A Multipopulation Evolutionary Algorithm for Solving Large-Scale Multimodal Multiobjective Optimization Problems

Ye Tian, *Member, IEEE*, Ruchen Liu, Xingyi Zhang, *Senior Member, IEEE*, Haiping Ma, Kay Chen Tan, *Fellow, IEEE*, and Yaochu Jin, *Fellow, IEEE*

***Abstract*—Multimodal multiobjective optimization problems (MMOPs) widely exist in real-world applications, which have multiple equivalent Pareto-optimal solutions that are similar in the objective space but totally different in the decision space. While some evolutionary algorithms (EAs) have been developed to find the equivalent Pareto-optimal solutions in recent years, they are ineffective to handle large-scale MMOPs having a large number of variables. This article thus proposes an EA for solving large-scale MMOPs with sparse Pareto-optimal solutions, i.e., most variables in the optimal solutions are 0. The proposed algorithm explores different regions of the decision space via multiple subpopulations and guides the search behavior of the subpopulations via adaptively updated guiding vectors. The guiding vector for each subpopulation not only provides efficient convergence in the huge search space but also differentiates its search direction from others to handle the multimodality. While most existing EAs solve MMOPs with 2–7 decision variables, the proposed algorithm is shown to be effective for benchmark MMOPs with up to 500 decision variables. Moreover, the proposed algorithm also produces a better result than state-of-the-art methods for the neural architecture search.**

***Index Terms*—Evolutionary algorithm (EA), large-scale optimization, multimodal multiobjective optimization, neural architecture search, sparse Pareto-optimal solutions.**

Manuscript received May 7, 2020; revised October 9, 2020; accepted December 10, 2020. Date of publication December 15, 2020; date of current version May 28, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0100100; in part by the National Natural Science Foundation of China under Grant 61672033, Grant 61822301, Grant 61876123, Grant 61906001, and Grant U1804262; in part by the Hong Kong Scholars Program under Grant XJ2019035; in part by the Anhui Provincial Natural Science Foundation under Grant 1808085J06 and Grant 1908085QF271; in part by the State Key Laboratory of Synthetical Automation for Process Industries under Grant PAL-N201805; in part by the CCF-Tencent Open Research Fund under Grant RAGR20200121; in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CityU11202418 and Grant CityU11209219; and in part by the Royal Society International Exchanges Program under Grant IEC\NSFC\170279. *(Corresponding author: Xingyi Zhang.)*

Ye Tian and Haiping Ma are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, Institutes of Physical Science and Information Technology, Anhui University, Hefei 230601, China (e-mail: field910921@gmail.com; hpma@ahu.edu.cn).

Ruchen Liu and Xingyi Zhang are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230601, China (e-mail: lrchen_1996@163.com; xyzhanghust@gmail.com).

Kay Chen Tan is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: kaytan@cityu.edu.hk).

Yaochu Jin is with the Department of Computer Science, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: yaochu.jin@surrey.ac.uk).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TEVC.2020.3044711.

Digital Object Identifier 10.1109/TEVC.2020.3044711

I. Introduction

In SOME real-world multiobjective optimization problems (MOPs), such as the architecture layout design [1] and rocket engine design [2], there exist multiple Pareto-optimal solutions, corresponding to the same or similar objective values, which are known as multimodal MOPs (MMOPs) [3]. Although the Pareto-optimal solutions of MMOPs have similar values in the objective space, they are considerably different in the decision space, hence, it is worth finding these equivalent solutions to provide more options to the decision maker [4]. This requirement makes MMOPs very challenging for the existing MOEAs, since most MOEAs aim at finding a set of well-converged solutions with good diversity in the objective space, which means that only a single decision vector is found for each optimal solution in the objective space. In contrast, all the different decision vectors corresponding to the same optimal solution should be found for solving MMOPs.

To find multiple equivalent Pareto-optimal solutions for each objective vector, a well-converged population with good diversity in both the objective and decision spaces is needed [5]. For this aim, some algorithms group the solutions in the decision space [6], [7], some measure the crowding distance of each solution in both the objective and decision spaces [4], [8], some preserve the diversity of one population in the objective space and preserve the diversity of another population in the decision space [5], [9], and yet others maintain multiple solutions with good diversity in the decision space for each single-objective subproblem in decomposition-based MOEAs [10], [11].

In spite of the effectiveness of these MOEAs on benchmark MMOPs, they only provide good optimization performance for a small number of decision variables. While most state-of-the-art MOEAs have been tested on MMOPs with 2–7 decision variables [4], [5], [12], real-world problems often have dozens or even hundreds of decision variables [13]. It is obvious that large-scale MMOPs are very challenging, since existing MOEAs for MMOPs do not develop specific strategies for handling large-scale optimization, and can hardly find the

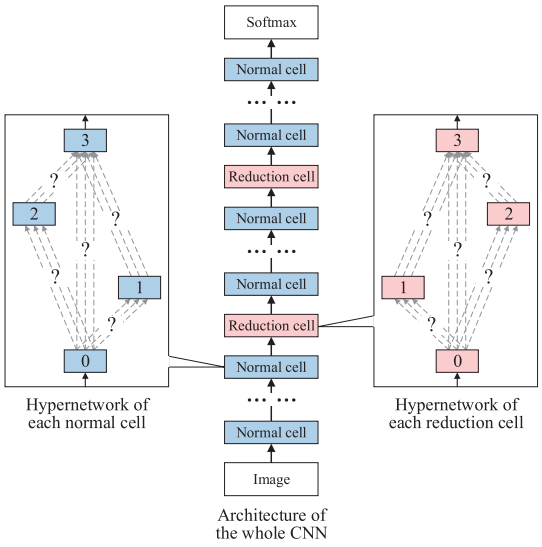

Fig. 1. Illustration of the one-shot neural architecture search. It aims to find the best connections (i.e., layers) among many connections in the two hypernetworks, which is a large-scale MMOP with sparse Pareto-optimal solutions.

Pareto-optimal solutions in a huge decision space due to the curse of dimensionality [14]. Even if a few Pareto-optimal solutions can be found arbitrarily, most MOEAs cannot find multiple equivalent Pareto-optimal solutions, since it is difficult to measure the population diversity in a high-dimensional decision space [15].

As a consequence, MOEAs for solving large-scale MMOPs should be capable of driving the solutions to quickly converge in the huge decision space and make them converge toward different directions to find all the equivalent Pareto-optimal solutions. In this work, we propose an evolutionary algorithm (EA) having the above abilities for solving the large-scale MMOPs, whose Pareto-optimal solutions are sparse. A sparse Pareto-optimal solution means that most of its decision variables are 0, which widely exists in many real-world MMOPs, such as neural network training [13], feature selection [16], and neural architecture search [17]. Taking the neural architecture search as an example, which is currently one of the hottest topics in deep learning, the one-shot neural architecture search aims to find the best architectures for the normal cell and reduction cell in the convolutional neural network (CNN) [18]. As illustrated in Fig. 1, the search space is represented by two hypernetworks with multiple candidate connections (i.e., layers) between each pair of nodes, and an architecture is a subgraph consisting of one connection between each pair of nodes. First, it is a large-scale optimization problem with sparse optimal solutions, since the hypernetworks contain a large number of candidate connections while the selected connections account for a very small proportion; in the experiment in Section IV, the number of candidate connections is 224 but only 28 connections need to be selected, i.e., there are 224 real decision variables and 196 of them should be 0. Second, it is also a multimodal optimization problem [19], since many works reported different architectures with very similar classification performance [20], [21], and multiple architectures should be found in order to provide sufficiently different members in an ensemble [22]. The main contributions of this work are summarized as follows.

1) A multipopulation multimodal EA is proposed to solve large-scale MMOPs with sparse Pareto-optimal solutions, termed MP-MMEA. The main novelty of MP-MMEA lies in the guiding vector-assisted evolution of subpopulations, where a guiding vector is adaptively adjusted to improve both the convergence and diversity of each population. The guiding vectors cannot only lead the subpopulations to evolve toward sparse optimal solutions efficiently but also diversify the search directions of subpopulations in the decision space, thus enabling the algorithm to find multiple equivalent Pareto-optimal solutions. Besides, a merge-and-divide operation is proposed to automatically adjust the number of subpopulations during the search process.
2) The proposed MP-MMEA is employed to search for the optimal architectures of deep CNNs. According to the experimental results on CIFAR-10 [23], it is interesting to observe that the neural architectures found by MP-MMEA have slightly lower test accuracies than some state-of-the-art methods, but the ensemble of the found neural architectures obtains a higher test accuracy than the ensemble of the state-of-the-art ones. The results reveal that the diverse solutions of MMOPs not only provide more options to the decision maker but are also useful in certain specific fields such as ensemble learning.
3) A sparse multimodal multiobjective test suite is designed for performance assessment. In contrast to the existing benchmark MMOPs having small-scale and nonsparse Pareto-optimal solutions, the proposed benchmark MMOPs are scalable in terms of the number of objectives, decision variables, equivalent Pareto-optimal sets, and sparsity of Pareto-optimal solutions, able to assess the ability of MOEAs in finding sparse equivalent Pareto-optimal solutions in a huge decision space. Experimental results indicate that the proposed MP-MMEA is capable of solving the proposed benchmark MMOPs with up to 500 decision variables.

The remainder of this article is organized as follows. Section II reviews the state-of-the-art MOEAs for MMOPs and provides the motivation of this work. Section III describes the details of the proposed MP-MMEA, and Section IV presents the experimental results on the neural architecture search. Section V details the proposed test suite and examines the performance of MP-MMEA on the test suite. Finally, conclusions and future work are given in Section VI.

II. Related Work and Motivation

A. Existing MOEAs for MMOPs

Since the concept of multimodal multiobjective optimization was first proposed in 2005 [3], [8], a number of MOEAs have been designed for MMOPs. In the following, we review some popular and state-of-the-art MOEAs for MMOPs. Readers are referred to [22] and [24] for a comprehensive survey of all the relevant algorithms and techniques.

Omni-optimizer [8] is one of the first MOEAs for solving MMOPs. In order to preserve the population diversity in both the objective and decision spaces, it measures the crowding distance of each solution in both the objective and decision spaces, and either the crowding distance in the objective space or in the decision space is used in different cases. Later, similar ideas were adopted in other MOEAs for solving MMOPs [25], [26].

