Multiple Tasks for Multiple Objectives: A New Multiobjective Optimization Method via Multitask Optimization

Jian-Yu Li, *Member*, *IEEE*, Zhi-Hui Zhan, *Senior Member*, *IEEE*, Yun Li, *Fellow*, *IEEE*, Jun Zhang, *Fellow*, *IEEE*

Abstract—Handling conflicting objectives and finding multiple Pareto optimal solutions are two challenging issues in solving multiobjective optimization problems (MOPs). Inspired by the efficiency of multitask optimization (MTO) in finding multiple optimal solutions of multitask optimization problem (MTOP), we propose to treat MOP as a MTOP and solve it by using MTO. By transforming the MOP into a MTOP, not only that the difficulty in handling conflicting objectives can be avoided, but also that MTO can help efficiently find well-distributed multiple optimal solutions for MOP. With the above idea, this paper proposes a new multiobjective optimization method via MTO, with the following three contributions. Firstly, a theorem is proposed to theoretically show the relationship between MOP and MTOP and how MOP can be transformed into a MTOP. Secondly, based on the theoretical analysis, a multiple tasks for multiple objectives (MTMO) framework is proposed for solving MOP efficiently. Thirdly, a MTMO-based evolutionary algorithm is developed to solve MOP, together with two novel strategies. One is a target point estimation strategy for transforming the MOP into a MTOP automatically and accurately. The other is an archive-based implicit knowledge transfer strategy for efficiently transferring knowledge across multiple tasks to enhance the optimization results of multiple tasks together. The superiority of the proposed algorithm is validated in extensive experiments on 15 MOPs with objective numbers varying from 3 to 20 and with six state-of-the-art algorithms as competitors. Therefore, solving MOP and even many-objective optimization problem via MTO is a new, promising, and efficient method.

***Index Terms*—Multiobjective optimization problem, multitask optimization problem, evolutionary computation, multiple tasks for multiple objectives, knowledge transfer, transforming**

Manuscript received XXXX; revised XXXX; accepted XXXX. This work was supported in part by the National Natural Science Foundations of China (NSFC) under Grants 62176094 and 92270105, in part by the Key Program of the Natural Science Foundation of Guangdong Province under Grant 2021B1515120078, in part by the Ministry of Science and Technology of China under Grant G2022032012L, and in part by National Research Foundation of Korea under Grant NRF2022H1D3A2A01093478 and Grant NRF-2020R1C1C1013806. *(Corresponding authors: Zhi-Hui Zhan, Jun Zhang.)*

J.-Y. Li is with the College of Artificial Intelligence, Nankai University, Tianjin 300350, China, and also with the Hanyang University, ERICA, South Korea.

Z.-H. Zhan are with the School of Computer Science and Engineering, South China University of Technology, 51006 Guangzhou, China. (e-mail: zhanapollo@163.com).

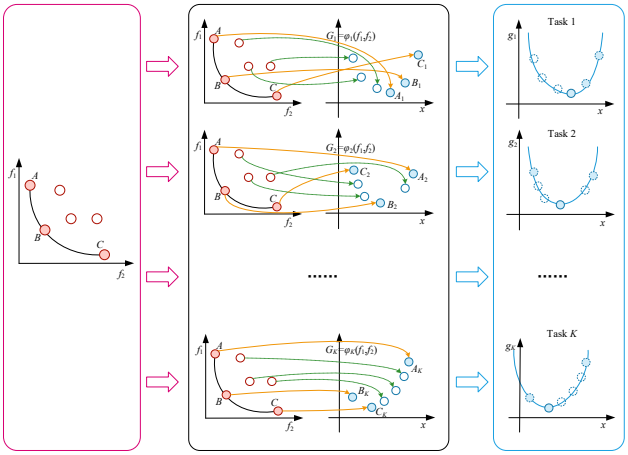
Y. Li is with the Industrial Artificial Intelligence Centre, Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China, and also with the i4AI Ltd, London WCIN3AX, United Kingdom.

J. Zhang is with the Hanyang University, ERICA, South Korea.

I. Introduction

Multiobjective optimization problem (MOP) and multitask optimization problem (MTOP) are both foundational and significant optimization problems that widely exist in various real-world applications, which have attracted great attention in the evolutionary computation (EC) community [1]-[4]. Generally speaking, the MOP refers to the optimization problem with more than one objective to be optimized. Moreover, in the MOP, no solution can be better than all other solutions on all the objectives at the same time. Therefore, the MOP requires the EC algorithm to find a set of Pareto optimal solutions, i.e., the solutions that are better than other solutions on at least one objective. Differently, the MTOP refers to the optimization problem with multiple optimization tasks to be optimized together. That is, the MTOP requires the EC algorithm to find a set of solutions that consists of the optimal solutions that are corresponding to all the tasks. Note that the MOP with a large number of objectives (e.g., larger than three) and the MTOP with a large number of tasks are often called many-objective optimization problem (MaOP) and many-task optimization problem (MaTOP), respectively, which emphasizes that the complexity and difficulties of the problems are mainly due to the large number of objectives or tasks. As MaOP and MaTOP are kinds of more complex MOP and MTOP, respectively, the MaOP and MaTOP are also referred as MOP and MTOP in the following contents for clarity, respectively.

As both MOP and MTOP are very significant in real-world applications, EC algorithms for these two types of problems have attracted increasing attention and have led to two fast-growing research branches. So far, the researches on MOP and MTOP are still relatively independent and have different focuses. In general, the research on MOP focuses on how to generate, select, and evolve a set of solutions to approach the Pareto front, i.e., finding the optimal non-dominated solutions. Existing work for MOP can be roughly classified into six categories, which are dominance-based methods [5]-[7], decomposition-based methods [8][9], indicator-based methods [10][11], multiple populations coevolution-based methods [12][13], preference information-based methods [14][15], and other methods [16][17]. While for MTOP, the researchers focus on how to obtain and transfer knowledge among multiple tasks during the evolutionary search for enhancing the optimization of all tasks. Existing approaches for MTOP can be mainly categorized into two categories, i.e., the implicit knowledge



(a) Solutions in a MOP (b) Transforming the MOP into K optimization tasks with different fitness functions respectively (c) Solutions of K tasks in the MTOP

Fig. 1. Illustration of relationship of multiple global optimal solutions in minimization MOP and MTOP, where A, B, and C are the Pareto optimal solutions of the MOP.

transfer-based approaches [18] and the explicit knowledge transfer-based approaches [19][20]. Note that in recent years, the multiobjective MTOP has also attracted the research interest of both the MOP and MTOP research communities [21]-[24], because this kind of problem generally contains both the multiobjective and multitask characteristics.

Interestingly, both MOP and MTOP are in fact multiple solutions optimization problems. That is, both MOP and MTOP have multiple solutions that are equally important, and all of these solutions can be regarded as the optimal solutions to the problem. To better illustrate this, Fig. 1 gives an example to show the relationship of the multiple global optimal solutions in MOP and MTOP. Fig. 1(a) shows some solutions for a minimization MOP, where the solutions A, B, and C are the Pareto optimal solutions, because they are not dominated by any other solutions. Fig. 1(c) shows the K optimal solutions of K different minimization tasks in a MTOP, respectively, where the fitness function of the kth task is denoted as g_k. Moreover, between Fig. 1(a) and Fig. 1(c), Fig. 1(b) illustrates that a MOP can be transformed into a MTOP with multiple single-objective optimization tasks, so that each Pareto optimal solution in the MOP can be regarded as the global optimal solution of a corresponding task in the transformed MTOP. To be specific, in Fig. 1(b), given the two objective functions of the MOP as f_1 and f_2, each solution in the MOP has two objective values. Then, the objective value of these solutions can: first, be mapped by the $G_1=\varphi_1(f_1, f_2)$, so that the Pareto solution A is the optimal solution of G_1, where G_1 can be regarded as the fitness function of the first task in the MTOP (i.e., g_1); second, be mapped by the $G_2=\varphi_2(f_1, f_2)$, so that the Pareto solution B is the optimal solution of G_2, where G_2 can be regarded as the fitness function of the second task in the MTOP (i.e., g_2); and so on. By doing so, finding a set of Pareto optimal solutions for the MOP is equivalent to finding a set of optimal solutions that correspond to different tasks in the MTOP.

In this sense, it is interesting that the MOP and MTOP share something in common and have meaningful relationships. That is, as both the MOP and MTOP aim to find multiple optimal solutions, the EC algorithms and knowledge transfer methods for MTOP may be also useful for finding the Pareto optimal solutions for MOP. This interesting feature has attracted some attention in the EC community and has been discussed in some recent work [21], [22]. However, we could only find a few studies that attempt to connect the MOP with the MTOP. For example, when solving a complex MOP, Gupta *et al.* [23] proposed to add a relatively simpler MOP transformed from the original MOP, so as to form two tasks (both are MOPs) and use the MTOP algorithms to solve the MTOP with two MOP tasks. When solving a high-dimensional MOP, Feng *et al.* [25] proposed to transform the original MOP into several simplified smaller-scale MOPs, so that the additional simplified MOPs can assist the optimization of the original large-scale MOP. Qiao *et al.* [26] created an additional MOP with relaxed constraints, so as to help solve the original constrained MOP. However, these approaches mainly extend the MOP to a multiobjective MTOP by simplifying the original complex MOP into multiple simpler MOP tasks, which is different from the focus of this paper, i.e., solving the MOP more effectively by treating the whole MOP as a MTOP.

Although some works mentioned above have attempted to connect the MOP with MTOP, to the best of our knowledge, there is still no work on the research topic to solve the MOP by treating the MOP itself as a MTOP. Treating the MOP (with multiple conflicting objectives) as a MTOP (with multiple relevant tasks) has two advantages. First, the difficulty in handling conflicting objectives can be avoided because the MTOP does not have conflicting objectives. Second, as efficient tools for finding multiple optimal solutions simultaneously, multitask optimization (MTO) methods, such as knowledge transfer methods across tasks, can be used to find multiple optimal solutions for MOP more efficiently. Therefore, this is a new emerging research topic in the EC community. To fill this gap, this paper makes the first attempt to transform the MOP as a MTOP, so as to solve the MOP more efficiently. In this paper, we prove a theorem showing the relationship between MOP and MTOP and how a MOP can be transformed into a MTOP theoretically. Based on the theoretical analysis, a multiple tasks for multiple objectives (MTMO) framework is proposed to solve MOP via MTO. Moreover, this paper develops a MTMO-based evolutionary algorithm (MTMOEA) with two novel strategies to solve the MOP more efficiently. One is a target point estimation (TPE) strategy to help transform the MOP into a MTOP more accurately and automatically. The other is an archive-based implicit knowledge transfer (AIKT) strategy to transfer knowledge across different tasks. Therefore, this paper is the first to propose a new research topic with a novel idea of solving MOP by a new method via MTO. The main contributions of this paper are summarized as the following three aspects:

Firstly, in solid theory, this paper is the first to propose a theorem about optimization equivalence between MOP and MTOP and how a MOP can be transformed into a MTOP mathematically, based on which MOP can be solved via MTO.

Secondly, in generic framework, this paper proposes the

MTMO framework that is suitable for solving MOP via MTO, which is also generic that can be cooperated with different kinds of EC algorithms.

Thirdly, in efficient algorithm design, this paper proposes the efficient MTMOEA based on the MTMO framework to solve the MOP efficiently.

To investigate the proposed MTMOEA, extensive experiments are conducted on all the 15 problems in the MaF benchmark suite [27], i.e., MaF1 to MaF15, which have been used as the benchmark problems for IEEE CEC competition in recent years. For more comprehensive studies, all the 15 MaF problems are with the objective number as 3, 5, 10, and 20, i.e., 15×4=60 problems in total, so as to fully investigate the proposed MTMOEA.

The rest contents are organized as follows: Section II briefly introduces the background of MOP and MTOP and their related work, while Section III describes the relationship between MOP and MTOP, and details the proposed MTMOEA. Section VI provides the experimental settings, metrics, comparisons, and analyses. In the end, Section V is the conclusion.

