Micro Many-Objective Evolutionary Algorithm With Knowledge Transfer

Hu Peng, Zhongtian Luo, Tian Fang, and Qingfu Zhang, *Fellow, IEEE*

***Abstract*—Computational effectiveness and limited resources in evolutionary algorithms are interdependently handled during the working of low-power microprocessors for real-world problems, particularly in many-objective evolutionary algorithms (MaOEAs). In this respect, the balance between them will be broken by evolutionary algorithms with a normal-sized population, but which doesn't include a micro population. To tackle this issue, this paper proposes a micro many-objective evolutionary algorithm with knowledge transfer (μMaOEA). To address the oversight that knowledge is often not considered enough between niches, the knowledge-transfer strategy is proposed to bolster each unoptimized niche through optimizing adjacent niches, which enables niches to generate better individuals. Meanwhile, a two-stage mechanism based on fuzzy logic is designed to settle the conflict between convergence and diversity in many-objective optimization problems. Through efficient fuzzy logic decision-making, the mechanism maintains different properties of the population at different stages. Different MaOEAs and micro multi-objective evolutionary algorithms were compared on benchmark test problems DTLZ, MaF, and WFG, and the results showed that μMaOEA has an excellent performance. In addition, it also conducted simulation on two real-world problems, MPDMP and MLDMP, based on a low-power microprocessor. The results indicated the applicability of μMaOEA for low-power microprocessor optimization.**

***Index Terms*—Micro many-objective evolutionary algorithm, many-objective optimization problems, knowledge transfer, low-power microprocessors.**

I. Introduction

MANY-OBJECTIVE optimization problem (MaOP) can be mathematically defined as:

$$\begin{aligned} &\min F(x) = [f_1(x), f_2(x), \ldots, f_M(x)], \quad M > 3 \\ &\text{s.t.} \quad x \in \Omega \end{aligned} \tag{1}$$

Received 10 May 2024; accepted 20 July 2024. Date of publication 9 September 2024; date of current version 23 January 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62266024 and in part by the Science and Technology Plan Projects of Jiangxi Provincial Education Department under Grant GJJ2201906. *(Corresponding author: Hu Peng.)*

Hu Peng is with the School of Computer and Big Data Science, Jiujiang University, Jiujiang 332005, China, and also with the Jiujiang Key Laboratory of Digital Technology, Jiujiang 332005, China (e-mail: hu_peng@whu.edu.cn).

Zhongtian Luo and Tian Fang are with the School of Computer and Big Data Science, Jiujiang University, Jiujiang 332005, China (e-mail: luo_zhongtian@aliyun.com).

Qingfu Zhang is with the Department of Computer Science, City University of Hong Kong, Hong Kong 999077, China (e-mail: qingfu.zhang@cityu.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TETCI.2024.3451309, provided by the authors.

Recommended for acceptance by Q. Lin.

Digital Object Identifier 10.1109/TETCI.2024.3451309

where $\Omega \subset \mathbb{R}^D$ is a D-dimensional decision space with decision variable vector $\mathbf{x} = (x_1, x_2, \ldots, x_D)$. $F : \Omega \to \Theta \subset \mathbb{R}^M$ is an M-dimensional objective vector $f_1(x), f_2(x), \ldots, f_M(x)$ mapped from the decision space Ω to the objective space Θ.

Given two different decision vectors $p, q \in \Omega$, p pareto dominates q (denoted as $p \prec q$), if and only if $f_i(p) \leq f_i(q)$ for every $i \in \{1, 2, \ldots, M\}$ and $f_j(p) < f_j(q)$ for at least one index $j \in \{1, 2, \ldots, M\}$, where M is the number of objectives. The $Pareto\ set$ is defined as $PS = \{p \in \Omega | \nexists\ q \in \Omega, q \prec p\}$. Accordingly, the Pareto optimal objective vector set is called the $Pareto\ front$, defined as $PF = \{F(x) |\ x \in PS\}$.

Since the 1990s, lots of researchers have devoted themselves to the research of evolutionary multi-objective optimization (EMO) with two and three objectives [1]. However, in real-world problems [2] such as micro robots distance optimization [3] and hybrid electric vehicle control [4], there are four or more optimization objectives, even up to 10 to 15 objectives. More objectives mean bigger challenges. In micro robots distance optimization, the robots are equipped with low-power microprocessors to balance the resources and performance. Low-power microprocessors have a lower configuration compared to mainstream computer platforms, which limits their computing resources and memory capability, making it difficult for multi-objective evolutionary algorithm (MOEA) with a normal-sized population to handle many objectives successfully.

Fortunately, Coello et al. [5] opened the door of micro multi-objective evolutionary algorithms (μMOEAs). In many MOEAs, the population size is usually greater than 100, whereas it is usually limited to less than 20 in μMOEAs. Therefore, μMOEAs can solve the problems encountered by MOEAs with a normal-sized population on low-power microprocessors in a simple and computationally efficient way. It is worth noting that although there has been some work on μMOEAs, there is only very little work on micro many-objective evolutionary algorithms.

The difficulties encountered by micro many-objective evolutionary algorithms exist in both μMOEAs and many-objective evolutionary algorithms (MaOEAs). On the one hand, due to the micro population size of μMOEAs, it is difficult to maintain diversity during the evolutionary process. On the other hand, there are many difficulties in MaOPs with the increasing number of objectives [6], [7], [8]. First, more objectives result in a sharp increase in the number of non-dominated solutions, which can severely deteriorate the selection pressure between the PF and solutions [9]. Second, as the size of the objective space increases, the conflict between convergence and diversity becomes more

aggravated [10], leading many diversity strategies (e.g., [10], [11]) to prefer selecting dominance solutions. Third, the distance between solutions may be very far in high-dimensional objective space. Therefore, recombination may be inefficient since the evolutionary operator often produces an offspring far away from its parents. Last, some performance indicators, especially hypervolume (HV) [12], are computationally expensive to calculate, which harms the development of indicators on MaOEAs. So MaOEA with a micro population becomes even more difficult to tackle the above issues.

Additionally, many works have proven that knowledge transfer has excellent information scheduling and optimization capability [13], [14], [15]. In order to solve the above difficulties, knowledge transfer is introduced into the algorithm proposed in this paper to address micro many-objective optimization. In this work, knowledge is all the efficient information possessed by niches that is beneficial for evolution progress. Knowledge enables niches to generate better individuals, and based on this, a knowledge-transfer strategy and fuzzy two-stage mechanism are proposed. The adjacent niches of an unoptimized niches are optimized with the knowledge-transfer strategy to transfer excellent knowledge to the sandwiched niches, ultimately through the knowledge gap between niches to achieve efficient optimization of the entire population. The fuzzy two-stage mechanism controls different stages through fuzzy logic, thereby achieving knowledge inheritance and maintenance. They enable algorithms to present excellent performance in micro many-objective optimization. Besides, it has shown good performance in the experiments and simulations. The main contributions of this paper can be summarized as follows.

- To better balance the computational effectiveness and limited resources in MaOPs, a micro population in many-objective optimization with knowledge transfer (μMaOEA) is developed. The experimental and simulation results both demonstrate its excellent performance.
- A knowledge-transfer strategy has been proposed to improve the evolutionary quality of the population, which achieves efficient knowledge transfer between niches by forming knowledge gaps.
- A two-stage mechanism based on fuzzy logic has been designed to allow inheritance and maintenance of knowledge through fuzzy logic to decide the stage, and each stage uses corresponding methods for environmental selection to enhance the different properties of the population.

The structure of the remaining paper is as follows. In Section II, related work and motivations in μMaOEA are introduced. Section III provides a detailed description of μMaOEA. Section IV is the experimental results and analysis. Section V is the simulation of μMaOEA and comparative algorithms in real-world problems. Section VI is the summary of this paper and prospects for future work.

II. Related Work and Motivation

This section provides an overview of preliminaries, including related work and the motivation behind the proposed algorithm.

A. Decomposition Framework and Methods

Zhang et al. introduced the decomposition idea into MOEA in 2007 and developed the promising MOEA/D [16]. MOEA/D has now become one of the most popular MOEAs. MOEA/D decomposes N weight vectors into N sub-problems, each of which is a single-objective optimization problem. Different weight vectors determine the evolutionary direction of different subproblems. Each subproblem is associated with a solution, and the solutions are recombined with their neighbors to obtain offspring. Moreover, the subproblems are optimized at the same time through the aggregation function to update the population, and only when the offspring has a better aggregate function value can they replace the parents. Through the above steps, a widespread population can be obtained on the true PF.

There are three main aggregation approaches in MOEAs [16], weighted sum approach (WS), tchebycheff approach (TCH), and penalty-based boundary intersection approach (PBI). As the penalty factor θ value changes, PBI can achieve the same aggregation effect as WS when $\theta = 0$ or TCH when $\theta = 1$. The definition of PBI is as:

$$\min g^{pbi}(x|w, z^*) = d_1 + \theta d_2 \tag{2}$$

where

$$d_1 = \frac{\|(F(x) - z^*)\, w\|}{\|w\|} \tag{3}$$

$$d_2 = \|F(x) - (z^* + d_1 w)\| \tag{4}$$

where w and $z^* = (z_1^*, \ldots, z_M^*)$ are the weight vector and the ideal point. d_1 is the projection distance of $(F(x) - z^*)$ on the weight vector w. d_2 is the distance between $F(x)$ and w. θ is a user-defined penalty factor that controls the bias towards convergence or diversity.