In [4], a multiobjective particle swarm optimization algorithm was proposed for solving MMOPs, termed MO_Ring_PSO_SCD. To effectively identify a large number of Pareto-optimal solutions over the whole decision space, a ring topology [27] was used to create niches for updating the particles. Additionally, the diversity measurement technique in Omni-optimizer was also adopted and enhanced with a new strategy for handling boundary particles. Later on, MO_Ring_PSO_SCD was enhanced in [28] and employed for solving the feature selection problem in [16].

Instead of calculating the crowding distance in the decision space, a two archive strategy and a recombination strategy were proposed in TriMOEA-TA&R [5] to preserve the population diversity. First, the properties of decision variables and the relations among them are analyzed, and the independent convergence-related decision variables are detected. Then, TriMOEA-TA&R evolves a convergence archive to locate diverse well-converged solutions in the independent convergence-related decision subspace and evolves a diversity archive to preserve the population diversity in the remaining decision subspace and the objective space.

In [29], a framework for enhancing the performance of decomposition-based MOEAs on MMOPs was proposed. Specifically, each offspring solution is first assigned to a subproblem via an assignment operation, and the offspring solution is compared to the others assigned to the same subproblem and close to it in the decision space, where the better ones are kept and the worse ones are ignored. The effectiveness of this framework was examined based on six decomposition-based MOEAs.

To remedy the weakness of existing MOEAs for solving MMOPs with an imbalance between convergence and diversity in the decision space, a convergence-penalized density method-based MOEA was proposed in [12], termed CPDEA. Since the imbalance between convergence and diversity leads to different difficulties in finding the equivalent Pareto-optimal solutions, the CPDEA uses a convergence-penalized density method as a selection criterion, rather than the convergence-first selection criteria used in the existing MOEAs. In addition, a double k-nearest neighbor method was used in CPDEA to measure the diversity of each solution in both the objective and decision spaces.

In [30], an MOEA with dual clustering was proposed for solving MMOPs. The MOEA divides the population in the decision space via a neighborhood-based clustering method and divides the population in the objective space via a hierarchical clustering method. The MOEA can maintain the population diversity in both the decision and objective spaces, and preserve the local Pareto-optimal sets with acceptable quality.

B. Motivation of This Work

It can be observed from the review of the existing MOEAs for MMOPs that most of them aim to find equivalent Pareto-optimal solutions by preserving the population diversity in the decision space. However, this idea is ineffective for large-scale MMOPs having a huge decision space, since it is difficult to measure the distance between points in a high-dimensional space. In particular, the crowding distance is ineffective to estimate the diversity in a high-dimensional space [31] and the Euclidean distance is also showed to be less meaningful in a high-dimensional space [32].

An alternative idea for handling large-scale MMOPs is to use multiple subpopulations, where each subpopulation is evolved independently for approximating one of the equivalent Pareto-optimal sets, and the population diversity in the decision space does not need to be measured. Besides, evolving multiple populations can retain some solutions that are dominated by the Pareto-optimal solutions but far from them in the decision space, which is more practical for solving real-world MMOPs [24]. It is worth noting that although a few MOEAs have been proposed to solve MMOPs via multiple subpopulations [28], [33], they are unsuited for large-scale MMOPs since they need to group the solutions based on the Euclidean distances in the decision space. More importantly, they do not explicitly distinguish the solutions in different subpopulations, which may not be able to spread the solutions across the decision space in order to find the equivalent Pareto-optimal solutions.

On the other hand, the existing MOEAs mainly focus on the diversification of the population in the decision space since it is assumed that all the equivalent Pareto-optimal solutions are easy to be approximated and thus are just ignored by general MOEAs. However, the Pareto-optimal solutions of large-scale MMOPs are difficult to be found due to the curse of dimensionality [34], and existing MOEAs for MMOPs do not develop specific strategies for large-scale optimization. It is worth noting that although large-scale MOPs have been properly handled by some dedicated MOEAs, these MOEAs cannot be used to solve large-scale MMOPs. This is because most MOEAs for large-scale MOPs use dimensionality reduction techniques [35], [36] or problem transformation strategies [37], [38] to drastically reduce the decision space, which drive the solutions toward an optimal subspace but ignore most parts of the decision space, and thus cannot find multiple equivalent Pareto-optimal solutions. In addition, some other MOEAs solve the large-scale MOPs by dividing the decision variables into several groups and optimizing each group of decision variables alternatively [39], [40]. These MOEAs usually consume a large number of function evaluations for the division of decision variables, which is inefficient for solving real-world large-scale MMOPs.

This work proposes a multipopulation EA for solving the large-scale MMOPs with sparse Pareto-optimal solutions. To address the above issues caused by a huge decision space, a guiding vector is adaptively updated for each subpopulation in the proposed MP-MMEA. The guiding vectors are used to distinguish the search directions of different subpopulations,

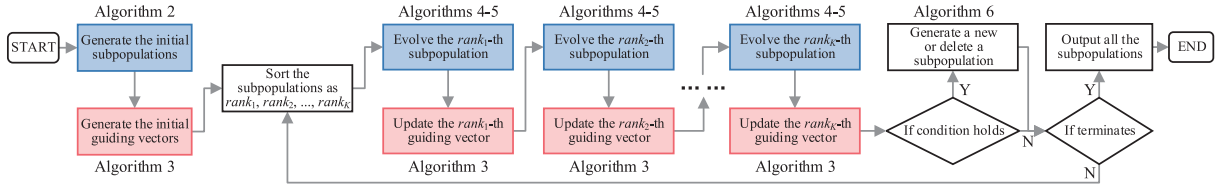

Fig. 2. Flow chart of the proposed MP-MMEA.

Algorithm 1: Framework of MP-MMEA

Input: N (size of combined population)
Output: P (combined population)

1 $P \leftarrow Initialization(N)$;
2 $K \leftarrow 2$; //Initial number of subpopulations
3 $[subP_1, \ldots, subP_K] \leftarrow$ Randomly divide P into K subpopulations with equal size;
4 **while** *termination criterion is not fulfilled* **do**
5 $[rank_1, \ldots, rank_K] \leftarrow$ Sort all the subpopulations according to the average non-dominated ranks of solutions in each subpopulation;
6 $\mathbf{r} \leftarrow$ A binary vector of zeros;
7 **for** $i = 1$ *to* K **do**
8 $\mathbf{gv}_{rank_i} \leftarrow UpdateGV(subP_{rank_i})$;
9 $Q \leftarrow$ Select $2\lfloor N/K \rfloor$ parents from $subP_{rank_i}$ via binary tournament selection;
10 $subP_{rank_i} \leftarrow subP_{rank_i} \cup Variation(Q, \mathbf{gv}_{rank_i})$;
11 $[subP_{rank_i}, \mathbf{r}] \leftarrow EnvironmentalSelection(subP_{rank_i}, \lfloor N/K \rfloor, \mathbf{gv}_{rank_i}, \mathbf{r})$;
12 **if** *the generation number is a multiple of* T **then**
13 $[K, subP_1, \ldots, subP_K] \leftarrow MergeAndDivide(N, K, subP_1, \ldots, subP_K)$;
14 $P \leftarrow subP_1 \cup \cdots \cup subP_K$;
15 **return** P;

which is based on the Hamming distances between solutions and guiding vectors in a binary space. Moreover, the guiding vector of each subpopulation is involved in the genetic operators for guiding the subpopulation to quickly find the Pareto-optimal solutions along its own search direction. In the next section, the details of MP-MMEA are elaborated.

III. Proposed Multipopulation Multimodal Evolutionary Algorithm

A. Procedure of MP-MMEA

The procedure of the proposed MP-MMEA is illustrated in Fig. 2, and the framework of MP-MMEA is presented in Algorithm 1. To begin with, N solutions are generated to form the initial population (line 1). Then, the initial number of subpopulations K is set to 2 (line 2), and the initial population is randomly divided into K subpopulations with equal size (line 3). In each generation of MP-MMEA, the solutions in all the subpopulations are combined and sorted by nondominated sorting [41], and the subpopulations are sorted according to the average nondominated ranks of solutions in each subpopulation (line 5). Since MP-MMEA focuses on solving MMOPs with two objectives, it is reasonable to distinguish between the qualities of solutions via the Pareto dominance relation. After sorting the subpopulations, they are evolved independently in the sorted order (line 7), i.e., the subpopulations with better solutions are evolved first. The evolution of each subpopulation consists of four steps, i.e., updating the guiding vector (line 8), selecting parents via binary tournament selection (i.e., two solutions are randomly picked up each time, and the one with a smaller nondominated rank is selected as a parent; while if the two solutions have the same nondominated rank, the one with a larger crowding distance is selected) (line 9), generating offspring solutions by the guiding vector-assisted genetic operators (line 10) and truncating the combined subpopulation by the guiding vector-assisted environmental selection (line 11). Finally, the merge-and-divide operation is periodically performed to adaptively adjust the number of subpopulations (line 13).

It is worth noting that most existing population initialization strategies and recombination operators can hardly generate sparse solutions. To solve this problem, inspired by some existing MOEAs for solving MMOPs [13] and sparse MOPs [42], each solution $\mathbf{x}$ in MP-MMEA is represented by a real vector **real** and a binary vector **bin** rather than the original decision vector $(x_1, x_2, \ldots)$, where $x_i = \text{real}_i \times \text{bin}_i$ and x_i is used for function evaluation. The real vector represents the decision variables of the solution in a continuous space, and the binary vector represents the search direction in a binary space. Hence, the optimization of the decision variables can be achieved by optimizing **real**, and many zero decision variables can be obtained by optimizing **bin**. As shown in Algorithm 2, MP-MMEA initializes the population based on the above representation, where the elements in all the real vectors are set to random values in the decision space, and the elements in all the binary vectors are initialized to 0. Then, for each binary vector, rand $\times D$ elements are selected and set to 1, where rand denotes a uniformly random value within [0, 1] and D denotes the number of decision variables. This way, many decision variables in the initial solutions can be 0 as many elements in the binary vectors remain 0.

It is noteworthy that a helper vector $\mathbf{r}$ is updated to guide the selection of elements from the binary vector **bin**. More specifically, the initial $\mathbf{r}$ is set to a vector of zeros, and each element in $\mathbf{r}$ is increased by 1, once the corresponding element in a binary vector is selected. When selecting an element from a binary vector, the binary tournament selection is performed according to $\mathbf{r}$, where a lower value in $\mathbf{r}$ corresponds to a higher probability to be selected. As a consequence, $\mathbf{r}$ can balance the selected elements in all the binary vectors. It is worth mentioning that $\mathbf{r}$ also makes contributions to the merge-and-divide operation for generating new subpopulations. In this case, $\mathbf{r}$ is set according to the guiding vectors of existing subpopulations rather than initialized to a vector of zeros.