II. Background and Related Work

A. Multiobjective Optimization Problem

Given a search space Ω, a minimization MOP can be formulated as follows:

$$\text{Minimize } F(x) = [f_1(x), f_2(x), ..., f_M(x)] \quad (1)$$

where F consists of M objective functions $f_1, f_2, \ldots$, and f_M, and maps decision variables $x \in \Omega$ to the objective space Ψ^M, i.e., F: $\Omega \rightarrow \Psi^M$. Based on Eq.(1), four essential concepts about MOP can be defined as follows.

Definition 1: Pareto domination. Given any two vectors $u=[u_1, u_2, \ldots, u_M]$ and $w=[w_1, w_2, \ldots, w_M]$ in the objective space, we say that ***u* dominates *w*** if $u_m \leq w_m$ for all $m=1, 2, \ldots, M$ and $u \neq w$, denoted as $u \prec w$.

Definition 2: Pareto optimal. A solution vector $x \in \Omega$ is **Pareto optimal** if there is no $x^* \in \Omega$ such that $F(x^*)$ dominates $F(x)$.

Definition 3: Pareto set. The Pareto set (*PS*) is a set of the Pareto optimal solutions, which can be represented as

$$PS = \{x \in \Omega \text{ and } x \text{ is Pareto optimal}\} \quad (2)$$

Definition 4: Pareto front. The Pareto front (*PF*) is regarded as the image of the Pareto solutions in the objective space [28]-[30], which can be represented as

$$PF = \{F(x) | x \in PS\} \quad (3)$$

B. Multitask Optimization Problem

As this paper proposes to transform the MOP into a MTOP, the definition of MTOP is given herein. Generally speaking, a MTOP can be formulated as follows. Given K optimization tasks (without loss of generality assuming they are all minimization problems), denoted as $T_1, T_2, \ldots, T_K$, where the corresponding objective function of the task T_k is g_k, the MTOP requires the algorithm to find the optimal solution x_k for each task T_k such that

$$x_k = \arg\min g_k(x), \ k = 1, 2, ..., K \quad (4)$$

where $x_1, x_2, \ldots$, and x_K can belong to the same or different search spaces.

C. Related Work on MOP

To date, there have been many studies about multiobjective optimization evolutionary algorithms (MOEAs), which can be roughly classified into six categories, as briefly reviewed in the following contents.

The first category is based on dominance. In this category, the Pareto dominance-based methods are representative, which select individuals based on the Pareto dominance relationship. For example, nondominated sorting genetic algorithm II (NSGA-II) [5] and the improved strength Pareto EA [6] are two representative Pareto dominance-based evolutionary algorithms. As a representative dominance-based swarm intelligence algorithm, multiobjective particle swarm optimization (PSO) determines the learning direction of particles based on Pareto dominance [7]. Moreover, Yang *et al.* [32] proposed the grid-based evolutionary algorithm (GrEA), which introduced the grid dominance to enhance the selection toward the optimal direction. Tian *et al.* [33] proposed a modified NSGA-II with a strengthened dominance relation (NSGA-II-SDR) to balance the convergence and diversity.

The second category is the decomposition-based approach. The main idea of this category is to decompose the MOP into several subproblems according to a set of weight vectors, and then the population continuously evolves toward the PF by solving each subproblem. Since the decomposition-based MOEA (MOEA/D) proposed by Zhang and Li [8] in 2007, many decomposition-based algorithms have been studied. For example, Zhu *et al.* [34] proposed a MOEA/D algorithm with a detect and escape method that can escape from optimization stagnation. Chen *et al.* [35] proposed a novel metric to measure the subspace contribution to the population convergence and allocate resources accordingly. Moreover, MOEA/D variants have also been proposed to solve MaOPs. For example, Yuan *et al.* [36] calculated the vertical distance between the individual and weight vector to maintain the diversity of solutions in the high objective space, so as to balance the convergence and diversity. Also to balance population convergence and diversity, Cheng *et al.* [37] proposed a reference vector-guided evolutionary algorithm (RVEA) that used angle-penalized distance based on reference vectors. Although experimental results have shown the efficiency of decomposition-based algorithms, it has also been shown that the setting of weight vectors can be significant to the algorithm performance, and MOPs with different objectives may need different weight vectors [38].

The third category is based on indicators. The methods of this category evaluate and select individuals based on some performance preference indicators. For instance, the inverted generalized distance (IGD) [10] and hypervolume (HV) [11] indicators, which are widely-used indicators for the investigation of MOEAs, were adopted in [40] and [41] to guide the evolution, respectively. Note that the G in IGD should represent "generalized", which means a more generalized distance between two sets. Moreover, different strategies have also been studied with performance preference indicators for better individual selection and population evolution, such as the stochastic ranking strategy [42].

The fourth category is based on multiple populations and

coevolutionary methods. Zhan *et al.* [12] the first time proposed multiple populations for multiple objectives (MPMO) framework, where each population aims at one objective and all the populations coevolve to find the Pareto solutions. Due to the efficiency of MPMO, the MPMO framework has been extended to various real-world applications and obtained positive results, e.g., cloud workflow scheduling [43], supply chain configuration [44], job-shop scheduling [45], airline crew rostering optimization [46], and transportation optimization [47]. Moreover, based on the MPMO, Liu *et al.* [48] and Yang *et al.* [49] proposed enhanced algorithms for efficiently solving MaOP. Following the MPMO variants, other multiple population and coevolution methods have also been studied [13], such as the many-objective PSO based on coevolution [50] and the multiswarm-based algorithm [51].

The fifth category is based on preference information. For example, Yi *et al.* [14] studied preference angle and reference information-based dominance for selecting individuals, which can help guide the evolution. Moreover, the knee point is typical and widely-used preference information in this category, which can accelerate convergence and maintain diversity [15]. Therefore, many researchers studied how to use the knee point to solve MOP. For example, Zhang *et al.* [52] proposed a knee point-driven EA (KnEA) that utilized the knee point to guide the selection process to cover the shortage of Pareto dominance. Yu *et al.* [53] combined the α-dominance and knee-oriented dominance relationship to further identify the knee regions.

The sixth category refers to other methods that do not belong to the above categories. In this category, many pieces of research are about hybrid algorithms that mix more than one method in the above five categories. For example, NSGA-III [16] is proposed by integrating the NSGA-II with a reference points-based method. Besides, some novel methods have also been studied to solve MOP. Feng *et al.* [25] extended the original MOP to multiobjective MTOP by constructing several simplified MOP variants, so as to reduce dimension difficulties and utilize the knowledge transfer to solve the original MOP more easily. Liu *et al.* [54] put forward a novel multiobjective framework for many-objective optimization (Mo4Ma) framework, so as to maintain good diversity and convergence in high-dimensional objective space.

Although the methods and algorithms in the above categories have attempted to solve MOP in different ways, they do not solve the MOP as a MTOP. Differently, this paper fully recognizes the relationship between MOP and MTOP and proposes to solve the MOP as a MTOP, where the knowledge transfer methods for solving MTOP can be utilized to enhance the overall optimization results. The proposed MTMOEA, as a kind of new method for solving MOP by using MTO, is different from existing MOEA methods such as the decomposition-based methods. First, in problem transformation, although both MTMOEA and MOEA/D will transform the MOP into multiple tasks or subproblems to obtain multiple Pareto optimal solutions, their transformation mechanisms are different. Specifically, the proposed MTMOEA uses dynamic target points on an approximated PF rather than fixed weights in the decomposition method for problem transformation. That is, our method can transform a MOP into a MTOP automatically and accurately. Therefore, our method is more general and the decomposition can be essentially regarded as a special case. As a result, MTMOEA can yield the adaptive ability to obtain more promising performance when compared with MOEA/D. Second, in the optimization mechanism, the proposed MTMOEA focuses more on the task relationship and knowledge transfer among relevant tasks to obtain better optimization results. This is due to that knowledge transfer is an essential component in MTO for enhancing the optimization results of multiple tasks, which is also a important feature and research issue in MTO. For example, in the MTMOEA, the AIKT is proposed to implicitly transfer knowledge among individuals in relevant tasks. Therefore, the AIKT can help to generate better offspring for multiple tasks, which can enhance the overall optimization efficiency. Based on the above, the MTMOEA (and the paradigm of solving MOP via MTO) can help to extend the research scope of MOEA on both the problem transformation and optimization mechanism, which is also a new method and new paradigm for solving MOP.

D. Related Work on MTOP

Although MTOP is an emerging research topic, it has obtained fast development in recent years [55]-[58]. Existing works for MTOP can be roughly classified into two categories: implicit knowledge transfer-based approach and explicit knowledge transfer-based approach.

In the implicit knowledge transfer-based approach, knowledge transfer is achieved implicitly via evolutionary operators with the individuals for different tasks. In this category, the MFEA [4] and its variants are representative algorithms. The MFEA employs a single population in a unified search space, where each individual in the population targets one of the multiple tasks based on its skill factor (e.g., its fitness for each task). Then, knowledge transfer can be achieved implicitly via crossover operations with individuals for different tasks. Due to the efficiency of the basic MFEA framework, many variants have been studied and proposed based on this framework. For example, Bali *et al.* [18] studied the online transfer parameter estimation and proposed MFEA-II. Gong *et al.* [57] studied the dynamic resource allocation strategy for enhancing the MFEA. Ding *et al.* [58] proposed a generalized MFEA with a decision variable translation and shuffling strategies. Zhou *et al.* [59] studied the adaptive knowledge transfer for MFEA and obtained promising results. Besides, more implicit knowledge transfer methods have also been researched based on the evolutionary operators in different EC algorithms, such as genetic programming [60].

Differently, the second category focuses on explicit knowledge transfer between multiple populations or swarms for different tasks. Feng *et al.* [19] proposed an autoencoding-based explicit genetic transfer, so as to transfer knowledge between populations aiming at different tasks. Lin *et al.* [24] explored a more positive transfer by transferring valuable solutions and proposed an algorithm for multiobjective MTOPs. Different from the task-specific knowledge transfer mentioned above, Li *et al.* [20] further proposed a novel meta-knowledge transfer method to transfer the meta-knowledge among populations for different tasks. Wu *et al.* [61] considered the shift invariance between tasks and proposed a successful solution transfer method to transfer the suitable parameters between similar tasks. Jiang *et al.* [62]

proposed to transfer both the shape knowledge and domain knowledge between similar tasks to solve MaTOP. In addition, Wu *et al.* [63] proposed an orthogonal transfer method to handle the heterogeneous search spaces and dimension similarities to achieve high-quality transfer. In addition, Jiang *et al.* [64] also proposed a block-level knowledge transfer to enhance the knowledge transfer between dimensions of the same or different tasks.

III. The Proposed MTMOEA

A. Treating Multiobjective Optimization Problem as Multitask Optimization

To begin with, the theorem about "*Optimization Equivalence between MOP and MTOP*" (i.e., **Theorem 1)** is put forward to show the relationship between MOP and MTOP.

Theorem 1: If a MOP needs to find K target points on the PF, then the MOP can be regarded as a MTOP with K single-objective optimization tasks.

Proof: Without loss of generality, we denote the K target points on the PF (i.e., in the objective space) that need for a MOP as $\mathbf{Z}_1$, $\mathbf{Z}_2$, …, and $\mathbf{Z}_K$. Then, the optimization goal of the MOP is to find K solutions x_1, x_2, …, and x_K, such that:

$$x_k = \arg\min\left(d(F(x),\mathbf{Z}_k)\right),\ k=1,2,...,K \tag{5}$$

where $F(x)$ is the objective vector of x as defined in Eq.(1), and $d(\mathbf{a}, \mathbf{b})$ measures the distance between vector points $\mathbf{a}$ and $\mathbf{b}$. By defining $g_k(x|\mathbf{Z}_k)= d(F(x), \mathbf{Z}_k)$, the Eq.(5) can be rewritten as:

$$x_k = \arg\min g_k(x|\mathbf{Z}_k),\ k=1,2,...,K \tag{6}$$

Compared with Eq.(4), i.e., the problem formulation of MTOP, we can see that Eq.(6) is actually a form of MTOP with K single-objective optimization tasks, where $g_k(x|\mathbf{Z}_k)$ is the fitness function of the kth task. Therefore, the proof is finished.