B. Micro Multi-Objective Evolutionary Algorithms

Micro multi-objective evolutionary algorithms (μMOEAs) are efficient for searching spaces with micro-sized populations [17], [18]. In many MOEAs, the population size is usually greater than 100, however, the population size of μMOEA is usually less than 20, which enables the algorithm to search for solutions at low computational costs. It is worth noting that there are problems of premature convergence and lack of diversity in the evolution process with micro populations [19]. Therefore, it is particularly important for μMOEAs to have effective strategies.

The existing μMOEAs can be roughly divided into three different classes. The first is μMOEAs based on the micro-GA framework, representing algorithms such as micro-GA [5] and μGA^2 [20]. For the first time in dealing with micro multi-objective optimization, they applied the advantages of genetic algorithm to the micro population and several elitisms to maintain the population and archive, which also provided a lot of theoretical support. The second class of μ MOEAs applies many traditional diversity strategies. Due to the tendency of μMOEAs to become premature during convergence, it is crucial to maintain diversity appropriately. Both AMGA [21] and

AMGA2 [22], as well as μMOGA [23], employ a crowded distance sorting method for environmental selection. The third class introduces decomposition methods into μMOEAs. The decomposition method not only maintains diversity but also reduces the time complexity of μMOEAs, which is helpful for the application of μMOEAs. In many real-world problems, normal-sized algorithms may not be feasible, while μMOEAs can solve them well. Peng et al. proposed μMMABC [24] for solving microgrid energy optimization. In addition, they also introduced the MOEA/D framework into μMOEAs, proposed μMOEA [17], and simulated the semi-autogenous grinding optimization problem on embedded processors. Beyond the three classes, there are also some μMOEAs that use other methods for optimization, such as fuzzy inference systems and hybrid technologies, which have been applied in μFAME [25] and μMOSM [19], respectively.

With the development of μMOEA, it has also been applied to many real-world problems. J.Mendoza et al. used a μMOEA to solve the multi-objective location of automatic voltage regulators in a radial distribution network [26]. Liu et al. developed a μMOEA to deal with the landing position of astronauts in [27], so that engineers can design airdrop protection devices to ensure the safety of astronauts. There are also microgrid energy optimization [24] and semi-autogenous grinding optimization problem based on embedded processors [18] mentioned in the previous paragraph. Besides, the simulation of low-power microprocessors is also presented in Section V.

C. Motivation

The balance between computational effectiveness and limited resources needs to be considered in many real-world optimization problems, such as micro robots [3], so economical and efficient low-power microprocessors are often applied in them [18]. However, low-power microprocessors run MaOEAs [28] with a normal-sized population may terminate programs due to limited computing resources and memory capability. This issue makes it difficult for some MaOPs to be solved.

Based on the analysis in Section II-B, it is necessary to focus on diversity strategies in μMOEAs, which will increase the difficulty in solving real-world problems with limited computing resources and memory capability. Therefore, the proposed algorithm uses a decomposition method is a wise choice. On the one hand, diversity is maintained through uniformly distributed weight vectors, while on the other hand, niches are obtained by dividing the objective space with weight vectors.

There are currently many MaOEAs that use an entire population to search for space due to their efficiency [29], [30]. However, it is worth noting that this requires a lot of resources and does not fully consider the effective information between niches. In fact, cooperative evolution among niches obtained by dividing a population based on their knowledge can enhance the evolutionary quality of the population. In this work, knowledge is defined as all the information possessed by a niche that is beneficial for evolution, enabling niches to produce more outstanding individuals. Once a high-quality solution is found during the evolutionary process, its excellent genetic information will spread to the entire population [16]. Therefore, it is important to efficiently utilize knowledge between niches for population evolution. It is worth noting that optimized adjacent niches will result in a knowledge gap between them, meaning that the optimized niches have better knowledge than the unoptimized ones. In the decomposition method, the parent individuals are selected from the neighborhood, which will influence the quality of offspring individuals [31]. Two adjacent niches of an unoptimized niche have been optimized, and then the unoptimized niche itself is optimized. In this way, adjacent niches of unoptimized niche have better knowledge. Subsequently, by optimizing the unoptimized niche, better knowledge from adjacent niches can be transferred to the unoptimized niche. The optimized niche after receiving excellent knowledge will produce better individuals. After completing multiple knowledge transfers by repeating the above operations, each niche is efficiently optimized many times to achieve the optimization of the entire population. Moreover, this idea is also practical for low-power microprocessors with limited computing resources and memory capability, as it only optimizes one niche at a time.

Convergence and diversity are often conflicting in MaOPs with a large search space [7], which means that individuals with good convergence often have poor diversity, and vice versa. Therefore, it is difficult for a MaOEA to maintain convergence and diversity within a population at the same time. It is proposed in [32] that if convergence and diversity are handled at different stages, they can be well maintained in the population. Inspired by this idea, a two-stage mechanism based on fuzzy logic control is designed in this paper. In the first stage, local search based techniques are used to enhance convergence, and the shift-based density estimation [33] is used to maintain both convergence and diversity to avoid falling into local optimum in the second stage. Through the above mechanism, not only can the convergence of the population be well maintained, but diversity is also the same.

III. The Proposed $\mu MaOEA$

In this section, a strategy based on knowledge transfer is proposed to improve the quality of population evolution, which achieves efficient knowledge transfer between niches by forming knowledge gaps. Meanwhile, a two-stage mechanism based on fuzzy control is designed to enhance selection pressure, which enables the algorithm to use different approaches according to the features of different stages. The following subsections provide a detailed description of μMaOEA.

A. Knowledge-Transfer Strategy

A knowledge-transfer strategy proposed in this paper aims to improve the population evolution quality by efficiently utilizing the knowledge gap between niches. It is suggested by integrating the ideas from knowledge transfer and merge sort, which not only endow the strategy with self-learning of knowledge transfer, but also equip it with low time complexity and divide-and-conquer approach from merge sort. As the efficient information scheduling and optimization capability of knowledge transfer were validated in [13], [14], [15], it has been recognized as

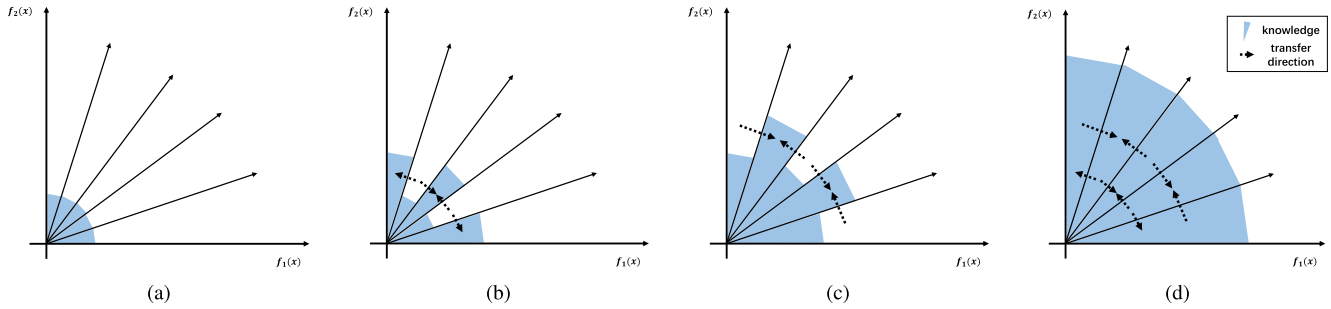


Fig. 1. An illustration of knowledge-transfer strategy. (a) shows the initial knowledge contained in merge blocks. (b) and (c) show how knowledge is transferred during the knowledge-transfer strategy, including the direction of knowledge transfer and the capability of knowledge in each merge block. (d) shows the knowledge contained in each merge block after the knowledge-transfer strategy is finished.

an effective way to address challenges in EMO. In this work, knowledge is defined as all the information that is beneficial to evolution in the niches, enabling the niches with excellent knowledge to generate excellent individuals.

Many MaOEAs evolve with an entire population, which leads to the neglect of some useful information between niches. The increase in computing burden caused by optimizing an entire population is not what μMOEAs hope to happen. If the population is divided into multiple niches and then processed separately, it will improve the feasibility of the algorithm on low-power microprocessors with limited computing resources and memory capability. In addition, if the effective information between the niches is utilized to promote the optimization, it will enhance the evolutionary quality of the entire population.