Algorithm 2: *Initialization*(N)

Input: N (size of combined population), **r** (helper vector)
Output: P (initial combined population)

$D \leftarrow$ Number of decision variables;
$[\mathbf{real}^1, \ldots, \mathbf{real}^N] \leftarrow N$ real vectors with random values;
$[\mathbf{bin}^1, \ldots, \mathbf{bin}^N] \leftarrow N$ binary vectors of zeros;
if **r** *is not given* **then**
  r $\leftarrow 1 \times D$ vector of zeros;
for $i = 1$ *to* N **do**
  for $j = 1$ *to* $rand \times D$ **do**
    $[m, n] \leftarrow$ Randomly select two dimensions;
    if $r_m < r_n$ **then**
      $\text{bin}_m^i \leftarrow 1$;
      $r_m \leftarrow r_m + 1$;
    else
      $\text{bin}_n^i \leftarrow 1$;
      $r_n \leftarrow r_n + 1$;
$P \leftarrow$ Generate N solutions by $\mathbf{real}^1, \ldots, \mathbf{real}^N$ and $\mathbf{bin}^1, \ldots, \mathbf{bin}^N$;
return P;

Algorithm 3: *UpdateGV*($subP$)

Input: $subP$ (subpopulation)
Output: **gv** (updated guiding vector)

$R \leftarrow$ Find the non-dominated solutions in $subP$;
$\mathbf{v} \leftarrow 1 \times D$ vector of zeros;
for $i = 1$ *to* $|R|$ **do**
  v $\leftarrow$ Use the i-th solution in R to update **v** by (1);
gv $\leftarrow$ Update **gv** by (2);
return **gv**;

B. Guiding Vector-Assisted Search

The guiding vector defines the search direction of each subpopulation, which is updated according to the sparse distribution of the solutions in the subpopulation and, in turn, influences the generation of offspring solutions for the subpopulation. For updating the guiding vector $\mathbf{gv}^i$ of the ith subpopulation, as shown in Algorithm 3, the nondominated solutions in the subpopulation are first determined to form a solution set R. Then, a real vector $\mathbf{v}$ is set to a vector of zeros, and for each solution $\mathbf{x} \in R$ and its $\lfloor 0.1|R| \rfloor$th nearest neighbor $\mathbf{y}$ in the decision space, each element in $\mathbf{v}$ is updated by

$$v_i = \begin{cases} v_i + 1, & \text{if } \text{bin}_i^x = 1 \text{ and } \text{bin}_i^y = 1 \\ v_i + 0, & \text{if } \text{bin}_i^x = 0 \text{ and } \text{bin}_i^y = 0 \\ v_i + 0.5, & \text{otherwise} \end{cases} \quad (1)$$

where v_i denotes the ith element in $\mathbf{v}$ and bin_i^x denotes the ith element in the binary vector of solution $\mathbf{x}$. After repeating the above procedure for $|R|$ times, the guiding vector $\mathbf{gv}^i$ can be updated by the obtained $\mathbf{v}$

$$\mathbf{gv}^i = \frac{1}{2}\mathbf{gv}^i + \frac{1}{2|R|}\mathbf{v}. \quad (2)$$

The initial value of $\mathbf{gv}^i$ is set to $(1/|R|)\mathbf{v}$.

In this way, the guiding vector $\mathbf{gv}^i \in [0, 1]^D$ can represent the probability of each decision variable equal to nonzero in the ith subpopulation. Therefore, the offspring solutions generated based on the guiding vector can have a similar sparse distribution to the solutions in the subpopulation, and different guiding vectors can drive the subpopulations to search for different sparse Pareto-optimal solutions. That is, the guiding vectors can accelerate the convergence of the subpopulations toward different directions.

Algorithm 4: *Variation*(P, **gv**)

Input: P (parent population), **gv** (guiding vector)
Output: Q (offspring population)

$Q \leftarrow \emptyset$;
while $P \neq \emptyset$ **do**
  $[\mathbf{x}, \mathbf{y}] \leftarrow$ Randomly select two parents from P;
  $P \leftarrow P \setminus \{\mathbf{x}, \mathbf{y}\}$;
  $\mathbf{real}^q \leftarrow$ Perform simulated binary crossover and polynomial mutation on $\mathbf{real}^x$ and $\mathbf{real}^y$;
  $\mathbf{bin}^q \leftarrow \mathbf{bin}^x$;
  //Crossover
  for $i = 1$ *to* D **do**
    if $\text{bin}_i^x \neq \text{bin}_j^y$ **then**
      if $rand < gv_i$ **then**
        $\text{bin}_i^q \leftarrow 1$;
      else
        $\text{bin}_i^q \leftarrow 0$;
  //Mutation
  if $rand < 0.5$ **then**
    $[m, n] \leftarrow$ Randomly select two dimensions where $\mathbf{bin}^q$ is 1;
    if $gv_m < gv_n$ **then**
      $\mathbf{bin}_m^q \leftarrow 0$;
    else
      $\mathbf{bin}_n^q \leftarrow 0$;
  else
    $[m, n] \leftarrow$ Randomly select two dimensions where $\mathbf{bin}^q$ is 0;
    if $gv_m > gv_n$ **then**
      $\mathbf{bin}_m^q \leftarrow 1$;
    else
      $\mathbf{bin}_n^q \leftarrow 1$;
  $Q \leftarrow Q \cup \{\mathbf{q}\}$;
return Q;

In order to use the guiding vector to accelerate the convergence of each subpopulation, the guiding vector is embedded in the genetic operators as shown in Algorithm 4. As can be seen, two parents are randomly selected to generate one offspring solution **q** each time, where the real vector $\mathbf{real}^q$ is generated by simulated binary crossover [43] and polynomial mutation [44], while the binary vector $\mathbf{bin}^q$ is generated by the guiding vector-assisted crossover and mutation. For the crossover operator, $\mathbf{bin}^q$ is first set to the same as one parent $\mathbf{bin}^x$. Then, for each dimension where the two parents $\mathbf{bin}^x$ and $\mathbf{bin}^y$ are different, the corresponding element in $\mathbf{bin}^q$ is set to 1 if a random value is smaller than the corresponding element in the guiding vector **gv**, and set to 0, otherwise. For the mutation operator, either the following two operations are performed with the same probability: 1) selecting one dimension where $\mathbf{bin}^q$ is 1 via binary tournament selection according to the value of each dimension in **gv** (the smaller the better) and

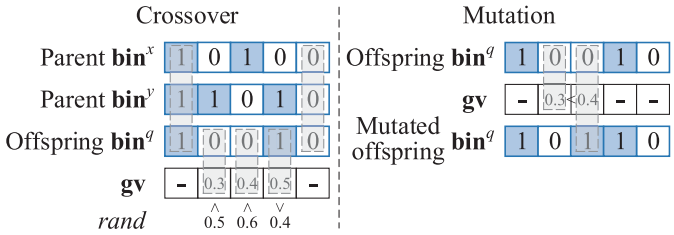

Fig. 3. Illustrative example of the guiding vector-assisted crossover and mutation operators.

Algorithm 5: *EnvironmentalSelection*($subP$, N, $\mathbf{gv}$, $\mathbf{r}$)

Input: $subP$ (subpopulation), N (size of subpopulation), $\mathbf{gv}$ (guiding vector), $\mathbf{r}$ (helper vector)
Output: $subP$ (subpopulation), $\mathbf{r}$ (helper vector)

if $\mathbf{r}$ *is given* **then**
  Calculate the Hamming distance between the binary vector of each solution in $subP$ and $\mathbf{r}$ as a new objective of the solution;
$[F_1, F_2, \dots] \leftarrow NondominatedSort(subP)$;
$k \leftarrow \min_i |F_1 \cup \dots \cup F_i| \geq N$;
$F_k \leftarrow$ Delete $|F_1 \cup \dots \cup F_k| - N$ solutions from F_k with the worst crowding distance;
$subP \leftarrow F_1 \cup \dots \cup F_k$;
if $\mathbf{r}$ *is given* **then**
  $\mathbf{r} \leftarrow$ Use $\mathbf{gv}$ to update $\mathbf{r}$ by (3);
return $subP$ and $\mathbf{r}$;

setting the corresponding element in $\mathbf{bin}^q$ to 0 or 2) selecting one dimension where $\mathbf{bin}^q$ is 0 via binary tournament selection according to the value of each dimension in $\mathbf{gv}$ (the larger, the better) and setting the corresponding element in $\mathbf{bin}^q$ to 1. As a consequence, the proposed genetic operators take advantage of the guiding vector with reasonable randomness, where each variable of the offspring solution is more likely to be 0 if the corresponding element in the guiding vector is smaller, and vice versa. Hence, the subpopulation can generally converge toward the direction determined by the guiding vector. An illustrative example of the proposed genetic operators is presented in Fig. 3.

To differentiate the search directions of subpopulations, the guiding vectors are also involved in the environmental selection as shown in Algorithm 5. As presented in Algorithm 5, the first environmental selection at each generation (where $\mathbf{r}$ is a vector of zeros at this time) of MP-MMEA is totally the same as NSGA-II [45], i.e., the subpopulation is sorted by nondominated sorting and the last selected nondominated front is truncated by crowding distance. To make the search directions of the subsequent subpopulations different from the previous ones, a helper vector $\mathbf{r}$ is maintained to record which dimensions of the decision variables have been searched by the previous subpopulations at the current generation. After truncating a subpopulation, each element of $\mathbf{r}$ is updated by the guiding vector $\mathbf{gv}$ of the subpopulation

$$r_i = \begin{cases} 1, & \text{if } r_i = 1 \text{ or } \mathrm{gv}_i > 0.5 \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

That is, the ith dimension is searched by the subpopulation if $gv_i > 0.5$ and $\mathbf{r}$ can record all the dimensions having been searched. Afterward, the Hamming distance between the binary vector of each solution in the subsequent subpopulations and $\mathbf{r}$ is calculated, and this distance value is attached to the objective vector of the solution as a new objective (the larger the better). By doing so, the solutions in the subsequent subpopulations can have large Hamming distances to $\mathbf{r}$, so that unexplored dimensions can be searched and the search directions of the subsequent subpopulations are different from the previous ones.