According to **Theorem 1**, we can treat the MOP as a MTOP when assuming that the MOP just needs to find K target points on the PF (K is a positive and finite number). In fact, this assumption can be supported in practical applications due to two reasons. First, when solving real-world MOPs, as the number of Pareto optimal solutions can be very large and increase rapidly as the number of objectives increases, finding a sufficient number (rather than all) of Pareto optimal solutions is a more realistic and possible way for solving the MOP, where each Pareto optimal solution corresponds to a point on the PF (as defined in **Definition 4**). Second, the final output of algorithms, such as many existing MOP algorithms [9]-[12], is always only a subset of Pareto optimal solutions, i.e., only some points on (or close to) the PF, because the available computational budgets, memory, and storage are not infinite but are often expensive and limited. Therefore, it is rational and acceptable to treat the MOP as MTOP, so as to get enough Pareto optimal solutions as the final results for the corresponding MOP. In addition, as solving MOP as a MTOP with K tasks just need to find K (rather than all) Pareto optimal solutions, the problem complexity reduces as K decreases, which is a great advantage. Based on the above, treating MOP as MTOP is a potential direction for solving MOP.

When transforming the MOP into a MTOP via Eq.(6), various distance measurements can be used to calculate the function $d(F(x), \mathbf{Z}_k)$), so as to build up the fitness function of each task (i.e., the $g_k(x|\mathbf{Z}_k)$). Without loss of generality, this paper uses the Euclidian distance, a widely-used distance metric, to compute $d(F(x), \mathbf{Z}_k)$) in the implementation. That is, the fitness function in Eq.(6) can be rewritten as:

$$\begin{aligned} g_k(x\,|\,\mathbf{Z}_k) &= d\left(F(x),\mathbf{Z}_k\right) \\ &= \left\|F(x)-\mathbf{Z}_k\right\| \end{aligned} \tag{7}$$

where $\|\mathbf{a}\|$ will return the Euclidian norm of $\mathbf{a}$. Note that besides the Euclidian distance, other distance measurements can be also used and may have different advantages for different MOPs [30]. However, as the focus of this paper is not on the influence of different distance measurements, the simple Euclidian distance is used herein while the investigations of others can be potential as future work.

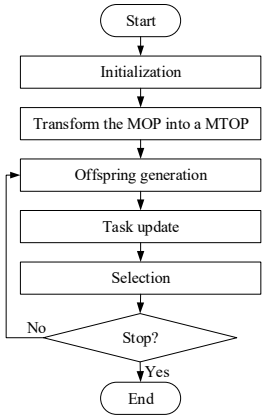


Fig. 2. The flowchart of the MTMO framework.

In addition, as the real PF is unknown prior, if we want to solve the MOP as a MTOP formulated as Eq.(6), the target points $\mathbf{Z}_1$, $\mathbf{Z}_2$, …, and $\mathbf{Z}_K$ need to be estimated. That is, the target point of each task in the transformed MTOP should be estimated. Therefore, this paper proposes the TPE strategy to estimate the target points to help transform the MOP into the MTOP more accurately, which is described in Section III-C.

B. Multiple Tasks for Multiple Objectives Framework

To solve the MOP as a MTOP efficiently, this paper proposes the MTMO framework. The general flowchart of the MTMO is shown in Fig. 2. In the MTMO, the multiple tasks (e.g., K tasks) of a MTOP are generated based on Eq.(7) and can be updated in every generation. For solving the MTOP with K tasks, the MTMO evolves a population with N individuals. Note that N should not be less than K (i.e., $N \geq K$), so that there can be at least one individual corresponding for each task. During the evolution, three procedures in the MTMO are conducted iteratively until the stop criterion is met, where the three procedures are: offspring generation, task update, and selection. The offspring generation procedure generates offspring (i.e., new individuals) via evolutionary operators, and then evaluates the objective value of the new individuals. Based on the objective value and the target points, the fitness of individuals for different tasks (called task fitness) can be calculated according to Eq. (7), which does not need additional fitness evaluations. To solve MOP as a MTOP more efficiently, this paper proposes the AIKT strategy to generate new

Algorithm 1: Target Point Estimation

Input: $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L$ - the L potential points;
Λ - the set of found non-dominated solutions;
F - consists of M objective functions;
λ - the scale factor for estimating PF;
Output: $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K$ - the K estimated target points.

1: **Begin**
2: // to approximate the PF with potential points
3: **For** i=1 to L **Do** // for each $\mathbf{Q}_i$
4: $q_i \leftarrow$ the point with smallest vertical distance in Λ;//see Eq. (11)
5: $r_i \leftarrow$ the scale value of $\mathbf{Q}_i$ based on $F(q_i)$;// see Eq.(9)
6: $\mathbf{Q}_i \leftarrow$ adjust $\mathbf{Q}_i$ with λ, r_i, and $\|\mathbf{Q}_i\|$; // see Eq.(8)
7: **End For**
8: // to obtain estimated target points based on potential points
9: $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K \leftarrow$ the K cluster centroids of $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L$;
10: **End**

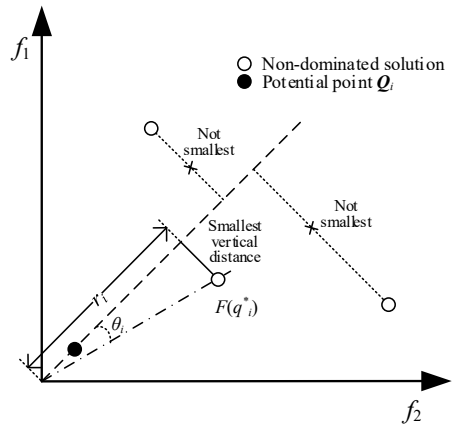


Fig. 3. The relationship among the variables in TPE.

individuals via knowledge transfer, which will be detailed in Section III-D. Based on the information of the individuals in the current population, the task update procedure updates the fitness function of all tasks, i.e., updates the estimated target points $\mathbf{Z}_k$ ($1 \le k \le K$) in Eq. (7). Herein, as mentioned before, this paper proposes the TPE strategy for the task update procedure to help more accurately update the tasks, which will be detailed in Section III-C. Then, according to the updated fitness functions of all tasks, the selection procedure selects N better individuals to form the new population. The new population will enter the loop to continue the evolution if the stop criterion is not met. Otherwise, the algorithm outputs the best-found solutions and finishes.

C. Target Point Estimation

The TPE is proposed to estimate the target points for transforming a MOP into a MTOP automatically and efficiently. To estimate the target points that are distributed on the PF evenly, the TPE has two main steps: 1) to approximate the PF, and 2) to obtain some points (e.g., K points) on the approximated PF evenly as the estimated target points.

The pseudo code of TPE is shown in **Algorithm 1**, where line 2 to line 7 and line 8 to line 9 are for approximating the PF and for computing estimated target points, respectively. Note that when approximating the PF, as the PF may consist of an infinite number of points, L potential points are used to represent the approximated PF.

Algorithm 2: Archive-based Implicit Knowledge Transfer

Input: P_g - the population at generation g;
arch - the archive;
N - the population size;
T - the neighborhood size for knowledge transfer;
Output: *Offspring* - the generated offspring;

1:**Begin**
2: *Offspring* $\leftarrow$ empty set;
3: **For** i=1 to N **Do**
4: *Index* $\leftarrow$ the index set of T individuals nearest to $P_{g,i}$ in *arch*;
5: $j \leftarrow$ a random index in *Index*;
6: // generate a new offspring via evolutionary operator
7: $x \leftarrow$ perform evolutionary operator with $P_{g,i}$ and $arch_j$;
8: *Offspring* $\leftarrow$ *Offspring* $\cup$ { x };
9: **End For**
10:**End**

To begin with, the inputs of the **Algorithm 1** are L original potential points that are uniformly pre-sampled in $[0, 1]^M$, where M is the number of objectives, i.e., the dimension of the objective space. Then, to approximate the shape of PF, the non-dominated solutions are obtained from the current population and stored in a set, denoted as Λ. With the non-dominated solutions in Λ, each of the L original potential points (e.g., $\mathbf{Q}_i$) will be scaled in the objective space as:

$$\mathbf{Q}_i = \lambda \cdot r_i \cdot \frac{\mathbf{Q}_i}{\|\mathbf{Q}_i\|} \tag{8}$$

where λ is the scale factor that controls the distance between the approximated PF and the front shaped by non-dominated solutions in the current population, and the r_i is the scale value of i^{th} potential point, which is computed as

$$r_i = \frac{F(q_i^*) \cdot \mathbf{Q}_i^T}{\|\mathbf{Q}_i\|} \tag{9}$$

where q_i^* is the point that has the smallest vertical distance to the $\mathbf{Q}_i$ in the objective space. The q_i^* with the smallest vertical distance to the $\mathbf{Q}_i$ can be written as:

$$q_i^* = \arg\min_{q_j \in \Lambda} \left(\|F(q_j)\|^2 - (\|F(q_j)\| \cdot \cos\theta_{i,j})^2 \right)^{1/2} \tag{10}$$

where $\theta_{i,j}$ is the angle between vector $\mathbf{Q}_i$ and $F(q_j)$ in the objective space. As $\sin^2\theta_{i,j} + \cos^2\theta_{i,j} = 1$, the Eq.(10) can also be rewritten as

$$q_i^* = \arg\min_{q_j \in \Lambda} \left(\|F(q_j)\| \cdot \sin\theta_{i,j} \right) \tag{11}$$

To provide a clearer demonstration, Fig. 3 shows the relationship between the variables in TPE, where q_i^* is the non-dominated solution with the smallest vertical distance to $\mathbf{Q}_i$ in objective space, and r_i is the scale value for $\mathbf{Q}_i$ as calculated by Eq.(9). Note that the approximating strategy in the Eq.(9) and Eq.(11) is different from some existing PF approximating strategies in two aspects. First, the key points for approximating PF are calculated via different formulas. For example, the Eq.(8) is with a scaled factor λ while existing work is not [16], [65]. Second, the aim of the PF approximating strategy is different. In particular, the approximation strategy of this paper is used as the first step in the proposed TPE to help better estimate the target points for constructing the transformed MTOP, while those in existing work are not designed for constructing a MTOP.

Based on the PF represented by L scaled potential points (i.e.,

Algorithm 3: MTMOEA

Input: M- the number of objectives;
F - consists of M objective functions;
N- the population size;
K- the number of tasks (the number of targeted points);
L - the number of potential points;
λ - the ratio for approximating PF;
T - the neighborhood sizes for knowledge transfer;
MFES - the maximum number of available FEs;
Output: S - the final solution set with N solutions.