A merge controller (detailed in Section III-A-1)) is proposed to divide a population, and the resulting niches are called merge blocks. The initial population is randomly generated, so the initial knowledge of each merge block is assumed to be the same (Fig. 1(a)). As is well known, once high-quality solutions in evolution are found, excellent genetic information will quickly spread to the entire population. If the adjacent merge blocks of an unoptimized merge block have been optimized, they will contain better knowledge than the unoptimized merge block, then a knowledge gap between merge blocks is formed in this way. Under the MOEA/D framework, the parent individuals come from the neighborhood, whose quality will affect the quality of the offspring. Therefore, the sandwiched unoptimized merge blocks are optimized to utilize the knowledge gap, they will select neighboring individuals as parents to generate better offspring. As shown in Fig. 2, according to the division results of the merge controller, neighboring individuals located in the same merge block must come from adjacent optimized merge blocks due to the size of the neighborhood and merge block being set to be the same in this paper, which means that the parents contain excellent knowledge. At this point, the unoptimized merge block has been optimized, and the knowledge in the adjacent optimized merge blocks will be transferred to it. As shown in Fig. 1(b) and 1(c), the above operations are repeated in this way to transfer excellent knowledge from adjacent merge blocks to the sandwiched unoptimized merge blocks. The efficient evolution between niches enables the entire population to ultimately possess excellent knowledge, as shown in Fig. 1(d).

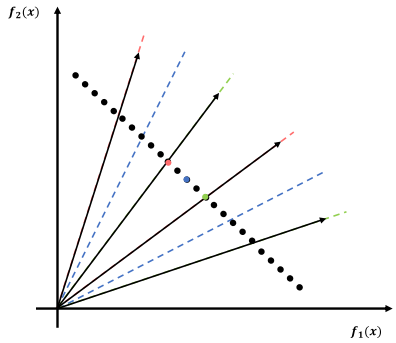


Fig. 2. An illustration of neighborhood relationships between merge blocks. Merge controller accurately controls the number of individuals in merge blocks, ensuring that neighboring individuals from one merge block exist in adjacent merge blocks (for example, the solution marked in the figure).

The knowledge gap greatly motivates the evolutionary potential between merge blocks. In such an evolutionary process, excellent knowledge is continuously transferred, providing good learning ability for population evolution. As the number of evolutions increases, there are more and more outstanding individuals in the merge blocks, which greatly improves the advantages of the neighboring regions. Then the ability to transfer knowledge will gradually be maximized, which provides an internal driving force for the transfer and learning ability of knowledge between merge blocks.

In Algorithm 1, it can see a detailed process of the knowledge-transfer strategy. According to the preprocessing of the merge controller, the population is divided to obtain merge blocks, and the evolutionary sequence of the merge blocks is determined. For each individual in the merge block, its nearest two neighbors are selected and applied to the nominal convergence operator to generate offspring and update the ideal point. The angle between the offspring and the weight is calculated with (5), and the nearest subproblem is associated. The adaptive θ value is obtained through (7), which guides the PBI aggregation function to select better solutions. If the function value of the offspring is smaller than the parent, the parent is replaced.

1) Merge Controller: To ensure that each selected merge block is adjacent to optimized merge blocks, a merge controller is proposed in μMaOEA. Based on the idea of taking

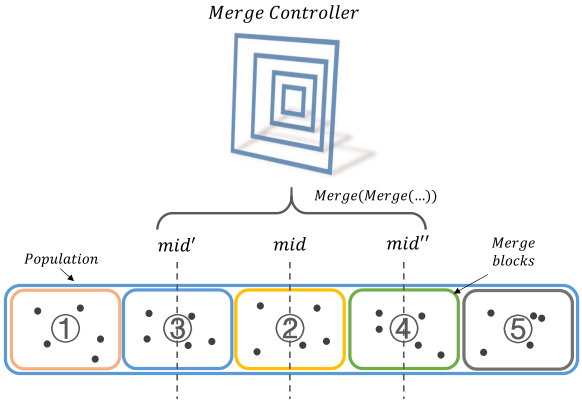


Fig. 3. The detailed progress of the merge controller obtains the optimization sequence of merge blocks through the recursive function. The figure shows how the population is divided and the obtains different mid indexes in the recursive stack.

Algorithm 1: Knowledge-Transfer Strategy.

Input: Population Pop, merge block P, weight vector W, neighbors B, population size NP, ideal point z^*
Output: Population Pop

1: **for** each $x_i \in P$ **do**
2: $y \leftarrow B(x_i)$ using norminal covergence operator;
3: Update the ideal point z^*;
4: /* Adaptive PBI Selection [34] */
5: **for** $j = 1$, to, NP **do**
6: $\alpha_j = \arccos \frac{(F(y)-z^*)\cdot W_j}{\|F(y)-z^*\|}$;
7: **end for**
8: $\varepsilon = \arg\min_{j\in\{1,\ldots,NP\}} \alpha_j$;
9: $\alpha_\varepsilon^{wi} = \arccos \frac{(F(X_\varepsilon)-z^*)\cdot W_\varepsilon}{\|F(X_\varepsilon)-z^*\|}$;
10: $\alpha_\varepsilon^{wb} = \min_{j\in\{1,\ldots,NP\},j\neq\varepsilon} \langle W_\varepsilon, W_j\rangle$;
11: $\theta_\varepsilon = k \cdot M \cdot (\alpha_\varepsilon^{wi} + \alpha_\varepsilon^{wb})$;
12: **if** $g^{PBI}(y|W_\varepsilon, z^*) < g^{PBI}(x_\varepsilon|W_\varepsilon, z^*)$ **then**
13: $x_\varepsilon = y$;
14: **end if**
15: **end for**

the medium of two ways in merge sort, the merge controller divides the population into merge blocks in this way. The merge controller generates a specific sequence to guide the evolution of merge blocks, aiming to achieve knowledge transfer under the knowledge gap. Fig. 3 illustrates the merge controller that determines the evolutionary sequence of merge blocks by constantly searching for mediums ($mid, mid', mid'' \ldots$) through a recursive function Algorithm 2.

As shown in Algorithm 2, first, it is necessary to input the minimum index ($left$ index) and maximum index ($right$ index) of merge blocks into the merge controller. Every time it runs to the first line of Algorithm 2, the merge controller calculates the medium index (mid index) of the current recursive stack. The second line adds the index of the current recursive stack to the sequence set. Lines 3 to 8 are the recursive entrance and termination, recursion occurs when the difference between

Algorithm 2: Merge Controller.

Input: $left$ index l, $right$ index r
Output: Merge block sequences Seq

1: $m = \lceil (l + r)/2 \rceil$;
2: $Seq = Seq \cup m$;
3: **if** $(m - l) > 1$ **then**
4: $Merge\ Controller(l, m)$;
5: **end if**
6: **if** $(r - m) > 1$ **then**
7: $Merge\ Controller(m, r)$;
8: **end if**

the mid index and the $left/right$ index is larger than 1. If $mid - left > 1$, input the mid and $left$ index into the recursive function Algorithm 2. If $right - mid > 1$, input the $right$ and mid index into the recursive function Algorithm 2. Otherwise, it means that the mid index is adjacent to the $left/right$ index, and all merge blocks have been sorted. At this time, the recursive stack is popped and the merged blocks sequence is output.

2) Nominal Convergence: Under the framework of μMaOEA, any meta-heuristic operator can be used to optimize the merge blocks. Nominal convergence is employed as a genetic operator in merge block optimization within the knowledge-transfer strategy, which makes the subsequent adaptive PBI selection more efficient.

Nominal convergence [5] is a concept from μMOEAs, which is defined as a lower number of evolutionary generations (typically 2 to 5). Goldberg pointed out that regardless of the length of the chromosome, a population size of 3 is sufficient to converge [35]. Meanwhile, Goldberg suggested applying genetic operators to a randomly generated population until it reaches nominal convergence and copying the optimal individuals from the converged population to another population, then the remaining individuals will be generated randomly. So far, the nominal convergence has been applied to some algorithms like micro-GA [5] and μGA^2 [20].

3) Adaptive PBI Selection: PBI approach [16] was chosen for μMaOEA due to its promising performance in MaOPs [36]. However, the performance of PBI is sensitive to the penalty factor θ, so the value of θ needs to be adjusted in different problems to make it perform well. To avoid this disadvantage, Han et al. proposed a useful approach that can adaptively adjust the value of θ according to changes [34], in the following form:

$$\alpha_i = \arccos \frac{(F(x) - z^*) \cdot W_i}{\|F(x) - z^*\|} \tag{5}$$

where $F(.)$ is tha objective value, z^* is the ideal point, W_i is the weight vector, the index of the corresponding subproblem:

$$\varepsilon = \arg\min_{i\in\{1,\ldots,NP\}} \alpha_i \tag{6}$$

On the one hand, the penalty factor θ determines convergence and diversity in the evolutionary process. On the other hand, PBI is directly related to weight distribution and its quality. The wider the angle between neighbors and the focused weight, the greater the improvement in the region. Therefore, the value of

Algorithm 3: Fuzzy Two-Stage Mechanism.