Algorithm 6: *MergeAndDivide*(N, K, $subP_1, \dots, subP_K$)

Input: N (size of combined population), K (number of subpopulations), $subP_1, \dots, subP_K$ (subpopulations)
Output: K (number of subpopulations), $subP_1, \dots, subP_K$ (subpopulations)

$[subP_a, subP_b] \leftarrow$ Select the two most similar subpopulations based on the similarity calculated by (4);
if *the similarity of* $subP_a$ *and* $subP_b$ *is larger than* 0.5 **then**
  //Merge
  $subP_a \leftarrow EnvironmentalSelection(subP_a \cup subP_b, \lfloor N/(K-1) \rfloor)$;
  Delete $subP_b$;
  $K \leftarrow K - 1$;
else if *all subpopulations have non-dominated solutions* **then**
  //Divide
  $[rank_1, \dots, rank_K] \leftarrow$ Sort all the subpopulations according to the average non-dominated ranks of solutions in each subpopulation;
  $\mathbf{r} \leftarrow$ A binary vector of zeros;
  for $i = 1$ *to* K **do**
    $[subP_{rank_i}, \mathbf{r}] \leftarrow EnvironmentalSelection(subP_{rank_i}, \lfloor N/(K+1) \rfloor, \mathbf{gv}_{rank_i}, \mathbf{r})$;
  $subP_{K+1} \leftarrow Initialization(\lfloor N/(K+1) \rfloor, \sum_{i=1}^{K} \mathbf{gv}_i)$;
  $K \leftarrow K + 1$;
return K and $subP_1, \dots, subP_K$;

C. Merge-and-Divide Operation

To adaptively adjust the number of subpopulations for solving different MMOPs, a merge-and-divide operation is periodically performed at the end of each generation, the procedure of which is given in Algorithm 6. To begin with, the similarities between each two subpopulations are calculated by

$$\mathrm{sim}(\mathrm{subP}_a, \mathrm{subP}_b) = \frac{\sum_{i=1}^{D} \mathrm{bin}_i^x \times \mathrm{bin}_i^y}{\min(\sum_{i=1}^{D} \mathrm{bin}_i^x, \sum_{i=1}^{D} \mathrm{bin}_i^y)} \tag{4}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the solutions having the best convergence (i.e., smallest sum of objective values) in subpopulations subP_a and subP_b, respectively, and a larger value of $\mathrm{sim}(\mathrm{subP}_a, \mathrm{subP}_b)$ indicates a larger overlap between the search directions of the two subpopulations. Then, the two most similar subpopulations are identified, and they are merged to form a single subpopulation if the similarity between them is larger than a threshold of 0.5. While if the similarity is smaller than the threshold, the solutions in all the existing subpopulations are combined and sorted by nondominated sorting, and a new subpopulation is initialized if all the existing subpopulations have nondominated solutions; otherwise, no subpopulation will be generated or merged.

When merging two subpopulations, the union set of the two subpopulations is obtained and truncated by the environmental selection of Algorithm 5 without a helper vector. When generating a new subpopulation, all the existing subpopulations are truncated by the environmental selection with a helper vector, since the size of subpopulation is reduced from $\lfloor N/K \rfloor$ to $\lfloor N/(K+1) \rfloor$. Then, a new subpopulation is initialized by Algorithm 2, where the helper vector $\mathbf{r}$ is set to the sum of all the existing guiding vectors to make the search direction of the new subpopulation different from existing ones. It is worth mentioning that the idea of merging and dividing subpopulations has been adopted in the existing algorithms such as [46], which adaptively merges and splits bad performing subpopulations. While this algorithm aims to enhance the convergence performance of subpopulations, the purpose of the merge-and-divide operation in MP-MMEA is totally different, since MP-MMEA aims to diversify the search directions of subpopulations for finding multiple equivalent Pareto-optimal solutions.

D. Computational Complexity of MP-MMEA

The nondominated sorting of all the solutions has a time complexity of $O(MN^2)$, where M denotes the number of objectives and N denotes the size of the combined population. The update of guiding vectors has a time complexity of $K \times O(\lfloor N/K \rfloor D) = O(ND)$, where D denotes the number of decision variables. The generation of offspring solutions has a time complexity of $K \times O(\lfloor N/K \rfloor D) = O(ND)$. For the environmental selection, the time complexity of calculating the Hamming distances between solutions and helper vector is $K \times O(\lfloor N/K \rfloor D) = O(ND)$, the time complexity of nondominated sorting is $K \times O(M\lfloor N/K \rfloor^2) = O(MN^2/K)$, and the time complexity of calculating crowding distance is $K \times O(M\lfloor N/K \rfloor \log \lfloor N/K \rfloor) = O(MN \log(N/K))$. For the merge-and-divide operation, the time complexity is the same as the environmental selection. In short, the total time complexity of one generation of MP-MMEA is $O(MN(N+D))$.

IV. Experiments on Neural Architecture Search

A. Problem Definition of Neural Architecture Search

To study the performance of MP-MMEA, it is applied to search for optimal neural architectures of CNNs for an image classification task. As illustrated in Fig. 1, the one-shot neural architecture search framework uses each node in the hypernetworks to represent a latent representation (i.e., a feature map in CNN) and uses each edge to represent an operation (i.e., a layer) $o^{(i,j)}$ that transforms node i to node j [18]. Given a set of candidate operations O like convolution and max pooling, the goal of the neural architecture search is to find the best operation $o^{(i,j)} \in O$ between each pair of nodes.

Obviously, this is a combinatorial optimization problem that cannot be easily tackled by most metaheuristics in continuous search space. Hence, the operation mixing weight between node i and node j is parameterized by a vector $\alpha^{(i,j)}$ of dimension $|O|$. That is, a continuous optimization problem is established by treating each operation as the result of a softmax over all the candidate operations [21]

$$\bar{o}^{(i,j)}(x) = \sum_{o \in O} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in O} \exp(\alpha_{o'}^{(i,j)})} o(x) \tag{5}$$

where $\alpha_o^{(i,j)} \in [0,1]$ is the decision variable denoting the weight of the oth candidate operation between node i and node j. Therefore, for a hypernetwork having N nodes, the total number of decision variables will be $C_N^2 \times |O|$. After the search, the final neural architecture is obtained by replacing the mixed operation $\bar{o}^{(i,j)}$ by the most likely one, i.e., $o^{(i,j)} = \text{argmax}_{o \in O}\, \alpha_o^{(i,j)}$.

To summarize, the neural architecture search problem can be defined as

$$\begin{aligned} \text{Minimize } & f_1(A) = f_{\text{valid\_error}}(A) \\ & f_2(A) = f_{\text{complexity}}(A) \end{aligned} \tag{6}$$

where A is the set of all $\alpha_o^{(i,j)}$ and the decision space is $[0,1]^{C_N^2 \times |O|}$. For each solution A, a neural architecture can be determined and a CNN can be trained based on the given training samples; note that the CNN needs not be trained from scratch, since a parameter sharing mechanism is used to record the weights of all the candidate operations and the weights only need to be updated for one epoch when evaluating a solution. Afterward, the validation error $f_{\text{valid\_error}}$ can be calculated based on given validation samples, and the complexity (i.e., ratio of nonzero elements in A) $f_{\text{complexity}}$ can be determined.

B. Experimental Settings

1) Structure of CNN: In the experiments, the proposed MP-MMEA is employed to search for the optimal architectures of a normal cell and a reduction cell, where the CNN consists of six normal cells and two reduction cells stacked alternately. The hypernetwork of each type of cell includes seven nodes, where the first two nodes denote the outputs of the previous two cells and the last node denotes the output of the current cell. The first two nodes do not connect to each other and all the intermediate nodes are connected to the last node, hence there are $C_6^2 - 1 = 14$ connections in each hypernetwork. Eight types of operations are defined in the candidate operation set O, including 3×3 separable convolution, 5×5 separable convolution, 3×3 dilated separable convolution, 5×5 dilated separable convolution, 3×3 max pooling, 3×3 average pooling, identity, and *zero*, hence the number of decision variables is $14 \times 8 \times 2 = 224$ in total. The number of filters in all the operations is set to 8, the stride size is set to 2 for the operations directly connected to the reduction cell's inputs, and the stride size is set to 1 for all the other operations.

2) Dataset: The widely used dataset CIFAR-10 [23] is adopted in the experiments, which contains 50 000 training samples and 10 000 test samples, where each sample is a 32×32 color image belonging to one of ten classes. During the search process, 45 000 training samples are used in the gradient learning to train the weights, and the remaining 5000 training samples are used as validation samples for the architecture search. Before evaluating each solution, the CNN is

TABLE I
TEST ACCURACY OF THE CNNS FOUND BY DIFFERENT ALGORITHMS WITH DIFFERENT ENSEMBLE METHODS ON CIFAR-10. THE BEST RESULT IS HIGHLIGHTED

Ensemble method	Baseline 1 [47]	Baseline 2 [48]	MO_Ring_PSO_SCD	SparseEA	MP-MMEA
Best single CNN	89.47%	93.99%	93.68%	91.10%	93.91%
Unweighted average	90.19%	94.55%	94.34%	92.71%	**95.03**%
Majority vote	N/A	94.33%	94.11%	92.52%	94.79%
Weighted average	90.23%	N/A	94.33%	92.71%	**95.03**%
Rank based weight average	90.32%	N/A	94.50%	92.51%	94.97%

TABLE II
TEST ACCURACY OF THE CNNS FOUND BY SEVERAL NEURAL ARCHITECTURE SEARCH METHODS ON CIFAR-10. THE BEST RESULT IS HIGHLIGHTED

Optimizer	Method	Encoding scheme	Test accuracy (%)	Search cost (GPU days)
Manual design	Wide ResNet [50]	-	94.67	-
	DenseNet [51]	-	95.90	-
Evolutionary algorithm	AmoeabaNet [52]	Blocks	96.63	3150
	E2EPP [53]	Blocks	94.70	8.5
	Hier-EA [54]	Hierarchical graph	96.25	300
	Large-Scale Evolution [55]	Hyperparameters	94.60	2666
	NSGA-Net [56]	Adjacent matrix	97.25	4
Reinforcement learning	BlockQNN-S [57]	Hyperparameters	95.62	96
	MetaQNN [58]	Hyperparameters	93.08	80
	NAS [17]	Hyperparameters	95.53	3150
	NASNet [20]	Blocks	96.27	2000
Gradient descent	DARTS [21]	Hypernetwork	97.00	**1.5**
	NAONet [59]	Blocks	96.82	200
Multi-modal evolutionary algorithm	The proposed MP-MMEA	Hypernetwork	**97.26**	2

trained by stochastic gradient descent for one epoch with a batch size of 96, an initial learning rate of 0.025, a momentum of 0.9, and a weight decay of 3×10^{-4}. After the search process terminates, a larger CNN is constructed based on each solution, where the number of training epochs is set to 600 and the number of filters is increased to ensure that each CNN has approximately 0.2M parameters.