1:**Begin**
2: /* Initialization */
3: $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L \leftarrow$ sample L potential points in $[0,1]^M$;
4: $g \leftarrow 1$; // the generation index
5: $P_g \leftarrow$ initialized population with N individuals;
6: Evaluate individuals in P_g with F;
7: $FEs \leftarrow N$; // the FEs cost for initialization
8: $arch \leftarrow P_g$; // Initial archive
9: /* Transforming the MOP into a MTOP */
10: $P_{can} \leftarrow P_g$;
11: $\Lambda \leftarrow$ non-dominated solutions in P_{can};
12: $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K \leftarrow$Perform TPE; //**Algorithm 1**
13: Develop the fitness functions of K tasks; // refer to Eq.(7)
14: **While** $FEs < MFES$ **Do**
15: /* Offspring Generation */
16: *Offspring* $\leftarrow$ generate new offspring via AIKT; //**Algorithm 2**
17: Evaluate the individuals in *Offspring* with F;
18: $FEs \leftarrow FEs + N$;
19: /* Task Update */
20: $P_{can} \leftarrow P_g \cup$ *Offspring*; // population with candidate individuals
21: $\Lambda \leftarrow$ non-dominated solutions in P_{can};
22: $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K \leftarrow$Perform TPE; //**Algorithm 1**
23: Update the fitness functions of K tasks; // refer to Eq.(7)
24: /* Multitask Selection */
25: $P_{g+1} \leftarrow$ perform selection among P_{can}; // **Algorithm 4**
26: /* Archive Update */
27: $arch \leftarrow$update archive with P_{can}; // **Algorithm 5**
28: $g \leftarrow g + 1$;
29: **End While**
30: $S \leftarrow P_g$; //the final solution set as the output
31:**End**

$\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L$), TPE obtains K points as the estimated target points. To obtain target points more evenly, the clustering technique is adopted herein. To be specific, this paper uses the K-means algorithm to cluster the L potential points into K groups, where the K centroids of the K groups are regarded as the K estimated targeted points, respectively. That is, the estimated targeted points can be obtained as:

$$\{\mathbf{Z}_1, \mathbf{Z}_2, ..., \mathbf{Z}_K\} = \text{Kmeans}(\{\mathbf{Q}_1, \mathbf{Q}_2, ..., \mathbf{Q}_L\}, K) \quad (12)$$

where Kmeans($\{\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L\}$,$K$) outputs the K centroids of the L potential points $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_L$. That is, the $\mathbf{Z}_1, \mathbf{Z}_2, \ldots$, and $\mathbf{Z}_K$ in Eq.(12) are the K estimated target points of the TPE. Note that the TPE is an automatic method for transforming a MOP into a MTOP, it has two parameters (L and λ) that need manual configurations. The influence study of L and λ are conducted in Section IV-D and F, respectively. In addition, the TPE is different from the decomposition method in that the TPE firstly approximates the PF, and then obtains some points on the approximated PF to automatically construct multiple optimization tasks, while the decomposition method uses a set of weight vectors to aggregate multiple objective functions into some subproblems as multiple tasks.

D. Archive-based Implicit Knowledge Transfer for Offspring Generation

The AIKT is proposed to generate offspring via knowledge transfer across multiple tasks, so as to enhance the optimization results for all tasks. **Algorithm 2** is the pseudo code of the AIKT. In general, the AIKT generates offspring based on an archive, where the archive contains individuals with good fitness for different tasks. For each individual i in the population at generation g (e.g., $P_{g,i}$), AIKT selects T individuals nearest to $P_{g,i}$ in the archive, where T is the neighborhood size for knowledge transfer. Herein, the "nearest" means the smallest Euclidean distance in the decision space between $P_{g,i}$ and other individuals in the archive. This is different from those approaches that select parents based on the distance in the objective space, e.g., MOEA/D selects parents based on the distance between corresponding aggregation weight vectors in the objective space. Then, a new offspring is generated via evolutionary operators with the $P_{g,i}$ and the random one of the T nearest individuals from the archive as the parents. That is, for generating the offspring, each parent pair contains two individuals, one from the current population (e.g., $P_{g,i}$) and one randomly selected from the T individuals in the archive that are nearest to $P_{g,i}$. Note that each individual in the population (e.g., $P_{g,i}$) can be good at a specific task, while the T individuals nearest to $P_{g,i}$ from the archive can be good at some other similar or relevant tasks. Therefore, the individual for one task can learn knowledge from other tasks via the evolutionary operators implicitly, so as to generate better offspring. Herein, without loss of generality, the widely-used evolutionary operators in the MOP research community, i.e., the simulated binary crossover and polynomial mutation [67], are used as the evolutionary operators in the AIKT.

E. The Complete Algorithm

Based on the above contents, this part derives the complete algorithm for solving MOP via MTO, which is named MTMOEA. The pseudo code of MTMOEA is given in **Algorithm 3.** After the initialization and transforming the MOP into a MTOP via the TPE, **Algorithm 3** iteratively carries out four procedures until the stop criterion is met (i.e., until the available FEs are consumed out). The four procedures are the offspring generation (lines 15-18), the task update (lines 19-23), the multitask selection (lines 24 and 25), and the archive update (lines 26 and 27). Finally, the MTMOEA outputs the final population as the solution set for the MOP. The details of the multitask selection and the archive update are as follows.

The multitask selection is proposed to select individuals based on the fitness functions of all different tasks. The pseudo code of the multitask selection is given in **Algorithm 4**. To be specific, multitask selection iteratively selects the promising individual for each task in a round-robin fashion until enough individuals are selected, as shown in lines 4 to line 9 of **Algorithm 4**.

The archive update is similar to the multitask selection (**Algorithm 4**), which also selects the promising individuals for each task in a round-robin fashion until enough individuals are selected. However, the archive update is to update the archive every generation to refresh the knowledge for different tasks, so as to enhance the AIKT for offspring generation. Therefore, different from the multitask selection, the archive update will not select a duplicated individual, so as to improve the individual diversity in the archive. Moreover, the maximum size of the archive is set the same as the population size. The

Algorithm 4: Multitask Selection

Input: F - consists of M objective functions;
N - the population size;
P_{cnd} - the candidate individual set for selecting N individuals;
$\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K$ - the K estimated target points;
Output: P_{g+1} - the population at generation g+1;

1: **Begin**
2: $P_{g+1} \leftarrow$ empty set;
3: $ni \leftarrow$ the smallest integer larger than N/K; // number of iterations
4: **For** i=1 to ni **Do**
5: **For** k=1 to K **Do**
6: $x \leftarrow$ the ith best individual in P_{can} for task k based on $\mathbf{Z}_k$;
7: $P_{g+1} \leftarrow P_{g+1} \cup \{x\}$;
8: **End For**
9: **End For**
10: $P_{g+1} \leftarrow$ the first N individuals in P_{g+1};
11: **End**

Algorithm 5: Archive Update

Input: P_{can} - the candidate individual set;
P_{g+1} - the population at generation g+1;
N_a - the maximum size of *arch*;
$\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_K$ - the K estimated target points;
Output: *arch* - the updated archive;

1:**Begin**
2: $arch \leftarrow P_{g+1}$;
3: Remove the duplicated solutions in *arch*;
4: Remove existing individuals in $arch \cap P_{can}$ from P_{can};
5: **While** the size of *arch* is smaller than N_a **Do**
6: **For** k=1 to K **Do**
7: $x \leftarrow$ the best individual in rest P_{can} for task k;
8: $arch \leftarrow arch \cup \{x\}$;
9: Remove x from P_{can}; // avoid duplicated selection
10: **End For**
11: **End While**
12: $arch \leftarrow$ the first N_a individuals in *arch*;
13: **End**

pseudo code of archive update is given in **Algorithm 5**, where line 3, 4, and 9 are to avoid selecting duplicated solutions. By doing so, we can ensure that the individuals in the archive are different, which can enhance individual diversity and enrich the knowledge for different tasks in the archive.

IV. Experimental Studies

A. Experiment Setup

So far, there have been many MOP benchmark test suites for evaluating algorithms, such as DTLZ [68] and WFG [69]. The problems in these different benchmark suits can help observe how the algorithm will behave on different MOPs. Recently, a test suite named MaF [27] has been proposed by combining the advantages of different benchmark suites including the DTLZ [68] and WFG [69] to better evaluate MOP algorithms. To be specific, the MaF benchmark suite contains 15 representative problems with different properties (i.e., MaF1 to MaF15) that are modified based on existing representative MOPs, as shown in Table I. As a result, the MaF1 to MaF15 have various characteristics and their objective number can be set according to the need of users. Moreover, the MaF has been adopted as the benchmark set for the IEEE CEC competition on MOP with more than three objectives since 2017. Therefore, this paper also adopts the MaF1 to MaF15 to investigate the proposed algorithm in the experimental part.

Furthermore, popular and state-of-the-art algorithms are used as the competitors and the number of objectives of the 15 problems is set as 3, 5, 10, and 20, so as to better study the efficiency of the proposed algorithm. The compared algorithms are: NSGA-III [16], NSGA-II-SDR [33], MOEA/D-DES [34], RVEA [37], KnEA [52], and Mo4Ma [54], which have been mentioned in Section II. The source codes of the first five algorithms are obtained from the PlatEMO platform [66]. Note that the parameters of crossover and mutation operators of different algorithms including the MTMOEA are the same. In addition, as for the parameters in MTMOEA, both the number of target points (i.e., the number of tasks) and the population size is N, the number of potential points is set as L=5×N, the scale factor λ is 0.95, and the neighborhood size T is $\lfloor 0.05 \times N \rfloor$.

Moreover, for some algorithms that require reference points/vectors, the population size is recommended to be set as the same as the number of reference points/vectors. To generate the reference points/vectors for a problem with M objectives, the Normal-boundary intersection (NBI) strategy [70] is widely used to determine the number of reference points/vectors. That is, the population size can be determined based on the NBI. In NBI, N reference points/vectors will be sampled, where $N = C_{M'}^{H_1+M'}$ with $M'=M-1$ and H_1 as the parameter that controls the division on each objective axis. For problems with many objectives, it is suggested to use a two-layer generation strategy with $N = C_{M'}^{H_1+M'} + C_{M'}^{H_2+M'}$, where H_1 represents the divisions on the outer layer while H_2 indicates the divisions on the insider layer

TABLE I
Description and Property of the Test problems

Problem	Description	Property of Pareto front
MaF1	Modified inverted DTLZ1	Linear
MaF2	DTLZ2BZ	Concave
MaF3	Convex DTLZ3	Convex, Multimodal
MaF4	Inverted badly-scaled DTLZ3	Convex, Multimodal
MaF5	Convex badly-scaled DTLZ4	Convex, Biased
MaF6	DTLZ5(I,M)	Concave, Degenerate
MaF7	DTLZ7	Mixed, Disconnected, Multi-modal
MaF8	Multi-Point Distance Minimization Problem	Linear, Degenerate
MaF9	Multi-Line Distance Minimization Problem	Linear, Degenerate
MaF10	WFG1	Mixed, Biased
MaF11	WFG2	Convex, Disconnected, Non-separable
MaF12	WFG9	Concave, Nonseparable, Biased Deceptive
MaF13	PF7	Concave, Unimodal, Nonseparable, Degenerate
MaF14	LSMOP3	Linear, Partially separable, Large scale
MaF15	Inverted LSMOP8	Convex, Partially separable, Large scale

TABLE II
Settings of Population Sizes for Problems with Different Number of Objectives

Number of objectives	(H_1, H_2)	Population size
3	(13, -)	105
5	(5, -)	126
10	(3, 2)	275
20	(2, 1)	230

Note that H1 and H2 are the simplex-lattice design factors for generating uniformly distributed reference/vectors on the outer boundaries and the inside layers, respectively.