Input: Archive A, population Pop, merge block P, evolutionary rate x, population size NP
Output: Archive A
1: $\xi = \frac{1}{e^{-\alpha(x-c)}+1}$;
2: Generate ch_i by using (9), $i = 1, 2, \ldots, NP$;
3: **if** $mean(ch) > \xi$ **then**
4: $A \leftarrow Pop$ using (10) and (11);
5: **else**
6: $A \leftarrow A \cup P$ using *SDE*;
7: **end if**

θ is determined by the weight vectors and population, and the number of objectives affects population diversity, they must be taken into consideration [37]. Consequently, the factor θ of the adaptive PBI strategy is used [34]:

$$\theta_\varepsilon = k \cdot M \cdot (\alpha_\varepsilon^{wi} + \alpha_\varepsilon^{wb}) \tag{7}$$

where k is a scaling parameter, M is the number of objectives, α_ε^{wi} is the angle between the weight ε and the current individual x_ε, and α_ε^{wb} is the angle between ε and the closest neighbour.

B. Fuzzy Two-Stage Mechanism

A two-stage mechanism based on fuzzy logic is designed in this section to inherit and maintain knowledge through efficient decision-making with a fuzzy function at different stages, whose feasibility in tackling problems with a micro population has been well validated in μFAME [25]. In MaOEA-IT [32] and μMMABC [24], not only has the two-stage method been proven to be effective in MaOPs, but it has also verified its feasibility based on a micro population in it. Unlike such two algorithms, the fuzzy two-stage mechanism proposed in this paper focuses on convergence in the early stage, while the later stage maintains both convergence and diversity. This is to prevent the algorithm process from unsuccessful converging to the true PF in the later stage. In the first stage, local search based techniques are used to improve the convergence of the algorithm in the early stage. It is worth noting that (10) is different from [24], in that it uses more diverse individuals in local search, so that diversity will not be completely lost when focusing on convergence. In the second stage, a widely applied and effective evaluation method is applied, shift-based density estimation (SDE) [33], which can simultaneously maintain a convergence and diversity of the population in MaOPs. Ultimately, the convergence and diversity of archives are achieved, and knowledge inheritance and maintenance are completed through the above stages.

The pseudo-code of the fuzzy two-stage mechanism is shown in Algorithm 3. First of all, it calculates the membership value and chaotic numbers through (8) and (9), respectively. Then, a fuzzy controller is designed to compare the size between the membership value and the mean chaotic number to make the current stage decision. As shown in lines 3 to 6, if the mean chaotic number is larger than the membership value, it indicates that the methods of the first stage should be executed to perform a local search on the archive. If the average value of the chaotic number is less than the membership value, it indicates that the second stage method should be executed. Then, the evolved merge block is integrated into the archive, followed by an environmental selection based on SDE, which is used to truncate the archive. Through the above, the fuzzy two-stage mechanism guides the algorithm to update the archive and empowers knowledge. During the algorithm process, both convergence and diversity of the archive can be maintained continuously and effectively.

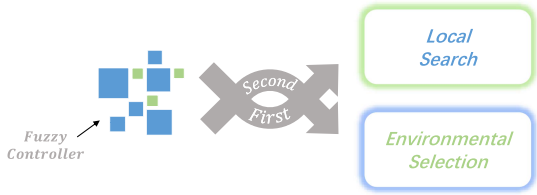


Fig. 4. The progress of the fuzzy two-stage mechanism. The figure shows that the fuzzy controller determines different stages, and the main approach of each stage include local search and environment selection based on SDE.

As shown in Fig. 4, a fuzzy controller is designed to make current stage decisions, and the applied fuzzy function is presented in (8). The purpose of using this function is to roughly divide the two stages into the early stage ($\approx$ 1/3 of the algorithm process) and the later stage ($\approx$ 2/3 of the algorithm process). According to (8), the membership function value is calculated by inputting the evolution rate to the fuzzy controller, where the evolution rate is the ratio of the number of current function evaluations to the maximum number of function evaluations. The calculation method for chaotic mapping is given by (9), and the rationale for utilizing the mean of chaotic numbers is due to their non-reproducibility and ergodicity. It is more possible than using a fixed value alone and indicates the global search at a higher speed than the random search that relies on probability [38]. In addition, the boundary of stage alternation is fuzzy through compared with the fuzzy function value. The algorithm enters a chaotic state during this period, increasing the likelihood of the algorithm. At this time, there is also a possibility of utilizing excellent knowledge from another stage at different stages, providing greater potential and diversity for knowledge transfer and empowering the algorithm adaptive ability while also reducing the possibility of falling into local optimum.

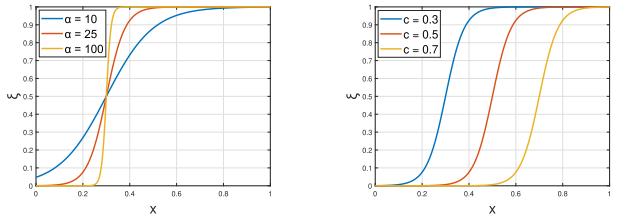
$$\xi = \frac{1}{e^{-\alpha(x-c)}+1} \tag{8}$$

where parameter α controls the slope, and parameter c controls the center of the fuzzy function.

$$ch_{k+1} = ch_k + ach_k(1-a) \tag{9}$$

where k is the index of individuals. ch_k, ch_{k+1} are the chaotic numbers. a is a parameter to regulate chaotic maps.

In the first stage, a local search is performed on the archive by (10) to enhance the convergence. After that, Gaussian disturbance (11) [39] is performed on the archive to maintain the diversity. Both of them improve the performance of the algorithm on micro populations and make knowledge more effective



(a) α (b) c

Fig. 5. The trend of the membership function in the fuzzy two-stage mechanism when the parameters α and c are set to different values. (a) shows the effect of α on the membership function and (b) shows the effect of c.

and pluralistic. Local search and Gaussian disturbance ensure that the algorithm has a certain diversity in the first stage while focusing on convergence.

$$x_i = x_j + rand(x_k - x_l) \tag{10}$$

where indexes j, k, l are randomly selected from $\{1, 2, \ldots, NP\}$, NP is the size of archive, $rand \in (0, 1)$.

$$\begin{cases} x_i' = x_i + N(\mu, \sigma^2), & \text{if } rand < \theta \\ x_i' = x_i, & \text{otherwise} \end{cases} \tag{11}$$

where $N(\mu, \sigma^2)$ is the Gaussian function, μ is the mean and σ^2 is the variance. In our work, $\mu = 0$, σ^2 is the difference between the upper and lower bounds of the decision vector divided by 20. $\theta = 1/D$, where D is the number of decision variables, $rand \in (0, 1)$.

In the second stage, the SDE is used to further enhance the ability of knowledge transfer through environmental selection. In the first stage, local search based techniques are applied to enhance the convergence of the archive. When the algorithm enters the second stage, both convergence and diversity are emphasized to prevent algorithm performance bias. According to [33], SDE, which can simultaneously evaluate diversity and convergence, has demonstrated outstanding performance in many-objective optimization. From the process, it will merge the evolved merge block with the archive and truncate it based on SDE to complete the environmental selection. The fuzzy two-stage mechanism focuses on convergence in the first stage, while emphasizing both convergence and diversity in the second stage by using these methods to improve the performance of micro populations in many-objective optimization.

Two parameters of the membership function in the fuzzy two-stage mechanism were studied in this paper. Parameter α controls the slope, and parameter c controls the center of the function. By setting many different values for parameters α and c, some representative values were selected to study their effects on the fuzzy two-stage mechanism. As shown in Fig. 5(a), ξ cannot fully cover [0, 1] when α is small. When the value is too large, the growth rate of ξ is too high, which will affect the decision-making of the fuzzy function. Therefore, only an appropriate value can balance the coverage of ξ and the fuzzy function effect. Based on experimental testing data, α was set to 25 to achieve the balance. μMaOEA processes different properties of the population in stages, and its 1/3 process enhances convergence, and 2/3 maintains both convergence and diversity. To determine the stages, it is necessary to adjust the center of the membership function, and Fig. 5(b) shows that $c = 0.3$ is appropriate for μMaOEA.

Algorithm 4: $\mu MaOEA$

Input: Merge block size N, population and archive size NP
Output: Archive A
1: Initialize population Pop and archive A;
2: Initialize ideal point z^*;
3: Generate weight vectors W with UR sampling;
4: Detect the neighbors B of each weight vector;
5: Divide population into merge blocks P_1,..., P_{max};
6: **while** $FEs < MaxFEs$ **do**
7: Update P_i with knowledge-transfer strategy;
8: Update archive A with fuzzy two-stage mechanism;
9: Update weight vectors W and neighbors B;
10: Shift to the next merge block P_{i+1};
11: **end while**

C. The Framework of $\mu MaOEA$

The main framework of μMaOeA is presented in Algorithm 4, which integrates the aforementioned techniques to optimize both the population and archive. Algorithm 4 describes a detailed process of μMaOEA. The first step is to initialize the number of function evaluations ($FEs = 0$), population, archive, weight, and ideal point. It's worth mentioning that the initialization of population and archive is based on Latin hypercube (LH) sampling [40], which enhances the diversity required for μMaOEA. The initial archive is the same as the initial population and weight, both of them are initialized with the uniform random (UR) sampling [41] method. At the same time, the neighbors of each weight are detected.

