol Sci, 2017, 60: 1577–1584
27 Lei X, Ding Y, Wu F X. Detecting protein complexes from DPINs by density based clustering with Pigeon-Inspired Optimization Algorithm. Sci China Inf Sci, 2016, 59: 070103
28 Qiu H X, Duan H B. Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. Sci China Technol Sci, 2015, 58: 1915–1923
29 Kohonen T. Automatic formation of topological maps of patterns in a self-organizing system. In: Proceedings of the 2nd Scandinavian Conference on Image Analysis, Simula, 1981. 214–220
30 Liu G, Yang H. Self-organizing network for variable clustering. Ann Oper Res, 2018, 263: 119–140

31 Jin H, Shum W H, Leung K S, et al. Expanding self-organizing map for data visualization and cluster analysis. Inf Sci, 2004, 163: 157–173
32 Tsai W P, Huang S P, Cheng S T, et al. A data-mining framework for exploring the multi-relation between fish species and water quality through self-organizing map. Sci Total Environ, 2017, 579: 474–483
33 Zhang H, Zhou A, Song S, et al. A self-organizing multiobjective evolutionary algorithm. IEEE Trans Evol Comput, 2016, 20: 792–806
34 Chen J H, Su M C, Cao R, et al. A self organizing map optimization based image recognition and processing model for bridge crack inspection. Autom Constr, 2017, 73: 58–66
35 Gu F, Cheung Y M. Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm. IEEE Trans Evol Comput, 2018, 22: 211–225
36 Haykin S S. Neural Networks and Learning Machines. Beijing: China Machine Press, 2009
37 Rudolph G, Naujoks B, Preuss M. Capabilities of EMOA to detect and preserve equivalent pareto subsets. In: Proceedings of International Conference on Evolutionary Multi-Criterion Optimization, Matsushima, 2007. 36–50
38 Tang L, Wang X. A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems. IEEE Trans Evol Comput, 2013, 17: 20–45
39 Zhou A M, Zhang Q F, Jin Y C. Approximating the set of pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm. IEEE Trans Evol Comput, 2009, 13: 1167–1189