3) Algorithms: MO_Ring_PSO_SCD [4], SparseEA [42], and the proposed MP-MMEA are tested in the experiments, where MO_Ring_PSO_SCD is a state-of-the-art algorithm for MMOPs and SparseEA is a state-of-the-art algorithm for sparse MOPs. The parameter settings are presented in Section V-C. Due to the very expensive function evaluation of the neural architecture search, the maximum number of function evaluations is set to 600 and the population size is set to 20. The experiments are conducted on PyTorch [49] with a single 1080Ti GPU.

C. Results on Neural Architecture Search

Table I lists the test accuracies of the CNNs found by the compared algorithms, where the CNNs are assembled by four ensemble methods, including unweighted average (i.e., average of the output probabilities of all the CNNs), majority vote (i.e., most frequent value of the labels of all the CNNs), weighted average (i.e., the weight of each CNN is determined by its validation error), and rank-based weighted average (i.e., the weight of each CNN is determined by its rank in terms of validation error). Besides, the results of multiple existing CNNs are also included as baselines, where the first one [47] (denoted as Baseline 1) trains five CNNs and uses three ensemble methods and the second one [48] (denoted as Baseline 2) adopts four popular CNNs and uses two ensemble methods. For Baseline 1 considering five simple CNNs as ensemble members, it exhibits a poor performance, having achieved a test accuracy of 90.32% with rank-based weight average. For Baseline 2 considering four popular CNNs, including a VGG [60] and three ResNet [61], it achieves a test accuracy of 94.55%. For the neural architectures obtained by each of MO_Ring_PSO_SCD, SparseEA, and MP-MMEA, five neural architectures with the best validation accuracy on a single class are selected as ensemble members. According to the table, MP-MMEA obtains better test accuracies than the other two MOEAs when using all the ensemble methods, having achieved the best test accuracy of 95.03%. The neural architectures found by MP-MMEA are presented in the supplementary materials I.

It is worth mentioning that in terms of the test accuracy of the best single CNN, Baseline 2 outperforms all the three MOEAs. However, the ensemble of CNNs found by MP-MMEA has a higher test accuracy than the ensemble of the CNNs considered in Baseline 2. Therefore, it is reasonable to regard the neural architecture search as a multimodal optimization problem and to solve it by MP-MMEA, which shows better performance than existing MOEAs.

Furthermore, to compare the performance of MP-MMEA to the state-of-the-art, five CNNs with 36 filters are constructed based on the neural architectures obtained by MP-MMEA, and they are integrated by the ensemble method of weighted average. Table II lists the results of MP-MMEA and some state-of-the-art neural architecture search methods with different optimizers and encoding schemes, where the test accuracy of the CNNs obtained by MP-MMEA is increased to 97.26% due to the use of more filters. It can be found from Table II that MP-MMEA has better performance than all the other methods, and it obtains the neural architectures by searching for

only 2 GPU days. To summarize, the proposed MP-MMEA is an effective tool for the neural architecture search in deep learning.

V. Experiments on Benchmark Problems

A. Existing Benchmark MOPs

To further study the effectiveness of MP-MMEA on large-scale MMOPs with sparse Pareto-optimal solutions, we compare the performance of MP-MMEA to more state-of-the-art MOEAs on benchmark problems. However, none of the existing benchmark MOPs has all the characteristics of large-scale decision variables, multimodality, and sparse Pareto-optimal solutions. Specifically, for the large-scale multiobjective test suite LSMOP [62], it has complex landscapes and variable linkages with good extensibility and generality, but it does not have equivalent or sparse Pareto-optimal solution. For the sparse multiobjective test suite SMOP [42], it has sparse Pareto-optimal solutions but it does not have equivalent Pareto-optimal solutions either. For the multimodal multiobjective test suites, such as Omni-test [8], MMF [4], MMMOP [5], multimodal polygon problems [63], and IDMP [12], they do not have sparse Pareto-optimal solutions. Although MMMOP, multimodal polygon problems, and IDMP are scalable in terms of the number of decision variables, the number of equivalent Pareto-optimal sets of them cannot be arbitrarily specified.

As a consequence, to assess the performance of MOEAs on large-scale MMOPs with sparse Pareto-optimal solutions, this work proposes a new test suite with high flexibility, where the number of objectives, decision variables, equivalent Pareto-optimal sets, and the sparsity of the Pareto-optimal solutions can be set arbitrarily.

B. Proposed Benchmark MMOPs

The benchmark MMOPs in the proposed test suite have the following formulation:

$$\begin{aligned}\text{Minimize}\ & f_1(\mathbf{x}) = h_1(\mathbf{x})\big(1+\min\big(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_{n_p}(\mathbf{x})\big)\big)\\ & f_2(\mathbf{x}) = h_2(\mathbf{x})\big(1+\min\big(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_{n_p}(\mathbf{x})\big)\big)\\ & \ldots\\ & f_M(\mathbf{x}) = h_M(\mathbf{x})\big(1+\min\big(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_{n_p}(\mathbf{x})\big)\big)\\ \text{s.t.}\ & \mathbf{x} = (x_1, x_2, \ldots, x_D)\\ & x_1, x_2, \ldots, x_{M-1} \in [0, 1]\\ & x_M, x_{M+1}, \ldots, x_D \in [-1, 2]\end{aligned} \tag{7}$$

where $f_1, f_2, \ldots, f_M$ denote the objectives, $h_1, h_2, \ldots, h_M$ denote the shape functions related to $x_1, x_2, \ldots, x_{M-1}$, $g_1, g_2, \ldots, g_{n_p} \geq 0$ denote the landscape functions related to $x_M, x_{M+1}, \ldots, x_D$, M is the number of objectives, and D is the number of decision variables. The component $\min(g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_{n_p}(\mathbf{x}))$ introduces n_p equivalent Pareto-optimal sets, since a solution can be Pareto optimal once one of $g_1, g_2, \ldots, g_{n_p}$ is minimized.

To create equivalent Pareto-optimal solutions with different sparse distribution, the landscape functions are defined as

$$g_i(\mathbf{x}) = g^{\text{one}}\big(x_{M+(i-1)S}, x_{M+(i-1)S+1}, \ldots, x_{M+iS-1}\big) + g^{\text{zero}}\big(x_M, \ldots, x_{M+(i-1)S-1}, x_{M+iS}, \ldots, x_D\big) \tag{8}$$

where $g^{\text{one}} \geq 0$ can be minimized when the related decision variables are set to specified nonzero values, $g^{\text{zero}} \geq 0$ can be minimized when all the related decision variables are set to zero, and S controls the sparsity of the Pareto-optimal solutions. As illustrated in Fig. 4, the optimal solutions for minimizing the landscape functions are obviously sparse since most decision variables are 0. More importantly, each pair of landscape functions g^{one} and g^{zero} cannot be minimized simultaneously, which ensures that the equivalent Pareto-optimal solutions have different sparse distribution and the number of equivalent Pareto-optimal sets is always equal to the number of landscape functions n_p.

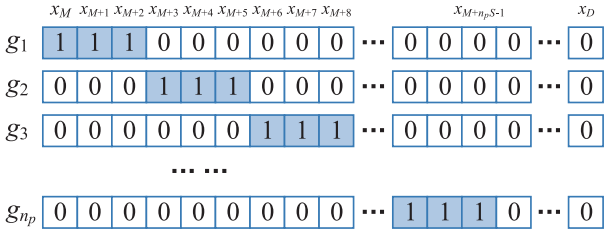

Fig. 4. Values of decision variables for minimizing the landscape functions $g_1, g_2, \ldots, g_{n_p}$, where "1" indicates that the decision variable is a nonzero value and "0" indicates that the decision variable is zero.

In the proposed test suite, we define three types of shape functions for $h_1, h_2, \ldots, h_M$ and five types of basic landscape functions for g^{one} and g^{zero}, and eight benchmark problems SMMOP1–SMMOP8 are constructed based on these functions. The proposed benchmark MMOPs have different characteristics and increasing difficulties, and the number of objectives M, the number of decision variables D, the number of equivalent Pareto-optimal sets n_p, and the sparsity of the Pareto-optimal solutions S all can be specified by users. The detailed definitions of the eight benchmark MMOPs are presented in the supplementary materials II.

C. Experimental Settings

The proposed MP-MMEA is compared to four MOEAs for MMOPs (i.e., MO_Ring_PSO_SCD [4], TriMOEA-TA&R [5], DN-NSGA-II [25], and Omni-optimizer [8]) and one MOEA for sparse MOPs (i.e., SparseEA [42]) on the proposed benchmark MMOPs. As reviewed in Section II-A, MO_Ring_PSO_SCD, DN-NSGA-II, and Omni-optimizer handle multimodality by the crowding distance in both the objective and decision spaces, while TriMOEA-TA&R uses two archives to preserve the population diversity in the decision space and objective space, respectively. Besides, sparseEA is currently the only MOEA for solving sparse MOPs, but it does not consider MMOPs at all. The experiments are conducted on PlatEMO [64].