After initialization, a merge controller is designed to preprocess, dividing the population into multiple merge blocks to expand knowledge capability and generate a sequence set. The sequences are composed of different indexes, which will determine the evolutionary sequence of each merge block. Besides, the merge controller will shift to the next merge block according to the sequence set when Algorithm 4 runs to line 11, which enables the algorithm to evolve the population in the subregion and also reduces the computational resources required by the algorithm.

In the main loop of Algorithm 4, the merge block is updated with the knowledge-transfer strategy, and then the evolved merge block is used to update the archive with the fuzzy two-stage mechanism. In the process of updating weight, users can decide on a weight update method to update weight and neighbors. In the experiment, the update weight method proposed in ADEA [34] was used. After that, the merge controller shifts to the next merge block, repeating the main loop until the termination condition is met.

D. Computational Complexity Analysis

The computational complexity in μMaOEA mainly depends on the knowledge-transfer strategy, fuzzy two-stage mechanism, and weight update method. The following is an analysis of these three parts.

In the knowledge-transfer strategy, it is necessary to update each individual in the merge block. In this process, it needed to calculate the angle between them and each weight. So the knowledge-transfer strategy requires $O(N \cdot NP)$ computations, N is merge block size, NP is the size of the population and archive. The complexity of the fuzzy two-stage mechanism consists of two stages. According to the proportion of stages, it can be roughly divided into 1/3 of local search, $O(1/3 \cdot NP)$, and 2/3 of environmental selection, $O(2/3 \cdot NP \cdot \log(NP))$. Therefore, the complexity of the fuzzy two-stage mechanism is $O(NP \cdot \log(NP))$. The complexity of weight update depends on the weight update method chosen by the user. Additionally, the triggering of each stage in the fuzzy two-stage mechanism and weight update needs to meet the judgment requirements. In conclusion, the overall complexity of the μMaOEA is $O(NP \cdot (N + \log(NP)) + X)$, where N is merge block size, NP is the population size, and X is the complexity of weight update method.

IV. Experimental Results and Analysis

This section focuses on multiple experiments on μMaOEA including experimental setting, comparison with MaOEAs and μMOEAs, and validity on knowledge-transfer strategy and fuzzy two-stage mechanism. They verify the better performance of μMaOEA in many-objective optimization, micro populations, and technologies respectively.

A. Experimental Setting

- *Test Problems*
  All the algorithms performance under DTLZ [42] (DTLZ1 $\sim$ DTLZ7), MaF [43] (MaF1 $\sim$ MaF15), and WFG [44] (WFG1 $\sim$ WFG9), 31 test problems in total.
  These three benchmark suites are widely used to test algorithm performance. Meanwhile, they possess characteristics including convex, non-convex, multi-modal, deceptive, non-uniform, irregular, large-scale, sparse, and many-objective problems, comprehensively measuring algorithm performance.
  The parameter settings of all the above problems are shown in Table S - I and Table S - II.
- *Comparison Algorithms*
  In comparison with MaOEAs, MaOEA-IT [32], MaOEA-IGD [45], MaOEA-R&D [46], TS-NSGA-II [47], NSGA-II/CSDR [48], MaOEA-IBP [49], MOEA/AD [50], **7** MaOEAs were compared in total.
  In comparison with μMOEAs, μGA [5], HMGA [51], μMOGA [23], AMGA2 [22], AMGA [21], μMOPSO [52], μMMABC [24], **7** μMOEAs were compared in total.
  The parameter settings of all the above algorithms are shown in Table S - III.
  In μMaOEA, the size of the archive and population is 20. The size of merge blocks and neighborhood is 5. In the nominal convergence operator, nominal convergence $= 2$. Parameters in fuzzy controller are set to $c = 0.3, \alpha = 25$. The parameter α in chaotic maps is set to 3.7.
- *Performance Metrics*
  a) Inverse generation distance (IGD) [53] is a widely used performance indicator in evolutionary multi-objective optimization, which can comprehensively evaluate the performance of algorithms (convergence and diversity) through measuring the average Euclidean distance from the approximate PF obtained by the algorithm to the $true\ PF$, in the following form:
  $$IGD(P, PF_{true}) = \frac{\sum_{p^* \in PF_{true}} \min dist(p_i, p^*)}{|PF_{true}|} \tag{12}$$
  where P is an approximation of PF, PF_{true} is the sampling points from the $true\ PF$, $dist(p^*, P_i)$ denotes the minimum Euclidean distance between the $p_i \in P$ and $p^* \in PF_{true}$.
  b) Hypervolume (HV) [54] is also a widely used comprehensive evaluation indicator in EMO, which measures convergence and diversity by calculating the sum of the hypervolumes between each solution on the PF and the reference point.
  $$HV = \bigcup_{i=1,\ldots,N} Vol(i) \tag{13}$$
  where $Vol(i)$ is the hypervolume between each solution i on PF and the reference point.
- *Crossover and Mutation*
  The parameter settings for the genetic operator [55], concerning the probability of simulated binary crossover (SBX) and polynomial mutation, are set to $p_c = 1, p_m = 1/D$, where D is the number of decision variables. The distribution indexes of SBX and polynomial mutation are both set to $\eta_c = 20, \eta_m = 20$.
- *Other Setting*
  The population size is 20 in comparison with MaOEAs. Archive size is 20 and population size is 5 in comparison with μMOEAs. All experimental results are based on the Wilcoxon rank sum test, with a significance level of 0.05. All the problems mentioned above are run 30 times with the maximum number of function evaluations set to 10000 * M, where M is the number of objective, in a MATLAB platform for evolutionary multi-objective optimization, PlatEMO [56].

B. Comparison With MaOEAs

μMaOEA is compared to seven state-of-the-art MaOEAs on 6, 9, 12, 15-objective DTLZ1-7, MaF1-15, and WFG1-9. Fig. 6 is obtained by collecting the Freedman rankings of indicators from algorithms under different objective numbers and problems, and then calculating the average values. The detailed data tables of this experiment are provided in supplementary materials and their statistical results are shown in Fig. 6. The

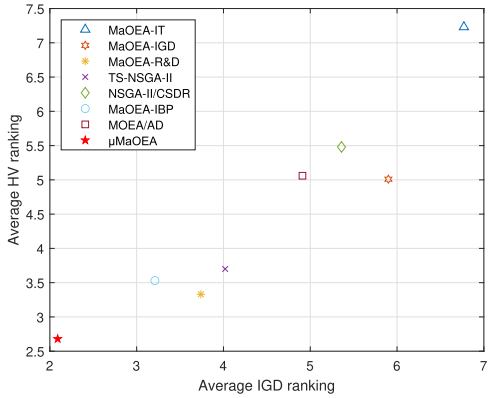


Fig. 6. Results of average IGD and HV Friedman ranking compared with MaOEA-IT, MaOEA-IGD, MaOEA-R&D, TS-NSGA-II, NSGA-II/CSDR, MaOEA-IBP, MOEA/AD on DTLZ, MaF, WFG problems.

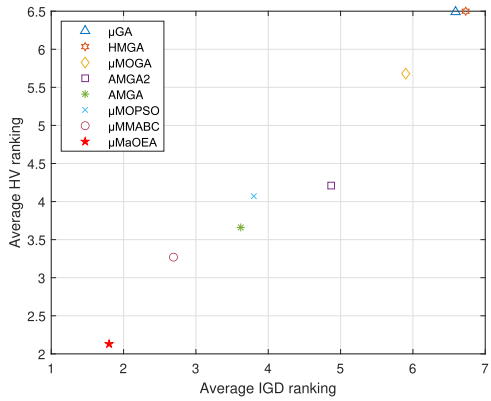
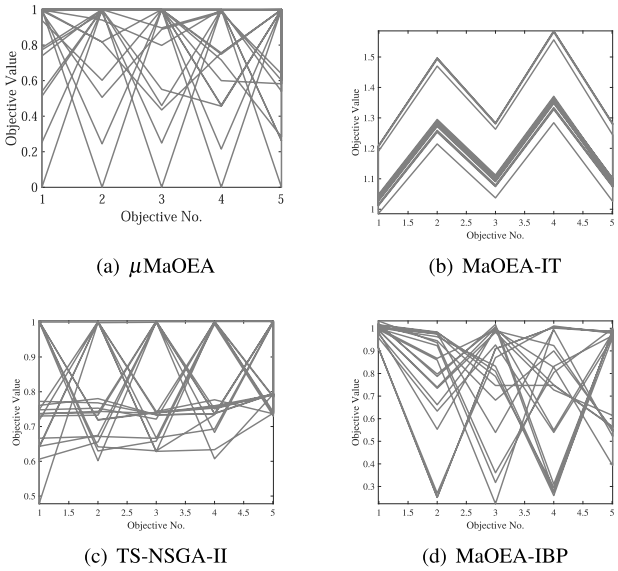


Fig. 8. Results of average IGD and HV Friedman ranking compared with μGA, HMGA, μMOGA, AMGA2, AMGA, μMOPSO, μMMABC on DTLZ, MaF, WFG problems.