TABLE III
STATISTICAL COMPARISONS RESULTS BETWEEN MTMOEA AND OTHER ALGORITHMS BASED ON IGD AND HV METRIC

Metric	Number of objectives	NSGA-III (+/≈/-)	NSGA-II-SDR (+/≈/-)	MOEA/D-DES (+/≈/-)	RVEA (+/≈/-)	KnEA (+/≈/-)	Mo4Ma (+/≈/-)
IGD	3	9/1/5	8/1/6	10/1/4	11/1/3	7/4/4	14/0/1
	5	8/1/6	11/1/3	9/3/3	10/1/4	9/0/6	13/1/1
	10	8/0/7	8/1/6	8/2/5	9/0/6	8/1/6	10/0/5
	20	10/1/4	5/2/8	7/4/4	9/0/6	9/2/4	7/3/5
	Total	35/3/22	32/5/23	34/10/16	39/2/19	33/7/20	44/4/12
HV	3	6/2/7	9/4/2	11/4/0	8/5/2	10/2/3	7/2/6
	5	9/1/5	7/3/5	10/3/2	11/1/3	9/1/5	7/1/7
	10	6/4/5	4/3/8	9/2/4	9/1/5	7/4/4	8/3/4
	20	7/3/5	7/2/6	8/3/4	8/2/5	7/3/5	10/2/3
	Total	28/10/22	27/12/21	38/12/10	36/9/15	33/10/17	32/8/20

The "+", "≈", and "-" represent that the proposed MTMOEA performs significantly better than, similar to, and significantly worse than the corresponding compared algorithm.

of the PF. Based on this, this paper also adopts the NBI strategy to determine the same population size N for all the algorithms, so as to provide a fair comparison. Particularly, Table II provides the value of corresponding H_1, H_2, and the population size for problems with a different number of objectives. In addition, without loss of generality, the initial potential target points (i.e., $\mathbf{Q}_1$, $\mathbf{Q}_2$, …, $\mathbf{Q}_L$) in the MTMOEA are also generated by the NBI. As the number of points generated by NBI can not be arbitrary, the number of generated points will be slightly smaller than L if L does not equal to $C_{M'}^{H_1+M'} + C_M^{H_2+M'}$ for any H_1 and H_2. That is, if $L \neq C_{M'}^{H_1+M'} + C_M^{H_2+M'}$, the NBI will sample $L' = C_{M'}^{H_1+M'} + C_M^{H_2+M'}$ points such that L' is closest to but smaller than L.

In addition, the maximum number of FEs is configured as $N \times 100$ for each algorithm on each problem, so as to evaluate the algorithm efficiency. To reduce statistical errors, each algorithm is tested 30 times independently on each problem and the average results are used for the comparisons.

B. Evaluation Metrics

To evaluate the algorithm performance, two widely-used metrics, i.e., IGD and HV, are adopted in the experimental studies and comparisons. The mean and standard values of the IGD and HV results over 30 runs are reported. As recommended in [27], the reference set of IGD contains 10000 reference points uniformly sampled on the true PF, and the reference point of HV is a vector of all ones, i.e., (1,1, …,1). Note that when calculating the HV, the objective value of each solution will be normalized based on a nadir point of the true PF, and more detailed calculation procedures can refer to [27]. Moreover, the Wilcoxon's rank sum test with a significant level α=0.05 is also used for algorithm comparisons. That is, according to the Wilcoxon's rank sum test, this paper uses the symbols "+", "≈", and "-" to show that the proposed algorithm performs significantly better than, similar to, and significantly worse than the compared algorithms, respectively. In addition, the best results in the comparison will be marked in **boldface**.

C. Comparisons with State-of-the-art Algorithms

Table III provides the statistical comparison results between MTMOEA and six state-of-the-art algorithms on 60 problems with a different number of objectives, where the detailed experimental results are given in Table S.I and Table S.II in the supplementary material.

As shown in Table III, the MTMOEA can outperform all compared algorithms on more than 30 problems (i.e., more than half of all the test problems) in term of IGD, and more than 27 problems in term of HV. This suggests the great problem-solving ability of MTMOEA. Moreover, Table S.I shows that the MTMOEA can obtain the best IGD results (as marked in **boldface**) on 17 problems, while the NSGA-III, NSGA-II-SDR, MOEA/D-DES, RVEA, KnEA, and Mo4Ma get the best IGD results only on 3, 14, 6, 7, 9, and 4 problems, respectively. In particular, in terms of both the IGD and HV, the MTMOEA significantly works better on MaF3, Ma4, MaF14, and MaF15 with different numbers of objectives. The distinct characteristics of these problems are that the MaF3 and MaF4 are multimodal MOPs while the MaF14 and MaF15 have large-scale MOPs, which both contain many local optima in the complicated and large-scale search space. The MTMOEA has special superior performance on these very complex problems. This may be due to that by transforming these MOPs into MTOPs, the MTMOEA can use MTO and knowledge transfer methods to efficiently find a set of optimal solutions with good diversity in the complicated and large-scale search space. In addition, as can be seen in Table III, no matter whether on tested MOPs (the number of objectives is 3) or MaOPs (e.g., the number of objectives is 5, 10, or 20), the proposed MTMOEA significantly outperforms almost all the compared algorithms on more than half problems, which show that the MTMOEA can work well on both the MOP and MaOP.

Moreover, for a better visualization of the algorithm efficiency, Fig. S.1 of the supplementary material plots the HV convergence curve of the seven algorithms during the progress of fitness evaluations on MaF9, MaF14, and MaF15 with 3, 5, 10, and 20 objectives, respectively. The three problems are selected because that the MaF9 is a classical problem with a linear Pareto front while both MaF14 and MaF15 are large-scale MOPs, which can help to observe the algorithm efficiency in different scenarios. As shown in Fig. S.1, the HV convergence of MTMOEA is competitive with other algorithms on the MaF9 with different objectives. While on MaF14 and MaF15, the MTMOEA can obtain significantly better HV results more quickly than other algorithms. This validates the optimization efficiency of MTMOEA, especially on large-scale MOPs.

Based on the above, the comparison results have shown the significant efficiency of MTMOEA. Therefore, solving MOP as a MTOP is promising.

D. Influence of the Number of Potential Points

In TPE, the target point of each task in the transformed MTOP is obtained based on the L potential points. Therefore,

TABLE IV
COMPARISONS OF IGD RESULTS AMONG MTMOEA VARIANTS WITH DIFFERENT NUMBER OF POTENTIAL POINTS

Problem	MTMOEA(L=5×N)	MTMOEA(L=2×N)	MTMOEA(L=3×N)	MTMOEA(L=10×N)	MTMOEA(L=20×N)
MaF1	6.4836E-2 (2.14E-3)	6.7770E-2 (2.12E-3) +	6.6645E-2 (2.39E-3) +	6.5190E-2 (2.38E-3) ≈	**6.3171E-2 (2.45E-3) -**
MaF2	5.4559E-2 (3.03E-3)	5.5497E-2 (2.21E-3) ≈	5.5190E-2 (2.58E-3) ≈	5.5801E-2 (3.31E-3) ≈	**5.4417E-2 (2.39E-3) ≈**
MaF3	**6.9995E+1 (7.99E+1)**	9.1141E+1 (1.05E+2) ≈	9.3968E+1 (1.64E+2) ≈	8.0919E+1 (1.01E+2) ≈	1.2081E+2 (1.75E+2) ≈
MaF4	2.0512E+1 (1.15E+1)	2.2346E+1 (1.71E+1) ≈	**1.8425E+1 (1.28E+1) ≈**	2.1444E+1 (1.16E+1) ≈	2.0535E+1 (1.46E+1) ≈
MaF5	3.8697E-1 (3.82E-1)	5.6889E-1 (5.74E-1) ≈	5.2406E-1 (6.01E-1) ≈	**3.3732E-1 (3.21E-1) -**	4.9234E-1 (5.70E-1) +
MaF6	1.2276E-2 (1.44E-3)	1.3919E-2 (1.34E-3) +	1.2851E-2 (1.44E-3) ≈	**1.2268E-2 (1.36E-3) ≈**	1.2477E-2 (1.36E-3) ≈
MaF7	3.3199E-1 (3.64E-2)	3.3199E-1 (2.82E-2) ≈	3.2603E-1 (2.69E-2) ≈	3.4289E-1 (3.85E-2) ≈	**3.2559E-1 (4.24E-2) ≈**
MaF8	3.0672E-1 (3.86E-1)	2.4491E-1 (1.28E-1) ≈	3.3110E-1 (3.59E-1) ≈	2.6308E-1 (2.12E-1) ≈	**1.9132E-1 (7.68E-2) ≈**
MaF9	1.7344E-1 (1.06E-1)	1.7148E-1 (1.07E-1) ≈	1.6951E-1 (1.01E-1) ≈	**1.5387E-1 (9.08E-2) ≈**	2.0352E-1 (2.27E-1) ≈
MaF10	6.9509E-1 (1.23E-1)	6.5042E-1 (1.27E-1) ≈	**6.2681E-1 (6.73E-2) -**	6.5732E-1 (7.45E-2) ≈	6.6352E-1 (9.37E-2) ≈
MaF11	1.8118E-1 (6.44E-3)	1.8403E-1 (6.41E-3) ≈	1.8218E-1 (8.05E-3) ≈	1.7948E-1 (6.52E-3) ≈	**1.7450E-1 (6.86E-3) -**
MaF12	2.4733E-1 (1.25E-2)	2.5488E-1 (2.17E-2) +	2.5655E-1 (2.17E-2) +	2.4124E-1 (9.98E-3) -	**2.3963E-1 (7.93E-3) -**
MaF13	8.7601E-2 (5.29E-3)	9.3550E-2 (6.80E-3) +	9.2010E-2 (5.09E-3) +	8.7274E-2 (4.29E-3) ≈	**8.4601E-2 (4.16E-3) -**
MaF14	1.9886E+0 (9.70E-1)	1.8959E+0 (7.40E-1) ≈	**1.7978E+0 (6.47E-1) ≈**	2.2309E+0 (1.05E+0) ≈	2.0574E+0 (8.21E-1) ≈
MaF15	3.4496E-1 (3.69E-2)	**3.2610E-1 (2.55E-2) -**	3.3418E-1 (3.55E-2) ≈	3.4808E-1 (3.60E-2) ≈	3.4965E-1 (2.61E-2) ≈
+/≈/-	NA	4/10/1	3/11/1	0/13/2	1/10/4

TABLE V
COMPARISONS OF IGD RESULTS AMONG MTMOEA VARIANTS WITH DIFFERENT SCALE FACTORS

Problem	MTMOEA (λ=0.95)	MTMOEA (λ=0.80)	MTMOEA (λ=0.85)	MTMOEA (λ=0.90)	MTMOEA (λ=1.00)
MaF1	6.4836E-2 (2.14E-3)	6.3737E-2 (2.22E-3) ≈	6.3985E-2 (2.02E-3) ≈	**6.3655E-2 (2.12E-3) ≈**	6.6394E-2 (2.67E-3) +
MaF2	5.4559E-2 (3.03E-3)	**5.1218E-2 (2.88E-3) -**	5.1444E-2 (2.35E-3) -	5.2234E-2 (2.62E-3) -	7.4465E-2 (3.62E-3) +
MaF3	6.9995E+1 (7.99E+1)	**4.8684E+1 (4.70E+1) ≈**	8.7964E+1 (8.79E+1) ≈	9.5461E+1 (8.84E+1) ≈	9.0673E+1 (1.06E+2) ≈
MaF4	2.0512E+1 (1.15E+1)	**1.3972E+1 (9.48E+0) -**	1.4284E+1 (1.16E+1) -	1.6408E+1 (1.00E+1) ≈	2.7048E+1 (2.04E+1) ≈
MaF5	**3.8697E-1 (3.82E-1)**	9.1376E-1 (8.70E-1) +	6.0406E-1 (4.44E-1) +	5.3236E-1 (5.03E-1) +	4.8821E-1 (5.18E-1) ≈
MaF6	1.2276E-2 (1.44E-3)	**1.1717E-2 (9.15E-4) ≈**	1.1813E-2 (8.85E-4) ≈	1.2128E-2 (1.40E-3) ≈	1.2315E-2 (1.47E-3) ≈
MaF7	3.3199E-1 (3.64E-2)	2.0080E+0 (1.10E-1) +	1.7447E+0 (3.59E-1) +	1.2641E+0 (3.01E-1) +	**1.9202E-1 (1.85E-2) -**
MaF8	3.0672E-1 (3.86E-1)	2.7600E-1 (1.76E-1) ≈	2.2660E-1 (1.33E-1) ≈	**2.0209E-1 (1.34E-1) ≈**	2.4593E-1 (1.81E-1) ≈
MaF9	1.7344E-1 (1.06E-1)	2.0513E-1 (1.25E-1) ≈	1.9184E-1 (1.64E-1) ≈	1.3766E-1 (5.10E-2) ≈	**1.3185E-1 (7.01E-2) ≈**
MaF10	6.9509E-1 (1.23E-1)	7.5386E-1 (1.19E-1) ≈	6.5522E-1 (9.78E-2) ≈	**6.4189E-1 (1.05E-1) -**	7.5649E-1 (8.94E-2) +
MaF11	**1.8118E-1 (6.44E-3)**	4.5533E-1 (4.74E-2) +	3.5509E-1 (3.95E-2) +	2.3896E-1 (2.80E-2) +	1.8542E-1 (5.94E-3) +
MaF12	**2.4733E-1 (1.25E-2)**	5.2345E-1 (1.16E-1) +	3.4542E-1 (3.54E-2) +	2.6047E-1 (1.19E-2) +	2.5575E-1 (1.03E-2) +
MaF13	8.7601E-2 (5.29E-3)	9.2429E-2 (5.24E-3) +	8.7320E-2 (4.55E-3) ≈	**8.6267E-2 (6.39E-3) ≈**	1.0011E-1 (4.66E-3) +
MaF14	1.9886E+0 (9.70E-1)	**1.0442E+0 (3.56E-1) -**	1.2035E+0 (3.96E-1) -	1.6800E+0 (7.36E-1) ≈	2.2241E+0 (8.90E-1) ≈
MaF15	3.4496E-1 (3.69E-2)	3.8694E-1 (7.61E-2) +	**3.3544E-1 (3.83E-2) ≈**	3.3619E-1 (4.16E-2) ≈	3.7018E-1 (4.81E-2) +
+/≈/-	NA	6/6/3	4/8/3	4/9/2	7/7/1