1) Algorithms: The parameters of all the compared MOEAs are tuned based on the settings suggested in their original papers, where in the supplementary materials III presents a detailed parameter sensitivity analysis. Specifically, the parameters p_{con}, σ_{niche}, and ϵ in TriMOEA-TA&R are set to 0.5, 0.1, and 0.01, respectively, the crowding factor in DN-NSGA-II is

TABLE III
IGDX VALUES OBTAINED BY MO_RING_PSO_SCD, TRIMOEA_TA&R, DN-NSGA-II, OMNI-OPTIMIZER, SPARSEEA, AND THE PROPOSED MP-MMEA ON SMMOP1–SMMOP8. THE BEST RESULT IN EACH ROW IS HIGHLIGHTED

Problem	D	MO_Ring_PSO_SCD	TriMOEA_TA&R	DN-NSGA-II	Omni-optimizer	SparseEA	MP-MMEA
SMMOP1	100	5.2643e+00 (1.16e-01) –	3.5294e+00 (1.25e-02) –	3.4564e+00 (1.24e-02) –	6.8214e+00 (1.75e+00) –	3.4664e+00 (2.34e-02) –	**2.3030e-01 (4.87e-03)**
SMMOP2		6.0918e+00 (1.32e-01) –	4.4744e+00 (5.24e-01) –	5.6843e+00 (7.72e-01) –	1.0151e+01 (9.41e-01) –	3.4373e+00 (4.58e-02) –	**3.4769e-01 (1.49e-02)**
SMMOP3		6.1020e+00 (2.24e-01) –	4.7584e+00 (5.16e-01) –	5.7523e+00 (7.59e-01) –	1.0479e+01 (6.71e-01) –	3.4957e+00 (1.75e-02) –	**4.2094e-01 (4.14e-02)**
SMMOP4		5.4926e+00 (1.03e-01) –	3.5208e+00 (4.55e-03) –	3.4514e+00 (7.56e-03) –	7.7016e+00 (6.25e+01) –	3.3607e+00 (4.02e-02) –	**2.3539e-01 (1.08e-02)**
SMMOP5		5.3584e+00 (1.42e-01) –	3.7541e+00 (1.07e-02) –	3.6344e+00 (4.34e-02) –	8.0875e+00 (2.66e-02) –	3.4395e+00 (3.44e-02) –	**2.3795e-01 (6.69e-03)**
SMMOP6		5.3119e+00 (1.31e-01) –	3.7648e+00 (4.34e-02) –	3.7173e+00 (2.44e-01) –	8.1516e+00 (2.21e-01) –	3.4620e+00 (1.22e-02) –	**3.6510e-01 (4.36e-02)**
SMMOP7		5.5020e+00 (1.14e-01) –	3.5201e+00 (2.82e-03) –	3.5095e+00 (1.18e-03) –	8.2164e+00 (5.26e-01) –	3.5062e+00 (4.25e-03) –	**2.0072e-01 (2.87e-02)**
SMMOP8		5.2054e+00 (6.47e-02) –	3.5438e+00 (1.62e-02) –	3.4692e+00 (1.25e-02) –	8.2403e+00 (3.17e-01) –	3.4673e+00 (2.50e-02) –	**1.8596e-01 (1.86e-02)**
SMMOP1	200	7.8041e+00 (1.05e-01) –	4.9899e+00 (1.17e-02) –	4.9278e+00 (1.22e-02) –	4.9368e+00 (1.09e-02) –	4.9634e+00 (4.01e-02) –	**5.2507e-01 (4.36e-02)**
SMMOP2		8.6250e-01(1.91e-01) –	8.5480e+00 (7.63e-01) –	1.0185e+01 (7.26e-01) –	9.0247e+00 (5.78e-01) –	4.9636e+00 (5.77e-02) –	**6.3939e-01 (4.30e-02)**
SMMOP3		8.7119e+00 (1.53e-01) –	8.7790e+00 (7.03e-01) –	1.1032+01 (9.53e-01) –	9.5563e+00 (7.70e-01) –	5.0071e+00 (7.31e-02) –	**9.8199e-01 (7.05e-02)**
SMMOP4		8.0210e+00 (1.42e-01) –	4.9122e+00 (1.83e-02) –	4.9203e+00 (1.82e-02) –	4.9208e+00 (2.32e-02) –	4.9277e+00 (3.29e-02) –	**6.3424e-01 (2.67e-02)**
SMMOP5		7.9317e+00 (5.68e-01) –	5.3896e+00 (4.25e-02) –	5.3525e+00 (3.62e-01) –	5.2232e+00 (4.16e-02) –	4.9319e+00 (3.71e-02) –	**6.3424e-01 (2.67e-02)**
SMMOP6		7.9337e+00 (1.70e-01) –	5.4240e+00 (4.10e-02) –	5.4170e+00 (4.38e-02) –	5.2790e+00 (4.14e-02) –	4.9409e+00 (5.96e-02) –	**1.4178e+00 (8.01e-02)**
SMMOP7		8.1112e+00 (1.85e-01) –	4.9776e+00 (3.36e-03) –	4.9686e+00 (3.34e-03) –	4.9674e+00 (1.41e-03) –	4.9659e+00 (1.92e-02) –	**9.0458e-01 (8.91e-02)**
SMMOP8		7.8109e+00 (1.02e-01) –	4.9975e+00 (1.36e-02) –	4.9329e+00 (1.39e-02) –	4.9450e+00 (7.45e-03) –	4.9630e+00 (3.03e-02) –	**8.4197e-01 (8.72e-02)**
SMMOP1	500	1.2777e+01 (1.13e-01) –	7.9124e+00 (1.45e-02) –	7.8713e+00 (4.17e-02) –	7.8618e+00 (1.54e-02) –	8.0140e+00 (6.51e-02) –	**2.3287e+00 (7.88e-02)**
SMMOP2		1.4001e+01 (3.00e-01) –	1.9323e+01 (8.65e-01) –	2.2172e+01 (7.97e-01) –	1.9302e+01 (8.22e-00) –	8.0561e+00 (7.94e-02) –	**2.3676e+00 (2.28e-02)**
SMMOP3		1.3902e+01 (3.83e-01) –	2.0274e+01 (6.42e-01) –	2.2401e+01 (6.97e-01) –	1.9812e+01 (7.14e-01) –	8.1833e+00 (1.11e-01) –	**2.8610e+00 (5.67e-02)**
SMMOP4		1.3113e+01 (1.46e-01) –	7.9574e+00 (3.32e-02) –	7.9231e+00 (5.48e-02) –	7.8617e+00 (1.94e-02) –	7.9551e+00 (7.84e-02) –	**2.5978e+00 (7.01e-02)**
SMMOP5		1.2966e+01 (3.35e-01) –	8.9936e+00 (7.79e-02) –	9.2158e+00 (1.03e-01) –	8.6813e+00 (8.64e-02) –	7.9856e+00 (8.34e-02) –	**2.7404e+00 (1.28e-01)**
SMMOP6		1.2776e+01 (1.44e-01) –	9.0600e+00 (7.13e-02) –	9.3688e+00 (1.06e-01) –	8.7665e+00 (5.83e-02) –	8.0951e+00 (8.03e-02) –	**4.6676e+00 (1.16e-01))**
SMMOP7		1.3023e+01 (1.60e-01) –	7.8865e+00 (4.41e-03) –	7.8980e+00 (5.57e-02) –	7.8627e+00 (4.36e-03) –	8.1515e+00 (1.14e-01) –	**3.2853e+00 (3.70e-01)**
SMMOP8		1.2743e+01 (2.13e-01) –	7.9439e+00 (2.75e-02) –	7.8779e+00 (3.27e-02) –	7.8608e+00 (1.16e-02) –	8.0392e+00 (8.16e-02) –	**2.6931e+00 (1.51e-01)**
$+/-/\approx$		0/24/0	0/24/0	0/24/0	0/24/0	0/24/0	

set to half the population size, and the proposed MP-MMEA performs the merge-and-divide operation each 50 generations. For fairness, the maximum number of function evaluations for all the MOEAs is set to 250 000, 400 000, and 500 000 for problems with 100, 200, and 500 decision variables; to obtain a sufficient number of solutions for each equivalent Pareto-optimal set [4], [5], the total population size is set to 500 for all the MOEAs.

2) Operators: For MO_Ring_PSO_SCD, it generates offspring solutions by particle swarm optimization, where the parameters C_1, C_2, and W are set to 0.5, 0.5, and 0.1. For the other MOEAs, offspring solutions are generated by simulated binary crossover [43] and polynomial mutation [44], where the crossover probability is set to 1, the mutation probability is set to $1/D$, the distribution index is set to 20 as suggested in [5] and [12].

3) Benchmark Problems: For the proposed benchmark problems SMMOP1–SMMOP8, the number of objectives M is set to 2 so that the problems are MOPs, the number of decision variables D is set to 100, 200, and 500 so that the problems are large-scale optimization problems, the number of equivalent Pareto-optimal sets n_p is set to 4 so that the problems are MMOPs, and the sparsity parameter S is set to $\lceil 0.1(D-M+1)\rceil$ so that the Pareto-optimal solutions are sparse.

4) Performance Metrics: The performance metric IGDX [65] is adopted to evaluate the performance of each MOEA, which is widely used in the comparisons between the MOEAs for MMOPs. Let P^* denote a set of uniformly distributed points in the decision space and P denote the decision variables of an obtained population, the IGDX value is calculated by

$$\mathrm{IGDX}(P, P^*) = \frac{\sum_{\mathbf{v}\in P^*} \mathrm{dis}(\mathbf{v}, P)}{|P^*|} \tag{9}$$

where $\mathrm{dis}(v, P)$ is the minimum Euclidean distance between point $\mathbf{v}$ and solutions in P. To obtain the reference point set P^*, 1000 reference points are uniformly sampled in each equivalent Pareto-optimal set in the decision space. Since the number of objectives is two, the reference points can be easily sampled by varying the first variable used in the shape functions and fixing the other variables used in the landscape functions to their optimal values. For each MOEA on each benchmark problem, 30 independent runs are performed and the mean and standard deviation of the IGDX values are recorded. In addition, the Friedman test with Bonferroni correction at a significance level of 0.05 [66] is adopted to perform statistical analysis, where the symbols "+," "−," and "≈" indicate that the result obtained by an MOEA is significantly better, significantly worse, and statistically similar to that obtained by MP-MMEA, respectively.