(a) μMaOEA (b) MaOEA-IT

(c) TS-NSGA-II (d) MaOEA-IBP

Fig. 7. Parallel coordinates of non-dominated fronts obtained by μMaOEA, MaOEA-IT, TS-NSGA-II, MaOEA-IBP on the 5-objective MaF1.

closer the algorithm is to the coordinate origin, the higher its average ranking on the two indicators. It can be clearly seen from Fig. 6 that μMaOEA is much closer to the origin than the comparison algorithms, which means that μMaOEA performed better than seven comparison algorithms in the experiment.

DTLZ benchmark can be divided into two types according to the shape of PF. The extreme points of the first type are located on the coordinate axis, including DTLZ1-4. The second type is that not all extreme points are on the coordinate axis, including DTLZ5-7. Specifically, the extreme points of μMaOEA should be located on the coordinate axis, otherwise, it will result in an inaccurate estimation of the nadir point. Even in this situation, μMaOEA still has a powerful competitiveness in comparing algorithms.

Fig. 7 shows the non-dominated fronts of four algorithms on the 5-objective MaF1. It is not difficult to find from Fig. 7(b) that MaOEA-IT has not converged enough. Although TS-NSGA-II successfully converged to PF in Fig. 7(c), it did not maintain diversity well. In Fig. 7(d), though MaOEA-IBP has successfully converged to PF and its distribution is wider than TE-NSGA-II, but it is still not enough. As shown in Fig. 7(a), μMaOEA can successfully converge to the true PF while ensuring a uniform distribution in diversity.

C. Comparison With μMOEAs

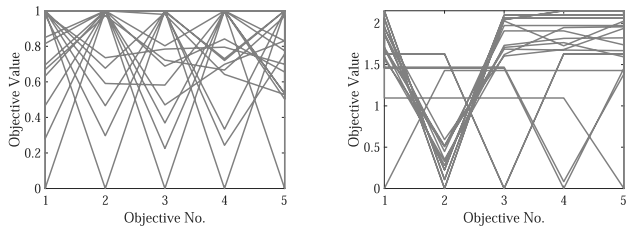
μMaOEA is compared to seven μMOEAs on 6, 9, 12, 15-objective DTLZ1-7, MaF1-15, and WFG1-9. Fig. 8 is also obtained by calculating the average Friedman ranking of the indicators, and the ranking results are shown in it. More detailed data result tables of it are provided in the supplementary materials. It can be clearly seen in Fig. 8 that μMaOEA is closer to the coordinate origin compared to the other 7 MOEAs, meaning it performs better among them.

μMaOEA mostly performs better than other MaOEAs on irregular PF problems WFG1, WFG2, DTLZ7, MaF8, and MaF9. Even on degenerate PF problems WFG3 and MaF6, μMaOEA can successfully search for feasible solutions well. In Fig. 9, a comparison is made with μMOPSO, AMGA2, and μGA in the objective space on the MaF1 with M = 5. It can be seen that μMOPSO demonstrates a poor convergence, while AMGA2 exhibits poor uniformity, and the main problem presented by μGA is its diversity. However, μMaOEA has demonstrated both excellent convergence and diversity compared to the other three comparison algorithms.

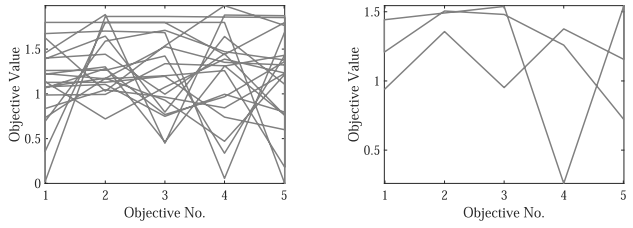
D. Validity on Knowledge-Transfer Strategy and Fuzzy Two-Stage Mechanism

In order to verify the validity of the knowledge-transfer strategy and fuzzy two-stage mechanism in μMaOEA, three variant algorithms have been correspondingly set.

- **Variant 1** μMaOEA-Fuzzy: μMaOEA-Fuzzy is a variant of μMaOEA that removes the knowledge-transfer strategy but retains the fuzzy two-stage mechanism. The other parts remain unchanged.

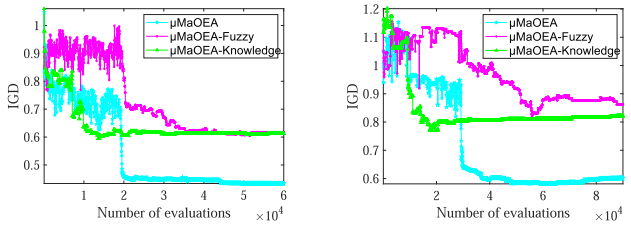


(a) μMaOEA (b) μMOPSO

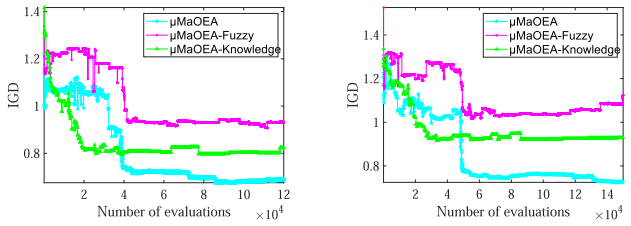


(c) AMGA2 (d) μGA

Fig. 9. Parallel coordinates of non-dominated fronts obtained by μMaOEA, μMOPSO, AMGA2, μGA on the 5-objective MaF1.



(a) 6M-DTLZ2 (b) 9M-DTLZ2



(c) 12M-DTLZ2 (d) 15M-DTLZ2

Fig. 11. The IGD convergence curves of μMaOEA and its variants on 6, 9, 12, 15 objective DTLZ2.

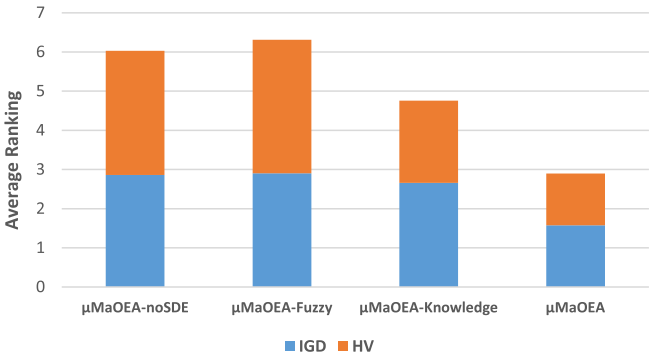


Fig. 10. Results of average IGD and HV Friedman ranking compared with three variants, μMaOEA-noSDE, μMaOEA-Fuzzy, μMaOEA-Knowledge on DTLZ, MaF, WFG problems.

- **Variant 2** μMaOEA-Knowledge: μMaOEA-Knowledge is a variant of μMaOEA that removes the fuzzy two-stage mechanism but retains the knowledge-transfer strategy. The other parts remain unchanged.
- **Variant 3** μMaOEA-noSDE: μMaOEA-noSDE is a variant of μMaOEA that removes the SDE from the fuzzy two-stage mechanism. The other parts remain unchanged.

Fig. 10 presents the comparison results of average IGD and HV Friedman ranking between μMaOEA and three variant algorithms. The results indicate that when a part of μMaOEA is removed, the algorithm performance will be greatly reduced. Especially after removing the knowledge-transfer strategy or SDE, the performance of the algorithm decreases the most. It means that they have played a significant role in improving algorithm performance. Fig. 11 validates the effectiveness of the knowledge-transfer strategy and fuzzy two-stage mechanism on DTLZ2. The convergence curve of IGD has significantly shifted upwards, and the performance of the variants has decreased significantly compared to μMaOEA.

The analysis of the performance degradation of the variants is provided below. Variant 1 prevents the algorithm from efficiently utilizing the excellent knowledge between niches to optimize the population. However, knowledge is crucial and an essential part of μMaOEA, which is why the algorithm performance deteriorates the most severely after removing the knowledge-transfer strategy. Variant 2 retains the knowledge-transfer strategy and eliminates the fuzzy two-stage mechanism, which makes the effective knowledge difficult to preserve and maintain. In addition, it puts the population in a conflict between convergence and diversity under this elimination. As is well known, SDE can effectively maintain both population convergence and diversity in MaOPs. Removing it in variant 3 will disrupt the performance of the algorithm in the second stage, leading to algorithm distortion.

V. Simulation

The proposal of μMaOEA is based on the MaOPs with limited computing resources and memory capability. Low-power microprocessors effectively balance the performance and computing resources, which fully represent the running environment of μMaOEA. Moreover, low-power microprocessors have many applications in embedded scenarios, such as micro robots [3]. One important function of micro robots is distance minimization optimization, therefore, the multi-point distance minimization problem [57] and multi-line distance minimization problem [58] were chosen in this paper.