the influence of L is studied in this part. Specifically, as the original MTMOEA uses L=5×N potential points, the original MTMOEA, denoted as MTMOEA(L=5×N) in this part, is compared with four variants with different L, which are denoted as MTMOEA(L=2×N), MTMOEA(L=3×N), MTMOEA(L=10×N), and MTMOEA(L=20×N), respectively. The MTMOEA(L=2×N) and MTMOEA(L=3×N) can represent the influence of a smaller L, while MTMOEA L=10×N) and MTMOEA(L=20×N) can reflect the influence of a larger L. Note that as mentioned earlier, the potential points are initially sampled by NBI, and therefore the actual number of potential points may be slightly smaller than L during the evolution.

The comparison results are given in Table IV. As can be seen in Table IV, MTMOEA(L=5×N) generally has better performance than MTMOEA(L=2×N) and MTMOEA(L=3×N) while slightly worse performance than MTMOEA(L=10×N) and MTMOEA(L=20×N). In detail, MTMOEA(L=5×N) performs significantly better/worse than MTMOEA(L=2×N) and MTMOEA(L=3×N) on 4/1 and 3/1 problems, respectively, while significantly better/worse than MTMOEA(L=10×N) and MTMOEA(L=20×N) on 0/2 and 1/4 problems, respectively. This indicates that the larger the L is, the better the algorithm performance can be. The reason may be that more potential points can help to estimate the PF more accurately and thus can provide more information to obtain better-estimated target points. However, it should also be noted that more potential points will also result in more computational costs. Therefore, L=5×N can be a good setting that balances the algorithm performance and computational cost, and is used in this paper.

E. Influence of the Scale Factor

This part analyzes the scale factor in TPE for estimating the PF, i.e., the value of λ in Eq.(8). As the λ in original MTMOEA is 0.95, this part compares the original MTMOEA with its variants using λ=0.80, λ=0.85, λ=0.90, and λ=1.00, so as to analyze the influence of λ. For simplicity, the original MTMOEA and the four variants are denoted as MTMOEA(λ=0.95), MTMOEA(λ=0.80), MTMOEA(λ=0.85), MTMOEA(λ=0.90), and MTMOEA(λ=1.00), respectively. The comparison results of the five algorithms on the 15 3-objective problems are given in Table V. As can be seen in Table V, MTMOEA(λ=0.95) generally performs similarly to MTMOEA(λ=0.85) and MTMOEA(λ=0.90), but perform significantly better than MTMOEA(λ=0.80) and MTMOEA(λ=1.00). To be specific, according to the Wilcoxon's rank sum test, MTMOEA(λ=0.95) performs similarly to MTMOEA(λ=0.85) and MTMOEA(λ=0.90) on 8 and 9 of the 15 problems (i.e., more than a half), and perform significantly better/worse than these two variants on 4/3 and 4/2 problems, respectively. As for the rest two variants, the

TABLE VI
COMPARISONS OF IGD RESULTS BETWEEN THE ORIGINAL MTMOEA AND THE MTMOEA VARIANT WITHOUT CLUSTERING

Problem	Obj.	MTMOEA	MTMOEA-w/o-C	Problem	Obj.	MTMOEA	MTMOEA-w/o-C
MaF1	3	6.4836E-2 (2.14E-3)	**5.9049E-2 (3.53E-4) -**	MaF1	20	**4.9678E-1 (8.27E-3)**	5.0130E-1 (7.39E-3) +
MaF2	3	5.4559E-2 (3.03E-3)	**5.3243E-2 (1.00E-3) -**	MaF2	20	**1.7910E-1 (1.36E-3)**	1.8227E-1 (1.63E-3) +
MaF3	3	6.9995E+1 (7.99E+1)	**6.6321E+1 (1.20E+2) ≈**	MaF3	20	**1.1336E+1 (2.08E+1)**	1.6183E+1 (2.97E+1) ≈
MaF4	3	**2.0512E+1 (1.15E+1)**	2.1773E+1 (1.14E+1) ≈	MaF4	20	7.3413E+5 (5.05E+5)	**6.1730E+5 (4.39E+5) ≈**
MaF5	3	**3.8697E-1 (3.82E-1)**	5.0943E-1 (5.64E-1) +	MaF5	20	1.6534E+5 (8.69E+3)	**1.6285E+5 (1.43E+4) ≈**
MaF6	3	**1.2276E-2 (1.44E-3)**	3.0348E-2 (6.13E-4) +	MaF6	20	**3.9075E-2 (3.36E-3)**	4.7283E-2 (3.62E-3) +
MaF7	3	**3.3199E-1 (3.64E-2)**	3.4570E-1 (3.38E-2) ≈	MaF7	20	7.8575E+0 (1.54E+0)	**7.7143E+0 (1.88E+0) ≈**
MaF8	3	**3.0672E-1 (3.86E-1)**	3.1694E-1 (2.56E-1) ≈	MaF8	20	**1.6790E+0 (1.58E-1)**	1.8738E+0 (3.53E-1) +
MaF9	3	**1.7344E-1 (1.06E-1)**	4.0007E-1 (4.65E-1) +	MaF9	20	**9.7734E+0 (7.35E+0)**	1.5434E+1 (4.88E+0) +
MaF10	3	6.9509E-1 (1.23E-1)	**6.3483E-1 (9.46E-2) ≈**	MaF10	20	**4.8222E+0 (2.85E-1)**	4.8872E+0 (1.25E-1) ≈
MaF11	3	1.8118E-1 (6.44E-3)	**1.7690E-1 (6.92E-3) -**	MaF11	20	**3.5614E+0 (2.07E-1)**	3.7082E+0 (2.22E-1) +
MaF12	3	2.4733E-1 (1.25E-2)	**2.3030E-1 (6.76E-3) -**	MaF12	20	**1.1138E+1 (4.94E-1)**	1.1716E+1 (4.38E-1) +
MaF13	3	8.7601E-2 (5.29E-3)	**7.8221E-2 (6.22E-3) -**	MaF13	20	1.9357E+0 (1.64E-1)	**1.8356E+0 (3.28E-1) ≈**
MaF14	3	1.9886E+0 (9.70E-1)	**1.3828E+0 (4.59E-1) -**	MaF14	20	2.1976E+0 (1.54E+0)	**2.0573E+0 (1.16E+0) ≈**
MaF15	3	3.4496E-1 (3.69E-2)	**3.0413E-1 (2.63E-2) -**	MaF15	20	**1.3726E+1 (1.80E+0)**	1.7380E+1 (2.53E+0) +
+/≈/-		NA	3/5/7	+/≈/-		NA	8/7/0

TABLE VII
COMPARISONS OF IGD RESULTS AMONG MTMOEA VARIANTS WITH DIFFERENT NEIGHBORHOOD SIZE FOR KNOWLEDGE TRANSFER

Problem	MTMOEA(T=⌊0.05×N⌋)	MTMOEA(T=1)	MTMOEA(T=⌊0.1×N⌋)	MTMOEA(T=⌊0.15×N⌋)	MTMOEA(T=⌊0.2×N⌋)
MaF1	**6.4836E-2 (2.14E-3)**	6.6159E-2 (2.46E-3) +	6.5278E-2 (2.41E-3) ≈	6.6223E-2 (2.46E-3) +	6.5729E-2 (2.60E-3) ≈
MaF2	**5.4559E-2 (3.03E-3)**	5.4730E-2 (3.12E-3) ≈	5.5441E-2 (2.24E-3) ≈	5.5523E-2 (2.51E-3) ≈	5.5517E-2 (2.43E-3) ≈
MaF3	**6.9995E+1 (7.99E+1)**	1.3076E+2 (1.64E+2) +	9.9064E+1 (1.62E+2) ≈	8.4812E+1 (1.15E+2) ≈	1.0563E+2 (1.46E+2) ≈
MaF4	2.0512E+1 (1.15E+1)	2.1636E+1 (1.55E+1) ≈	**1.8887E+1 (1.45E+1) ≈**	2.0700E+1 (1.54E+1) ≈	2.0190E+1 (1.23E+1) ≈
MaF5	3.8697E-1 (3.82E-1)	4.9840E-1 (8.89E-1) +	4.6732E-1 (4.75E-1) ≈	**3.2807E-1 (2.22E-1) ≈**	3.4765E-1 (3.17E-1) ≈
MaF6	1.2276E-2 (1.44E-3)	1.1976E-2 (9.28E-4) ≈	1.2453E-2 (1.26E-3) ≈	1.2302E-2 (1.23E-3) ≈	**1.1702E-2 (1.41E-3) ≈**
MaF7	3.3199E-1 (3.64E-2)	3.2709E-1 (2.73E-2) ≈	3.2810E-1 (3.19E-2) ≈	3.3400E-1 (3.51E-2) ≈	3.3040E-1 (3.30E-2) ≈
MaF8	3.0672E-1 (3.86E-1)	2.4938E-1 (2.44E-1) ≈	2.8319E-1 (3.09E-1) ≈	**2.1884E-1 (1.12E-1) ≈**	2.4096E-1 (1.65E-1) ≈
MaF9	1.7344E-1 (1.06E-1)	**1.3668E-1 (6.04E-2) ≈**	1.8660E-1 (1.69E-1) ≈	2.1624E-1 (2.52E-1) ≈	1.6765E-1 (1.19E-1) ≈
MaF10	6.9509E-1 (1.23E-1)	6.8033E-1 (9.45E-2) ≈	6.8489E-1 (1.21E-1) ≈	**6.5581E-1 (9.65E-2) ≈**	6.6749E-1 (9.49E-2) ≈
MaF11	1.8118E-1 (6.44E-3)	1.8138E-1 (6.30E-3) ≈	1.8244E-1 (5.45E-3) ≈	**1.7877E-1 (6.56E-3) ≈**	1.8058E-1 (8.00E-3) ≈
MaF12	2.4733E-1 (1.25E-2)	2.4392E-1 (6.06E-3) ≈	**2.4373E-1 (8.85E-3) ≈**	2.4543E-1 (8.52E-3) ≈	2.4752E-1 (7.92E-3) ≈
MaF13	**8.7601E-2 (5.29E-3)**	9.0581E-2 (6.05E-3) ≈	8.9884E-2 (6.68E-3) ≈	8.9429E-2 (5.34E-3) ≈	8.8804E-2 (6.71E-3) ≈
MaF14	1.9886E+0 (9.70E-1)	1.9618E+0 (6.49E-1) ≈	2.2782E+0 (9.41E-1) ≈	**1.8043E+0 (7.63E-1) ≈**	2.3048E+0 (9.03E-1) +
MaF15	3.4496E-1 (3.69E-2)	3.4655E-1 (4.64E-2) ≈	3.4273E-1 (3.68E-2) ≈	**3.4090E-1 (2.49E-2) ≈**	3.4506E-1 (2.72E-2) ≈
+/≈/-	NA	3/12/0	0/15/0	1/14/0	1/14/0

MTMOEA(λ=0.95) outperforms MTMOEA(λ=0.80) and MTMOEA(λ=1.00) on 6 and 7 problems, but only worse on 3 and 1 problems, respectively. The above results may be due to that during the evolutionary search, the PF approximated based on the information of current-best solutions is still higher than the real PF in the objective space, and therefore a scale factor slightly smaller than 1 can help pull the approximated PF to be closer to the real PF. However, if the scale factor is too small (e.g., λ=0.80), the estimated PF can be wrong after being scaled. In addition, as the λ=0.95 is the best among the five different settings, the λ=0.95 is adopted in MTMOEA in this paper.