D. Results on Benchmark Problems

Table III lists the IGDX values obtained by the six compared MOEAs on the eight benchmark MMOPs. It is

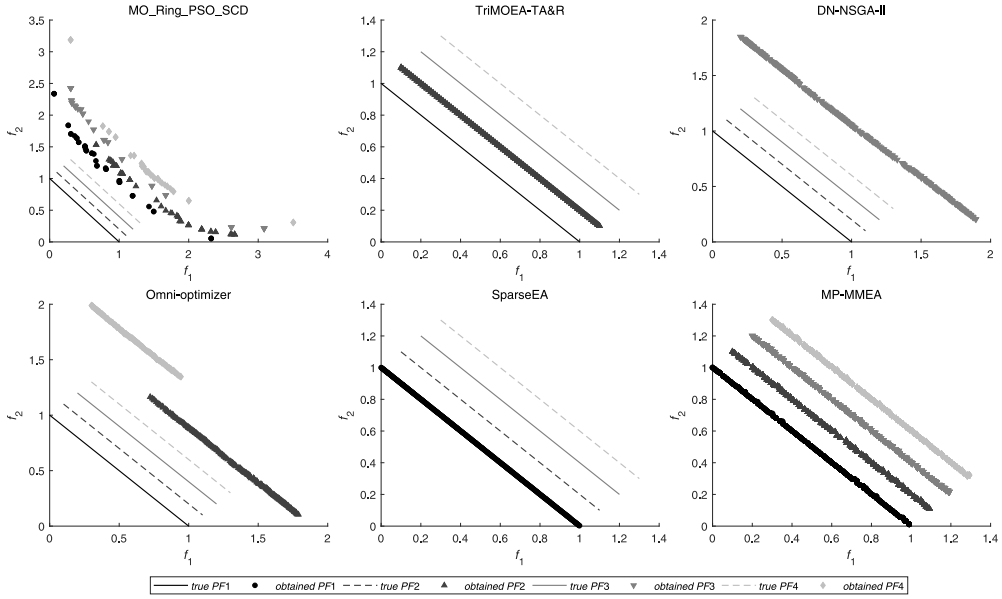

Fig. 5. Results with the median IGDX values obtained by MO_Ring_PSO_SCD, TriMOEA_TA&R, DN-NSGA-II, Omni-optimizer, SparseEA, and the proposed MP-MMEA on SMMOP1 in the objective space. Note that the four Pareto fronts are translated for different distances for better observation, which actually have the same objective values.

obvious that the proposed MP-MMEA always obtains the best result and significantly outperforms the other MOEAs on all the test instances, which demonstrates the effectiveness of MP-MMEA in solving large-scale MMOPs with sparse Pareto-optimal solutions. Besides, MO_Ring_PSO_SCD, DN-NSGA-II, and Omni-optimizer exhibit worse performance than TriMOEA_TA&R and SparseEA, which indicates that the explicit diversity preservation in both the objective and decision spaces is not effective for solving large-scale MMOPs. Besides, in the supplementary materials IV presents the IGD [67] values obtained by the compared MOEAs, where the solutions obtained by MP-MMEA are also better than those obtained by the other MOEAs in the objective space. As a consequence, MP-MMEA is effective for solving large-scale MMOPs with sparse Pareto-optimal solutions.

Fig. 5 depicts the populations with the median IGDX values obtained by the six compared MOEAs on SMMOP1 with 100 decision variables in the objective space, where SMMOP1 has a linear Pareto front and a unimodal landscape. Since the problem has four equivalent Pareto-optimal sets corresponding to the same Pareto front in the objective space, the Pareto fronts of the four equivalent Pareto-optimal sets are translated for different distances for better observation. For MO_Ring_PSO_SCD, DN-NSGA-II, and Omni-optimizer, they converge to none of the four Pareto fronts, which further indicates that the crowding distance in both the objective and decision spaces prevents the population from approximating any Pareto-optimal set in a huge decision space. This phenomenon is similar to NSGA-II on many-objective optimization problems, where the performance of NSGA-II can be improved if the crowding distance is eliminated [68]. For TriMOEA_TA&R and SparseEA, both of them can converge to one of the four Pareto fronts, which shows the effectiveness of the two-archive strategy in TriMOEA_TA&R and the tailored genetic operators in SparseEA. However, SparseEA does not consider MMOPs at all, and the strategy for handling multimodality in TriMOEA_TA&R seems ineffective for large-scale MMOPs. On the contrary, the proposed MP-MMEA drives multiple populations to approximate different equivalent Pareto-optimal sets via the guiding vectors, which can efficiently converge to all the four Pareto fronts. Moreover, Fig. 6 plots the populations with the median IGDX values obtained by the six compared MOEAs on SMMOP8 with 100 decision variables, where SMMOP8 has a concave Pareto front and a highly multimodal landscape with many local optimums. It can be found that the proposed MP-MMEA can still converge to all the four Pareto fronts, while TriMOEA_TA&R and SparseEA can converge to only one Pareto front and MO_Ring_PSO_SCD, DN-NSGA-II, and Omni-optimizer cannot converge to any Pareto front.

Finally, to verify the effectiveness of the merge-and-divide operation in the proposed MP-MMEA, Fig. 7 shows the variation of the number of subpopulations during the search process of MP-MMEA on SMMOP1 and SMMOP8 with 100 decision variables, where the number of equivalent Pareto-optimal sets n_p is set to 4, 6, and 8. It can be found that the number of subpopulations approximately equals the actual number of n_p on all the test instances. Hence, the effectiveness of the merge-and-divide operation in controlling the number of subpopulations can be confirmed. Furthermore, in the supplementary materials V verifies the effectiveness of the other components in MP-MMEA.

VI. Conclusion

Although large-scale MMOPs widely exist in real-world applications, existing work mainly focuses on solving the small-scale MMOPs. To fill the gap, this article has proposed a multipopulation multimodal EA for solving large-scale MMOPs with sparse Pareto-optimal solutions. To guarantee both the convergence and divergence of solutions in the

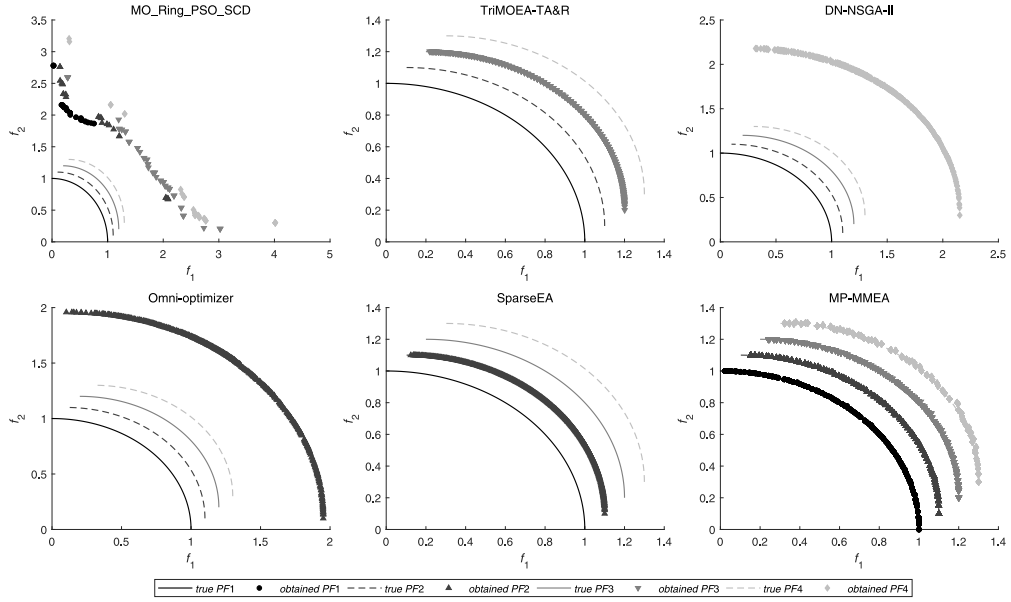

Fig. 6. Results with the median IGDX values obtained by MO_Ring_PSO_SCD, TriMOEA_TA&R, DN-NSGA-II, Omni-optimizer, SparseEA, and the proposed MP-MMEA on SMMOP8 in the objective space. Note that the four Pareto fronts are translated for different distances for better observation, which actually have the same objective values.

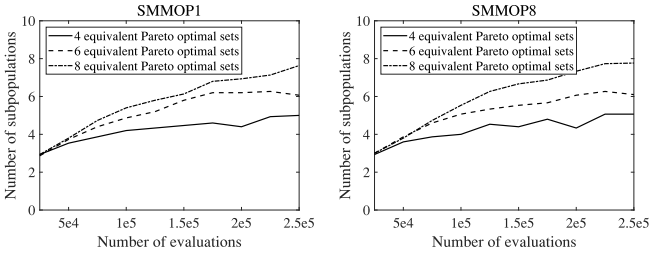

Fig. 7. Variation of the number of subpopulations during the search process of MP-MMEA on SMMOP1 and SMMOP8, where the number of equivalent Pareto-optimal sets is set to 4, 6, and 8.

decision space, the proposed MOEA evolves multiple subpopulations with the assistance of a set of guiding vectors. The guiding vectors define the search directions of subpopulations, which not only accelerates the convergence of subpopulations toward sparse Pareto-optimal solutions, but also differentiates the search directions of subpopulations from each other to find the equivalent Pareto-optimal solutions.

The proposed MOEA has been employed to search for the neural architectures of deep CNNs, which has exhibited better performance than some existing MOEAs and neural architecture search methods for an image classification task. Furthermore, a new test suite has been proposed for assessing the performance of the proposed MOEA, which contains eight large-scale MMOPs with sparse Pareto-optimal solutions. According to the experiments with up to 500 decision variables, it has been shown that the proposed MOEA is superior over state-of-the-art MOEAs for solving the benchmark MMOPs.

This article has improved the performance of MOEAs on large-scale MMOPs and verified the usefulness of the multimodal multiobjective optimization on the neural architecture search. In the future, it is reasonable to apply the proposed MOEA to other important applications [13], [16] and to enhance its performance on MMOPs with more than two objectives or complex variable linkages. Besides, it is interesting to modify the proposed MOEA to find local Pareto-optimal solutions and nonsparse Pareto-optimal solutions for MMOPs.

REFERENCES

[1] J. Michalek, R. Choudhary, and P. Papalambros, "Architectural layout design optimization," *Eng. Optim.*, vol. 34, no. 5, pp. 461–484, 2002.

[2] F. Kudo, T. Yoshikawa, and T. Furuhashi, "A study on analysis of design variables in Pareto solutions for conceptual design optimization problem of hybrid rocket engine," in *Proc. IEEE Congr. Evol. Comput.*, 2011, pp. 2558–2562.

[3] M. Sebag, N. Tarrisson, O. Teytaud, J. Lefèvre, and S. Baillet, "A multi-objective multi-modal optimization approach for mining stable spatio-temporal patterns," in *Proc. Int. Joint Conf. Artif. Intell. Org.*, 2005, pp. 859–864.

[4] C. Yue, B. Qu, and J. J. Liang, "A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 5, pp. 805–817, Oct. 2018.

[5] Y. Liu, G. G. Yen, and D. Gong, "A multimodal multiobjective evolutionary algorithm using two-archive and recombination strategies," *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 660–674, Aug. 2019.

[6] O. M. Shir, M. Preuss, B. Naujoks, and M. T. M. Emmerich, "Enhancing decision space diversity in evolutionary multiobjective algorithms," in *Proc. Int. Conf. Evol. Multi Crit. Optim.*, 2009, pp. 95–109.