A. Simulation Background

- $ESP32 - WROOM$

TABLE I
MEAN AND STANDARD DEVIATION IGD VALUES OF COMPARISON WITH μGA, NSGA-III AND MOEA/D ON MPDMP AND MLDMP

Problem	M	D	μGA	NSGA-III	MOEA/D	μMaOEA
MLDMP	6	2	5.7604e-3 (7.89e-3) -	2.0975e-2 (1.16e-2) -	7.2873e-2 (1.37e-3) -	8.9601e-2 (2.50e-3)
	9	2	2.2120e-4 (1.46e-3) -	7.8481e-3 (3.09e-3) -	1.5334e-2 (6.25e-4) -	2.1363e-2 (8.48e-4)
	12	2	1.4321e-5 (1.53e-4) -	8.4036e-4 (2.67e-4) -	1.1585e-3 (4.81e-5) -	2.7130e-3 (3.19e-4)
	15	2	3.4537e-7 (2.91e-5) -	1.9590e-4 (6.67e-5) -	1.7602e-4 (9.28e-6) -	4.6207e-4 (8.34e-5)
MPDMP	6	2	2.3805e-4 (1.13e-3) -	4.2647e-2 (4.87e-3) -	4.8591e-2 (1.10e-3) -	6.1768e-2 (1.17e-2)
	9	2	8.6294e-5 (3.18e-4) -	7.1331e-3 (9.81e-4) -	7.4939e-3 (3.76e-4) -	9.5906e-3 (6.67e-4)
	12	2	1.6061e-5 (6.78e-5) -	1.0960e-3 (1.85e-4) -	8.2854e-4 (4.32e-5) -	1.7052e-3 (1.55e-4)
	15	2	1.3550e-7 (7.39e-7) -	1.7326e-4 (3.38e-5) -	1.1926e-4 (7.28e-6) -	2.2888e-4 (3.65e-5)
+/-/=			0/8/0	0/8/0	0/8/0	—

ESP32-WROOM is a new generation of WIFI & Bluetooth dual mode and core wireless communication chip released by ESPRESSIF. The core of it is the ESP32-D0WDQ6. Two Xtensa 32-bit LX6 CPU cores can be individually controlled, with a computing power of up to 600 MIPS. The adjustment range of clock frequency is 80 MHz to 240 MHz, and it is equipped with 448 KB ROM, 520 KB SRAM, 16 KB RTC SRAM and 4 MB QSPI Flash. ESP32-WROOM, as an open source product, has rich peripheral interfaces. It is an excellent partner for mobile devices, IoT applications, and so on.

- $MicroPython$
  MicroPython, is a concise and efficient implementation of Python 3 that includes a subset of the Python standard library and is suitable for running on low-power microprocessors. It has good transportability and can run on many hardware platforms.
- $Setting$
  The low-power microprocessor is simulated and compared with μGA [5], NSGA-III [59] and MOEA/D [16] on objective dimensions of 6, 9, 12, 15 with a population and archive size of 20 and maximum number of function evaluations 10000. The algorithm parameter settings are the same as in Section IV. Each experiment is run 30 times independently.

B. Real-World Problems

1) Multi-Line Distance Minimization Problem: The decision space of the multi-line distance minimization problem (MLDMP) is a two-dimensional plane. MLDMP calculates the Euclidean distance from any point P = (x, y) to a series of M lines, each of which passes through an edge of a given regular polygon with mvertexes $(A_1, A_2, \ldots, A_M)$, where $M \geq 3$. The objective of MLDMP is to simultaneously optimize M values of distance.

One characteristic of the MLDMP is that the points within the regular polygon (including the boundaries) in decision space and their objective images are similar in Euclidean geometry, and this characteristic has achieved the visualization of MLMDP. Below are the three theorems of MLDMP, and the detailed proof of the three theorems can be found in [58].

Theorem 1: For any two interior solutions $P1(x_1, y_1)$, $P2(x_2, y_2)$ of an MLDMP $F = (f_1, f_2, \ldots, f_M)$ located within a regular polygon (including boundaries), the Euclidean distance between them is equal to a constant multiple of the Euclidean distance of its objective images, there have

$$\|P_1 - P_2\| = k \|F(P_1) - F(P_2)\| \tag{14}$$

Theorem 2: For an MLDMP with a regular polygon having M vertexes $(A_1, A_2, \ldots, A_M)$, the set of points located within the polygon (including boundary) forms the PS.

Theorem 3: Considering an MLDMP with a regular polygon of M vertexes $(A_1, A_2, \ldots, A_M)$, the feasible region $\Omega = \overline{\Phi} \wedge S$, where Φ is the union set of all the constrained polygons and S is a two-dimensional rectangle space in $\mathbb{R}^2$ (i.e., the rectangle constraint defined by the marginal values of decision variables). The PS of the MLDMP can be defined as the points inside the regular polygon (including the boundary).

2) Multi-Point Distance Minimization Problem: Multi-point distance minimization problem (MPDMP) is remarkably similar to MLDMP. Given a set $P = (p_1, p_2, \ldots, p_M)$ in the decision space. The objective vector is calculated by $F_i = dist(x, p_i)$, where x is a point in the decision space. Therefore, the objectives are to minimize the distances to a set of points, where the distance to points is considered an independent objective. The PS equals the convex closure of the points p_i, which is equivalent to the union of the volume enclosed by the convex hull and itself.

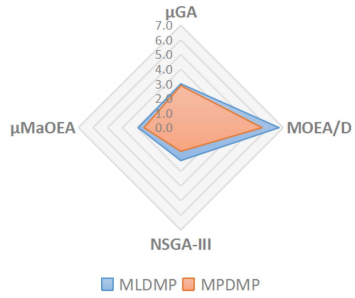
C. Simulation Results and Analysis

μMaOEA was compared with μGA, NSGA-III, and MOEA/D to conduct simulation based on a low-power microprocessor in the real-world problems, MPDMP and MLDMP. In Tables I and II, the data with a grey background indicates that the algorithm performs best on the problem. The symbol on the right side of the data indicates whether the null hypothesis is accepted in the Wilcoxon rank sum test at a significance level of 5%. "+", "-", "=" respectively indicate that the comparison algorithm is better than, worse than, and equal to μMaOEA.

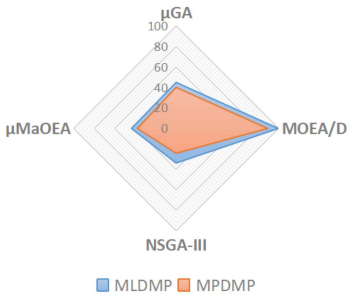
As shown in Tables I and II, μMaOEA performs better than the other three comparative algorithms on MPDMP and MLDMP. The knowledge-transfer strategy maintains the diversity of the population in a good state during the specific optimization sequence by dividing the population, and the subsequent operations further improve the performance of the population on two real-world problems. According to the analysis of [60], it can be concluded that MPDMP and MLDMP have a two-dimensional

TABLE II
MEAN AND STANDARD DEVIATION HV VALUES OF COMPARISON WITH μGA, NSGA-III AND MOEA/D ON MPDMP AND MLDMP

Problem	M	D	μGA	NSGA-III	MOEA/D	μMaOEA
MLDMP	6	2	5.7604e-3 (7.89e-3) -	2.0975e-2 (1.16e-2) -	7.2873e-2 (1.37e-3) -	8.9601e-2 (2.50e-3)
	9	2	2.2120e-4 (1.46e-3) -	7.8481e-3 (3.09e-3) -	1.5334e-2 (6.25e-4) -	2.1363e-2 (8.48e-4)
	12	2	1.4321e-5 (1.53e-4) -	8.4036e-4 (2.67e-4) -	1.1585e-3 (4.81e-5) -	2.7130e-3 (3.19e-4)
	15	2	3.4537e-7 (2.91e-5) -	1.9590e-4 (6.67e-5) -	1.7602e-4 (9.28e-6) -	4.6207e-4 (8.34e-5)
MPDMP	6	2	2.3805e-4 (1.13e-3) -	4.2647e-2 (4.87e-3) -	4.8591e-2 (1.10e-3) -	6.1768e-2 (1.17e-2)
	9	2	8.6294e-5 (3.18e-4) -	7.1331e-3 (9.81e-4) -	7.4939e-3 (3.76e-4) -	9.5906e-3 (6.67e-4)
	12	2	1.6061e-5 (6.78e-5) -	1.0960e-3 (1.85e-4) -	8.2854e-4 (4.32e-5) -	1.7052e-3 (1.55e-4)
	15	2	1.3550e-7 (7.39e-7) -	1.7326e-4 (3.38e-5) -	1.1926e-4 (7.28e-6) -	2.2888e-4 (3.65e-5)
+/-/=			0/8/0	0/8/0	0/8/0	—



(a) High-end computer



(b) low-power microprocessor

Fig. 12. Radar plots of comparing the average runtime of μGA, NSGA-III, and MOEA/D on high-end computer and low-power microprocessor.

decision space. As the number of objectives increases, it becomes more challenging to maintain diversity, and the difficulty of population converging towards the PF has significantly intensified. However, the data from Tables I and II shows that μMaOEA can still perform better.