F. Influence of the Target Points Selection

The estimated target points in the MTMOEA are obtained based on the potential point via the Kmeans clustering technique. Therefore, this part further analyzes the effect of clustering in obtaining estimated target points. That is, the MTMOEA is compared with its variant that does not use the Kmeans clustering, which is denoted as MTMOEA-w/o-C. As the number of estimated target points should be the same as N and the MTMOEA-w/o-C does not have a clustering technique to obtain N target points based on L potential points, the L is also set as N in MTMOEA-w/o-C, so that potential points can be directly selected as target points without clustering. The comparison results between MTMOEA and MTMOEA-w/o-C on 3- and 20-objective problems are provided in Table VI. As can be seen, MTMOEA has a worse performance than MTMOEA-w/o-C on 3-objective problems generally but have a much better performance on 20-objective problems. In particular, MTMOEA has worse results than MTMOEA-w/o-C on 7 of the 15 3-objective problems, while only having better results on 3 problems. But on the 15 20-objective problems, the MTMOEA performs significantly better than MTMOEA-w/o-C on 8 problems, similarly on 7 problems, and worse on none problems. This may be due to that when the number of objectives is small, the objective space is not large and the clustering technique cannot distinguish similar points into different clusters well, while for problems with a large number of objectives, the clustering technique can classify the potential points into corresponding groups and obtain the representative target point efficiently. Therefore, the clustering technique is useful.

G. Influence of the Neighborhood Size for Knowledge Transfer

In AIKT, the neighborhood size T can influence the knowledge transfer among individuals for different tasks. Therefore, this part analyzes the influence of T based on the comparisons between the original MTMOEA and its variants with different T. That is, the MTMOEA(T=⌊0.05×N⌋) is compared with the variants MTMOEA(T=1),

MTMOEA(T=⌊0.1×N⌋), MTMOEA(T=⌊0.15×N⌋), and MTMOEA(T=⌊0.2×N⌋), where MTMOEA(T=1) means that only the nearest individual will be selected for knowledge transfer. The comparison results are provided in Table VII. As shown in Table VII, the MTMOEA(T=⌊0.05×N⌋) have similar performance to MTMOEA(T=⌊0.1×N⌋) and slightly better performance than MTMOEA(T=⌊0.15×N⌋) and MTMOEA(T=⌊0.2×N⌋). To be more specific, MTMOEA(T=⌊0.05×N⌋) has similar performance on 15, 14, and 14 problems when compared with MTMOEA(T=⌊0.1×N⌋), MTMOEA(T=⌊0.15×N⌋), and MTMOEA(T=⌊0.2×N⌋), respectively. This indicates that the MTMOEA(T=⌊0.05×N⌋) is not that sensitive to the setting of T. Moreover, Table VII also shows that the MTMOEA(T=⌊0.05×N⌋) outperforms MTMOEA(T=1) on 3 problems, but does not have worse performance on any problems. This may be due to that a proper large T can help improve the knowledge transfer among individuals that are good at different tasks, and thus can enhance the optimization results. Note that the MTMOEA(T=⌊0.05×N⌋) obtains slightly better results than the rest variants, the T=⌊0.05×N⌋ is adopted in this paper.

H. Running Time Analysis

This part investigates the computational complexity of the proposed algorithm by the comparison and analysis of running time. The average wall-clock times of all seven algorithms over 30 independent runs are reported in Table S.III of the supplementary material. As shown in Table S.III, the MTMOEA has a smaller average ranking than the MOEA/D-DES, i.e., less computationally complex than MOEA/D-DES. Moreover, the MTMOEA can also run faster than KnEA and Mo4Ma on some problems, e.g., the MaF8 with 5 or 10 objectives. In addition, although the MTMOEA has a longer time cost than some fast algorithms such as the NSGA-III and NSGA-II-SDR, their gaps in time cost are only in seconds. Therefore, the MTMOEA is not much computationally complex. Note that previous experiments have shown the promising optimization ability of MTMOEA, slight additional time cost of MTMOEA could be acceptable.

V. Conclusion

This paper proposes a new multiobjective optimization method via MTO. For this aim, this paper shows the relationship between MOP and MTOP and how a MOP can be transformed into a MTOP mathematically and theoretically. Then, the TPE strategy has been proposed to help automatically transform the MOP into a MTOP more accurately. Moreover, this paper further proposes the MTMO framework with the AIKT strategy to solve the transformed MTOP more efficiently. In addition, based on the above, the complete algorithm MTMOEA is finally developed as an example to solve the MOP as a MTOP. To investigate the proposed MTMOEA, extensive experiments have been conducted on widely-used MOPs with 3 to 20 objectives, where some state-of-the-art MOP algorithms have also been used as competitors. The experimental results have shown the superior performance of the proposed MTMOEA. This shows that solving MOP as a MTOP is a promising direction for tackling MOP efficiently.

For future work, the proposed algorithm will be further extended to solve more difficult and complex MOPs with different properties. As the proposed algorithm has some tunable parameters (e.g., the number of potential points for approximating PF and the neighborhood size for knowledge transfer) that although have been investigated in the experimental part, we could use learning methods (e.g., learning-aided methods [71][72] and adaptive methods [73]-[76]) in the future to configure the parameters more automatically and adaptively according to the target problem. Furthermore, as solving the MOP as a MTOP is a generic idea, further exploration of MTOP algorithms and knowledge transfer methods is worth studying to solve complex MOPs more efficiently. Besides, the proposed MTMOEA will be further studied in distributed evnviroment [77]-[79] and applied to real-world MOPs. Another very interesting observation is that we also found that treating MOP as a multimodal optimization problem is also a promising way to solve MOP via multimodal optimization algorithm [80]. Therefore, we think that multiobjective optimization, multitask optimization, and multimodal optimization are connected with each other, leading to an interesting research direction of uniform optimization.

References

[1] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization-A new frontier in evolutionary computation research", *IEEE Comput. Intell. Mag.*, vol. 16, no. 1, pp. 22-33, Feb. 2021.

[2] Z. H. Zhan, L. Shi, K. C. Tan, and J. Zhang, "A survey on evolutionary computation for complex continuous optimization," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 59–110, Jan. 2022.

[3] J. Y. Li, Z. H. Zhan, and J. Zhang, "Evolutionary computation for expensive optimization: A survey," *Mach. Intell. Res.*, vol. 19, no. 1, pp. 3–23, 2022.

[4] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.

[5] 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.

[6] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," Eidgenössische Technische Hochschule Zürich, Institut für Technische Informatik und Kommunikationsnetze (TIK), TIK-Rep. 103, 2001.

[7] C. A. Coello Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. Congr. Evol. Comput.*, 2002, pp. 1051–1056.

[8] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[9] X. Ma, Y. Yu, X. Li, Y. Qi, and Z. Zhu, "A survey of weight vector adjustment methods for decomposition-based multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 24, no. 4, pp. 634–649, 2020.

[10] J. G. Falcón-Cardona and C. A. C. Coello, "Indicator-based multi-objective evolutionary algorithms: A comprehensive survey," *ACM Comput. Surv.*, vol. 53, no. 2, pp. 1–35, 2020.

[11] K. Shang, H. Ishibuchi, L. He, and L. M. Pang, "A survey on the hypervolume indicator in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 25, no. 1, pp. 1–20, 2021.

[12] Z. H. Zhan, J. J. Li, J. N. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.

[13] L. Miguel Antonio and C. A. C. Coello, "Coevolutionary multiobjective evolutionary algorithms: Survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 851–865, 2018.

[14] J. Yi, J. Bai, H. He, J. Peng, and D. Tang, "Ar-MOEA: A novel preference-based dominance relation for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 788–802,