[7] O. Kramer and H. Danielsiek, "DBSCAN-based multi-objective niching to approximate equivalent Pareto-subsets," in *Proc. Genet. Evol. Comput. Conf.*, 2010, pp. 503–510.

[8] K. Deb and S. Tiwari, "OMNI-optimizer: A procedure for single and multi-objective optimization," in *Proc. Int. Conf. Evol. Multi Crit. Optim.*, 2005, pp. 47–61.

[9] M. Kim, T. Hiroyasu, M. Miki, and S. Watanabe, "SPEA2+: Improving the performance of the strength Pareto evolutionary algorithm 2," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2004, pp. 742–751.

[10] C. Hu and H. Ishibuchi, "Incorporation of a decision space diversity maintenance mechanism into MOEA/D for multi-modal multi-objective optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2018, pp. 1898–1901.

[11] R. Tanabe and H. Ishibuchi, "A decomposition-based evolutionary algorithm for multi-modal multi-objective optimization," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2018, pp. 249–261.

[12] Y. Liu, H. Ishibuchi, G. G. Yen, Y. Nojima, and N. Masuyama, "Handling imbalance between convergence and diversity in the decision space in evolutionary multi-modal multi-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 551–565, Jun. 2020.

[13] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev*, vol. 38, no. 3, pp. 397–415, May 2008.

[14] Y. Tian, X. Zheng, X. Zhang, and Y. Jin, "Efficient large-scale multi-objective optimization based on a competitive swarm optimizer," *IEEE Trans. Cybern.*, vol. 38, no. 3, pp. 397–415, May 2008.

[15] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *Proc. Int. Conf. Database Theory*, 2001, pp. 420–434.

[16] C. Yue, J. J. Liang, B. Qu, K. Yu, and H. Song, "Multimodal multiobjective optimization in feature selection," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 302–309.

[17] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–16.

[18] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "SMASH: One-shot model architecture search through hypernetworks," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[19] Y. Gong, J. Zhang, and Y. Zhou, "Learning multimodal parameters: A bare-bones niching differential evolution approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2944–2959, Jul. 2018.

[20] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8697–8710.

[21] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *Proc. 7th Int. Conf. Learn. Represent.*, 2019, p. 6.

[22] X. Li, M. G. Epitropakis, K. Deb, and A. Engelbrecht, "Seeking multiple solutions: An updated survey on niching methods and their applications," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 518–538, Aug. 2017.

[23] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.

[24] R. Tanabe and H. Ishibuchi, "A review of evolutionary multi-modal multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 193–200, Feb. 2020.

[25] J. J. Liang, C. T. Yue, and B. Y. Qu, "Multimodal multi-objective optimization: A preliminary study," in *Proc. IEEE Congr. Evol. Comput.*, 2016, pp. 2454–2461.

[26] Y. Liu, H. Ishibuchi, Y. Nojima, N. Masuyama, and K. Shang, "A double-niched evolutionary algorithm and its behavior on polygon-based problems," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2018, pp. 262–273.

[27] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, Feb. 2010.

[28] W. Zhang, G. Li, W. Zhang, J. Liang, and G. G. Yen, "A cluster based PSO with leader updating mechanism and ring-topology for multimodal multi-objective optimization," *Swarm Evol. Comput.*, vol. 50, Nov. 2019, Art. no. 100569.

[29] R. Tanabe and H. Ishibuchi, "A framework to handle multimodal multiobjective optimization in decomposition-based evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 24, no. 4, pp. 720–734, Aug. 2020.

[30] Q. Lin, W. Lin, Z. Zhu, M. Gong, J. Li, and C. A. Coello Coello, "Multimodal multi-objective evolutionary optimization with dual clustering in decision and objective spaces," *IEEE Trans. Evol. Comput.*, early access, Jul. 13, 2020, doi: 10.1109/TEVC.2020.3008822.

[31] S. Kukkonen and K. Deb, "Improved pruning of non-dominated solutions based on crowding distance for bi-objective optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, 2006, pp. 1179–1186.

[32] R. Morgan and M. Gallagher, "Sampling techniques and distance metrics in high dimensional continuous landscape analysis: Limitations and improvements," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 456–461, Jun. 2014.

[33] E. M. Zechman, M. H. Giacomoni, and M. E. Shafiee, "An evolutionary algorithm approach to generate distinct sets of non-dominated solutions for wicked problems," *Eng. Appl. Artif. Intell.*, vol. 26, nos. 5–6, pp. 1442–1457, 2013.

[34] Y. Peng and H. Ishibuchi, "A decomposition based large-scale multi-modal multi-objective optimization algorithm," in *Proc. IEEE Congr. Evol. Comput.*, 2020, pp. 1–8.

[35] H. Qian and Y. Yu, "Solving high-dimensional multi-objective optimization problems with low effective dimensions," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 875–881.

[36] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "A new two-stage evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 748–761, Oct. 2019.

[37] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multi-objective optimization based on problem transformation," *IEEE Trans. Evol. Comput.*, vol. 22, no. 2, pp. 260–275, Apr. 2018.

[38] C. He *et al.*, "Accelerating large-scale multi-objective optimization via problem reformulation," *IEEE Trans. Evol. Comput.*, vol. 23, no. 6, pp. 949–961, Dec. 2019.

[39] X. Ma *et al.*, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 275–298, Apr. 2016.

[40] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 97–112, Feb. 2018.

[41] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "An efficient approach to non-dominated sorting for evolutionary multi-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 201–213, Apr. 2015.

[42] Y. Tian, X. Zhang, C. Wang, and Y. Jin, "An evolutionary algorithm for large-scale sparse multi-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 380–393, Apr. 2020.

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

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

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

[46] Y. Ge *et al.*, "Distributed differential evolution based on adaptive mergence and split for large-scale optimization," *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 2166–2180, Jul. 2018.

[47] X. Frazao and L. A. Alexandre, "Weighted convolutional neural network ensemble," in *Proc. Iberoamerican Congr. Pattern Recognit.*, 2014, pp. 674–681.

[48] C. Ju, A. Bibaut, and M. van der Laan, "The relative performance of ensemble methods with deep convolutional neural networks for image classification," *J. Appl. Stat.*, vol. 45, no. 15, pp. 2800–2818, 2018.

[49] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. NIPS Workshop*, 2017, p. 57.

[50] S. Zagoruyko and N. Komodakis, "Wide residual networks," Rep., 2016. [Online]. Available: arXiv: 1605.07146

[51] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.

[52] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 4780–4789.

[53] Y. Sun, H. Wang, B. Xue, Y. Jin, G. G. Yen, and M. Zhang, "Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 350–364, Apr. 2020.

[54] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018.

[55] E. Real *et al.*, "Large-scale evolution of image classifiers," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 6, 2017, pp. 4429–4446.

[56] Z. Lu *et al.*, "NSGA-Net: Neural architecture search using multi-objective genetic algorithm," in *Proc. Genet. Evol. Comput. Conf.*, 2019, pp. 419–427.

[57] Z. Zhong, J. Yan, W. Wu, J. Shao, and C. L. Liu, "Practical block-wise neural network architecture generation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2423–2432.

[58] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017.

[59] R. Luo, F. Tian, T. Qin, E. Chen, and T. Y. Liu, "Neural architecture optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7816–7827.

[60] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: arXiv:1409.1556.

[61] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[62] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4108–4121, Dec. 2017.
[63] H. Ishibuchi, Y. Peng, and K. Shang, "A scalable multimodal multiobjective test problem," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 310–317.
[64] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.
[65] A. Zhou, Q. Zhang, and Y. Jin, "Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1167–1189, Oct. 2009.
[66] J. Derrac, S. Garcia, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, 2011.
[67] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.
[68] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 348–365, Jun. 2014.

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

He is currently an Associate Professor with the Institutes of Physical Science and Information Technology, Anhui University and also a Postdoctoral Research Fellow with the Department of Computer Science, City University of Hong Kong, Hong Kong. His current research interests include multiobjective optimization methods and their applications.

Dr. Tian is a recipient of the 2018 and 2021 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award and the 2020 *IEEE Computational Intelligence Magazine* Outstanding Paper Award.

Ruchen Liu received the B.Sc. degree from Anhui University, Hefei, China, in 2018, where he is currently pursuing the M.Sc. degree with the School of Computer Science and Technology.

His current research interests include multimodal multiobjective optimization and deep learning.

Xingyi Zhang (Senior Member, IEEE) received the B.Sc. degree from Fuyang Normal College, Fuyang, China, in 2003, and the M.Sc. and Ph.D. degrees from the 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, multiobjective optimization, and membrane computing.

Prof. Zhang is a recipient of the 2018 and 2021 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award and the 2020 *IEEE Computational Intelligence Magazine* Outstanding Paper Award.

Haiping Ma received the B.E. degree from Anhui University, Hefei, China, in 2008, and the Ph.D. degree from the University of Science and Technology of China, Hefei, in 2013.

She is currently a Lecturer with the Institutes of Physical Science and Information Technology, Anhui University. Her current research interests include data mining and multiobjective optimization methods and their applications.

Kay Chen Tan (Fellow, IEEE) received the B.Eng. degree (First Class Hons.) in electronics and electrical engineering and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is a Full Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. He has published over 200 refereed articles and six books.

Prof. Tan is the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. He was the Editor-in-Chief of *IEEE Computational Intelligence Magazine* from 2010 to 2013. He currently serves as the Editorial Board Member of over ten journals. He is currently an IEEE Distinguished Lecturer Program Speaker and the Chief Co-Editor of Springer Book Series on *Machine Learning: Foundations, Methodologies, and Applications*.

Yaochu Jin (Fellow, IEEE) 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 of Computational Intelligence with the 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. He has (co)-authored over 300 peer-reviewed journal and conference papers and been granted eight patents on evolutionary optimization.

Dr. Jin is a recipient of the 2014, 2016, and 2020 *IEEE Computational Intelligence Magazine* Outstanding Paper Award, the 2018 and 2021 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 has been named a Highly Cited Researcher for 2019 and 2020 by the Web of Science group. He is the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS and *Complex and Intelligent Systems*. He is also an Associate Editor or Editorial Board Member of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, *Evolutionary Computation*, and *Soft Computing*. He was an IEEE Distinguished Lecturer from 2013 to 2015 and from 2017 to 2019.