2019.
[15] K. Deb and S. Gupta, "Understanding knee points in bicriteria problems and their implications as preferred solution principles," *Eng. Optim.*, vol. 43, no. 11, pp. 1175–1204, 2011.
[16] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.
[17] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many objective optimization algorithm based on dominance and decomposition," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 694–716, Oct. 2015.
[18] K. K. Bali, Y. S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 69-83, 2020.
[19] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y. S. Ong, K. Tan, and A. K. Qin, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3457–3470, Sept. 2019.
[20] J. Y. Li, Z. H. Zhan, K. C. Tan, and J. Zhang, "A meta-knowledge transfer-based differential evolution for multitask optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 4, pp. 719–734, Aug. 2022.
[21] A. Gupta and Y. S. Ong, "Back to the Roots: Multi-X evolutionary computation," *Cognit. Comput.*, vol. 11, no. 1, pp. 1–17, 2019.
[22] A. Gupta, L. Zhou, Y. S. Ong, Z. Chen, and Y. Hou, "Half a dozen real-world applications of evolutionary multitasking, and more," *IEEE Comput. Intell. Mag.*, vol. 17, no. 2, pp. 49–66, 2022.
[23] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Trans. Cybern.*, vol. 47, no. 7, pp. 1652–1665, Jul. 2017.
[24] K. J. Du, J. Y. Li, H. Wang, and J. Zhang, "Multi-objective multi-criteria evolutionary algorithm for multi-objective multi-task optimization," *Complex Intell. Syst.*, vol. 9, no. 2, pp. 1211–1228, Apr. 2023.
[25] Y. Feng, L. Feng, S. Kwong, and K. C. Tan, "A multivariation multifactorial evolutionary algorithm for large-scale multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 248–262, Apr. 2022.
[26] K. Qiao *et al.*, "Dynamic auxiliary task-based evolutionary multitasking for constrained multi-objective optimization," *IEEE Trans. Evol. Comput.*, 2022, DOI: 10.1109/TEVC.2022.3175065.
[27] R. Cheng *et al.*, "A benchmark test suite for evolutionary many-objective optimization," *Complex Intell. Syst.*, vol. 3, no. 1, pp. 67–81, 2017.
[28] R. Allmendinger, A. Jaszkiewicz, A. Liefooghe, and C. Tammer, "What if we increase the number of objectives? Theoretical and empirical implications for many-objective combinatorial optimization," *Comput. Oper. Res.*, vol. 145, no. 105857, pp. 1–17, 2022.
[29] A. Jaszkiewicz, "Many-objective Pareto local search," *Eur. J. Oper. Res.*, vol. 271, no. 3, pp. 1001–1013, Dec. 2018.
[30] M. Aghabeig and A. Jaszkiewicz, "Experimental analysis of design elements of scalarizing function-based multiobjective evolutionary algorithms," *Soft Comput.*, vol. 23, no. 21, pp. 10769–10780, 2019.
[31] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 263–282, Sep. 2002.
[32] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.
[33] Y. Tian, R. Cheng, X. Zhang, Y. Su, and Y. Jin, "A strengthened dominance relation considering convergence and diversity for evolutionary many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 331-345, Apr. 2019.
[34] Q. Zhu, Q. Zhang, and Q. Lin, "A constrained multiobjective evolutionary algorithm with detect-and-escape strategy," *IEEE Trans. Evol. Comput.*, vol. 24, no. 5, pp. 938–947, Oct. 2020.
[35] H. Chen, G. Wu, W. Pedrycz, P. N. Suganthan, L. Xing, and X. Zhu, "An adaptive resource allocation strategy for objective space partition based multiobjective optimization," *IEEE Trans. Syst., Man, Cybern., Syst.*, 2019, DOI: 10.1109/TSMC.2019.2898456.
[36] Y. Yuan, H. Xu, B. Wang, B. Zhang, and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 180–198, Apr. 2016.
[37] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 773–791, 2016.
[38] H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, "Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 169–190, Apr. 2017.
[39] Y. Zhou, Y. Xiang, Z. Chen, J. He, and J. Wang, "A scalar projection and angle-based evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2073–2084, Jun. 2019.
[40] Y. Sun, G. G. Yen, and Z. Yi, "IGD indicator-based evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 173–187, Apr. 2019.
[41] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, Mar. 2011.
[42] B. Li, K. Tang, J. Li, and X. Yao, "Stochastic ranking algorithm for many-objective optimization based on multiple indicators," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 924–938, Dec. 2016.
[43] Z. G. Chen *et al.*, "Multiobjective cloud workflow scheduling: A multiple populations ant colony system approach," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 2912–2926, 2019.
[44] X. Zhang, Z.-H. Zhan, W. Fang, P. Qian, and J. Zhang, "Multipopulation ant colony system with knowledge-based local searches for multiobjective supply chain configuration," *IEEE Trans. Evol. Comput.*, vol. 26, no. 3, pp. 512–526, Jun. 2022.
[45] S. C. Liu, Z. G. Chen, Z. H. Zhan, S. W. Jeon, S. Kwong, and J. Zhang, "Many-objective job-shop scheduling: A multiple populations for multiple objectives-based genetic algorithm approach," *IEEE Trans. Cybern.*, vol. 53, no. 3, pp. 1460–1474, Mar. 2023.
[46] S. Z. Zhou, Z. H. Zhan, Z. G. Chen, S. Kwong, and J. Zhang, "A multi-objective ant colony system algorithm for airline crew rostering problem with fairness and satisfaction," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6784–6798, 2020.
[47] J. Y. Li *et al.*, "A multipopulation multiobjective ant colony system considering travel and prevention costs for vehicle routing in covid-19-like epidemics," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25062–25076, Dec. 2022.
[48] X. F. Liu, Z. H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 587–602, Aug. 2019.
[49] Q. T. Yang, Z. H. Zhan, S. Kwong, and J. Zhang, "Multiple populations for multiple objectives framework with bias sorting for many-objective optimization," *IEEE Trans. Evol. Comput.*, 2022, DOI: 10.1109/TEVC.2022.3212058.
[50] Q. Lin *et al.*, "Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 32–46, Feb. 2018.
[51] J. L. Matos and A. Britto, "Multi-swarm algorithm based on archiving and topologies for many-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2017, pp. 1877–1884.
[52] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 761–776, Dec. 2015.
[53] G. Yu, Y. Jin, and M. Olhofer, "A multiobjective evolutionary algorithm for finding knee regions using two localized dominance relationships," *IEEE Trans. Evol. Comput.*, vol. 25, no. 1, pp. 145–158, Feb. 2021.
[54] S. C. Liu, Z. H. Zhan, K. C. Tan, and J. Zhang, "A multiobjective framework for many-objective optimization," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 13654–13668, Dec. 2022.
[55] M. Gao, J. Y. Li, C. H. Chen, Y. Li, J. Zhang, and Z. H. Zhan, "Enhanced multi-task learning and knowledge graph-based recommender system," *IEEE Trans. Knowl. Data Eng.*, 2023,DOI: 10.1109/TKDE.2023.3251897.
[56] J. Y. Li, K. J. Du, Z. H. Zhan, H. Wang, and J. Zhang, "Multi-criteria differential evolution: Treating multitask optimization as multi-criteria optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2021, pp. 183–184.
[57] M. Gong, Z. Tang, H. Li, and J. Zhang, "Evolutionary multitasking with dynamic resource allocating strategy," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 858-869, Oct. 2019.
[58] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multi-tasking for evolutionary optimization of expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 44-58, Feb. 2019.
[59] L. Zhou *et al.*, "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2563-2576, May 2021.
[60] J. Zhong, L. Feng, W. Cai, and Y. S. Ong, "Multifactorial genetic programming for symbolic regression problems," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 11, pp. 4492-4505, Nov. 2020.
[61] S. H. Wu, Z. H. Zhan, K. C. Tan, and J. Zhang, "Transferable adaptive differential evolution for many-task optimization," *IEEE Trans. Cybern.*,

2023, DOI: 10.1109/TCYB.2023.3234969.
[62] Y. Jiang, Z. H. Zhan, K. C. Tan, and J. Zhang, "A bi-objective knowledge transfer framework for evolutionary many-task optimization," *IEEE Trans. Evol. Comput.*, 2022, DOI: 10.1109/TEVC.2022.3210783.
[63] S. H. Wu, Z. H. Zhan, K. C. Tan, and J. Zhang, "Orthogonal transfer for multitask optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 1, pp. 185–200, Feb. 2023.
[64] Y. Jiang, Z. H. Zhan, K. C. Tan, and J. Zhang, "Block-level knowledge transfer for evolutionary multitask optimization," *IEEE Trans. Cybern.*, DOI: 10.1109/TCYB.2023.3273625, 2023.
[65] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 609–622, 2018.
[66] 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, 2017.
[67] 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.
[68] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," *Evol. Multiobjective Optim.*, pp. 105–145, 2005.
[69] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, 2006.
[70] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, 1998.
[71] Z. H. Zhan, J. Y. Li, S. Kwong, and J. Zhang, "Learning-aided evolution for optimization," *IEEE Trans. Evol. Comput.*, 2022, DOI: 10.1109/TEVC.2022.3232776.
[72] A. Gupta, Y. S. Ong, and L. Feng, "Insights on transfer optimization: Because experience is the best teacher," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 2, no. 1, pp. 51–64, 2017.
[73] Z. H. Zhan, Z. J. Wang, H. Jin, and J. Zhang, "Adaptive distributed differential evolution," *IEEE Trans. Cybern.*, vol. 50, no. 11, pp. 4633–4647, Nov. 2020.
[74] J. Y. Li, K. J. Du, Z. H. Zhan, H. Wang, and J. Zhang, "Distributed differential evolution with adaptive resource allocation," *IEEE Trans. Cybern.*, vol. 53, no. 5, pp. 2791–2804, May 2023.
[75] S. H. Wu, Z. H. Zhan, and J. Zhang, "SAFE: Scale-adaptive fitness evaluation method for expensive optimization problems," *IEEE Trans. Evol. Comput.*, vol. 25, no. 3, pp. 478–491, 2021.
[76] Y. Q. Wang, J. Y. Li, C. H. Chen, J. Zhang, and Z. H. Zhan, "Scale adaptive fitness evaluation-based particle swarm optimisation for hyperparameter and architecture optimisation in neural networks and deep learning," *CAAI Trans. Intell. Technol.*, 2022, DOI: 10.1049/cit2.12106.
[77] Z. H. Zhan, *et al.*, "Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version," *IEEE Trans. Parallel and Distributed Systems*, vol. 28, no. 3, pp. 704-716, March. 2017.
[78] Y. Guo, J. Y. Li, and Z. H. Zhan, "Efficient hyperparameter optimization for convolution neural networks in deep learning: A distributed particle swarm optimization approach," *Cybern. Syst.*, vol. 52, no. 1, pp. 36–57, 2020.
[79] J. Y. Li, Z. H. Zhan, R. D. Liu, C. Wang, S. Kwong and J. Zhang, "Generation-level parallelism for evolutionary computation: A pipeline-based parallel particle swarm optimization," *IEEE Trans. Cybern.*, vol. 51, no. 10, pp. 4848-4859, Oct. 2021.
[80] Z. G. Chen, Z. H. Zhan, and J. Zhang, "Bridge connecting multiobjective and multimodal: A new approach for multiobjective optimization via multimodal optimization," in *Proc. IEEE Int. Conf. Information, Cybern., and Comput. Social Syst.*, 2020, pp. 463-468.

Jian-Yu Li (Member, IEEE) received the Bachelor's degree and the Ph. D. degree in Computer Science and Technology from the South China University of Technology, China, in 2018 and 2022, respectively.

His research interests mainly include computational intelligence, data-driven optimization, machine learning including deep learning, and their applications in real-world problems, and in environments of distributed computing and big data.

Dr. Li has been invited as a reviewer of the *IEEE Transactions on Evolutionary Computation* and the *Neurocomputing*.

Zhi-Hui Zhan (Senior Member, IEEE) received the Bachelor's degree and the Ph. D. degree in Computer Science from the Sun Yat-Sen University, Guangzhou China, in 2007 and 2013, respectively.

He is currently the Changjiang Scholar Young Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. His current research interests include evolutionary computation, swarm intelligence, and their applications in real-world problems and in environments of cloud computing and big data.

Dr. Zhan was a recipient of the IEEE Computational Intelligence Society (CIS) Outstanding Early Career Award in 2021, the Outstanding Youth Science Foundation from National Natural Science Foundations of China (NSFC) in 2018, and the Wu Wen-Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017. He is one of the World's Top 2% Scientists for both Career-Long Impact and Year Impact in Artificial Intelligence and one of the Highly Cited Chinese Researchers in Computer Science. He is currently the Chair of Membership Development Committee in IEEE Guangzhou Section and the Vice-Chair of IEEE CIS Guangzhou Chapter. He is currently an Associate Editor of the *IEEE Transactions on Evolutionary Computation*, the *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, the *Neurocomputing*, the *Memetic Computing*, and the *Machine Intelligence Research*.

Yun Li (Fellow, IEEE) received the B.S. degree from Sichuan University, Chengdu, China, in 1984, the M.E. degree from University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1987, and the Ph.D. degree from University of Strathclyde, Glasgow, U.K., in 1990.

In 1989, he was an Intelligent Control Engineer with the U.K. National Engineering Laboratory, Glasgow. In 1990, he was a Postdoctoral Research Engineer with Industrial Systems and Control Ltd, Glasgow. From 1991 to 2018, he was an Intelligent Systems Lecturer, Senior Lecturer, and Professor with University of Glasgow, Scotland, and served as Founding Director of University of Glasgow Singapore, Singapore. He later served as the Founding Director of Dongguan Industry 4.0 Artificial Intelligence Laboratory, Dongguan, China, and of i4AI Ltd, London, U.K. He is currently a Changjiang Chair Professor at Shenzhen Institute for Advanced Study, UESTC.

Prof. Li is interested in the next generation, explainable artificial intelligence (AI) and has supervised over 30 PhD students focusing on computational AI and its engineering/industrial applications since 1991. He is a U.K. Chartered Engineer and is currently an Associate Editor of *IEEE Transactions on Emerging Topics in Computational Intelligence*. He has published 300 papers, and one of them since publication in the *IEEE Transactions on Control System Technology* in 2005 has been its most popular article almost every month.

Jun Zhang (Fellow, IEEE) obtained his PhD degree in Electrical Engineering from the City University of Hong Kong in 2002.

He is a professor with the School of Electrical and Engineering, Hanyang University ERICA, Ansan 15588, South Korea, and He holds a Distinguished Professorship in Nankai University. Prof. Zhang's research contributions span over 300 peer-reviewed publications, of which more than 180 appear in IEEE Transactions. His research interests include Computational Intelligence, cloud computing, Big data mining, and Power Electronic Circuits.

Professor Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and was appointed as a Cheung Kong Chair Professor in 2013 by the Ministry of Education, China. Presently, Prof. Zhang serves as an associate editor for both the *IEEE Transactions on Artificial Intelligence* and the *IEEE Transactions on Cybernetics*.