μMaOEA runs on low-power microprocessors with micro population size. Therefore, its runtime should not be too long, which will increase the burden on the device. Not only were comparative experiments set up based on comprehensive performance indicators, but the runtime of algorithms on low-power microprocessors was also compared. Fig. 12 shows that the runtime of μMaOEA is lower than the other three comparative algorithms. The rapid optimization of μMaOEA benefits from optimizing each niche separately in the knowledge-transfer strategy, which reduces the time complexity of the algorithm by reducing the number of individuals optimized each time.

VI. CONCLUSION

This paper has suggested a micro many-objective evolutionary algorithm with knowledge transfer (μMaOEA) to better balance the computational effectiveness and limited resources in MaOPs. A knowledge-transfer strategy has been proposed to improve the evolutionary quality of the population, which achieves efficient knowledge transfer between niches by forming knowledge gaps. Additionally, a two-stage mechanism based on fuzzy logic has been designed to allow inheritance and maintenance of knowledge through fuzzy logic to decide the stage, and each stage uses different approaches to improve the convergence and diversity of the population.

In the experiment, μMaOEA is compared with 7 MaOEAs and 7 μMOEAs on 31 benchmark test problems. Meanwhile, it also verifies the validity of the knowledge-transfer strategy and fuzzy two-stage mechanism. All the experimental results above demonstrate that μMAOEA can achieve efficient optimization in various problems. In addition, μMaOEA, MOEA/D, NSGA-III, and μGA are simulated on a low-power microprocessor, with two real-world problems, MPDMP and MLDMP, employed to evaluate their performance. The results demonstrate the feasibility and applicability of μMaOEA in tackling real-world problems on a low-power microprocessor.

From the research, it can be seen that there is potential in the micro many-objective optimization and its application value in real-world problems. As for the future, we will make a deeper insight into the research on dynamic multi-objective optimization with micro population, so as to deal with dynamic environments in the real-world more effectively. The MATLAB source code of μMaOEA can be downloaded from Hu Peng's homepage: https://whuph.github.io/index.html.

REFERENCES

[1] H. Peng, C. Mei, S. Zhang, Z. Luo, Q. Zhang, and Z. Wu, "Multi-strategy dynamic multi-objective evolutionary algorithm with hybrid environmental change responses," *Swarm Evol. Comput.*, vol. 82, 2023, Art. no. 101356.
[2] H. Peng et al., "Multi-strategy firefly algorithm with selective ensemble for complex engineering optimization problems," *Appl. Soft Comput.*, vol. 120, 2022, Art. no. 108634.
[3] M. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Appl. Soft Comput.*, vol. 9, no. 3, pp. 1102–1110, 2009.

[4] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and Y. Jin, "Evolutionary many-objective optimization of hybrid electric vehicle control: From general optimization to preference articulation," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 2, pp. 97–111, Apr. 2017.

[5] C. A. Coello Coello Coello and G. Toscano Pulido, "A micro-genetic algorithm for multiobjective optimization," in *Proc. Evol. Multi-Criterion Optim.*, 2001, vol. 1993, pp. 126–140.

[6] K. Deb, M. Abouhawwash, and H. Seada, "A computationally fast convergence measure and implementation for single-, multiple-, and many-objective optimization," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 1, no. 4, pp. 280–293, Aug. 2017.

[7] J. Wang, B. Cen, S. Gao, Z. Zhang, and Y. Zhou, "Cooperative evolutionary framework with focused search for many-objective optimization," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 4, no. 3, pp. 398–412, Jun. 2020.

[8] L. Ma, S. Cheng, M. Shi, and Y. Guo, "Angle-based multi-objective evolutionary algorithm based on pruning-power indicator for game map generation," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 6, no. 2, pp. 341–354, Apr. 2022.

[9] M. Garza-Fabre, G. T. Pulido, and C. A. C. Coello, "Ranking methods for many-objective optimization," in *MICAI 2009: Adv. Artif. Intell.*, 2009, pp. 633–645.

[10] J. Horn, N. Nafpliotis, and D. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *Proc. 1st IEEE Conf. Evol. Comput. IEEE World Congr. Comput. Intell.*, 1994, vol. 1, pp. 82–87.

[11] Y. Tian, X. Su, Y. Su, and X. Zhang, "EMODMI: A multi-objective optimization based method to identify disease modules," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 5, no. 4, pp. 570–582, Aug. 2021.

[12] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[13] Z. Liang, W. Liang, Z. Wang, X. Ma, L. Liu, and Z. Zhu, "Multiobjective evolutionary multitasking with two-stage adaptive knowledge transfer based on population distribution," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 7, pp. 4457–4469, Jul. 2022.

[14] L. Wu, D. Wu, T. Zhao, X. Cai, and L. Xie, "Dynamic multi-objective evolutionary algorithm based on knowledge transfer," *Inf. Sci.*, vol. 636, 2023, Art. no. 118886.

[15] 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.

[16] 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.

[17] Z. Luo et al., "A micro dynamic multi-objective evolutionary algorithm for small-scale smart greenhouse with low-power microprocessor," in *Proc. Genet. Evol. Comput. Conf.*, 2024, pp. 687–690.

[18] H. Peng, F. Kong, and Q. Zhang, "Micro multiobjective evolutionary algorithm with piecewise strategy for embedded-processor-based industrial optimization," *IEEE Trans. Cybern.*, vol. 54, no. 8, pp. 4763–4774, Aug. 2024.

[19] Y. Abdi, M. Asadpour, and Y. Seyfari, "MOSM: A hybrid multi-objective micro evolutionary algorithm," *Eng. Appl. Artif. Intell.*, vol. 126, 2023, Art. no. 107000.

[20] G. Toscano Pulido and C. A. Coello Coello, "The micro genetic algorithm 2: Towards online adaptation in evolutionary multiobjective optimization," in *Proc. Evol. Multi-Criterion Optim.*, 2003, pp. 252–266.

[21] S. Tiwari, P. Koch, G. Fadel, and K. Deb, "AMGA: An archive-based micro genetic algorithm for multi-objective optimization," in *Proc. Conf. Genet. Evol. Comput.*, 2008, pp. 729–736.

[22] S. Tiwari, G. Fadel, and K. Deb, "AMGA2: Improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization," *Eng. Optim.*, vol. 43, no. 4, pp. 377–401, 2011.

[23] G. P. Liu, X. Han, and C. Jiang, "An efficient multi-objective optimization approach based on the micro genetic algorithm and its application," *Int. J. Mechan. Mater. Des.*, vol. 8, pp. 37–49, Mar. 2012.

[24] H. Peng, C. Wang, Y. Han, W. Xiao, X. Zhou, and Z. Wu, "Micro multi-strategy multi-objective artificial bee colony algorithm for microgrid energy optimization," *Future Gener. Comput. Syst.*, vol. 131, pp. 59–74, 2022.

[25] A. Santiago, B. Dorronsoro, H. J. Fraire, and P. Ruiz, "Micro-genetic algorithm with fuzzy selection of operators for multi-objective optimization: μ fame," *Swarm Evol. Comput.*, vol. 61, 2021, Art. no. 100818.

[26] J. E. Mendoza, D. A. Morales, R. A. Lopez, E. A. Lopez, J.-C. Vannier, and C. A. C. Coello, "Multiobjective location of automatic voltage regulators in a radial distribution network using a micro genetic algorithm," *IEEE Trans. Power Syst.*, vol. 22, no. 1, pp. 404–412, Feb. 2007.

[27] X. Liu and Z. Zhang, "Optimization of astronaut landing position based on micro multi-objective genetic algorithms," *Aerosp. Sci. Technol.*, vol. 29, no. 1, pp. 321–329, 2013.

[28] Z. Liang, K. Hu, X. Ma, and Z. Zhu, "A many-objective evolutionary algorithm based on a two-round selection strategy," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1417–1429, Mar. 2021.

[29] C. L. d. V. Lopes, F. V. C. Martins, E. F. Wanner, and K. Deb, "Analyzing dominance move (MIP-DoM) indicator for multiobjective and many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 3, pp. 476–489, Jun. 2022.

[30] S. Liu et al., "A self-guided reference vector strategy for many-objective optimization," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 1164–1178, Feb. 2022.

[31] 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.

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

[33] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 348–365, Jun. 2014.

[34] D. Han, W. Du, W. Du, Y. Jin, and C. Wu, "An adaptive decomposition-based evolutionary algorithm for many-objective optimization," *Inf. Sci.*, vol. 491, pp. 204–222, 2019.

[35] D. E. Goldberg, "Sizing populations for serial and parallel genetic algorithms," in *Proc. 3rd Int. Conf. Genet. Algorithms*, 1989, pp. 70–79.

[36] Q. Zhang, W. Zhu, B. Liao, X. Chen, and L. Cai, "A modified PBI approach for multi-objective optimization with complex Pareto fronts," *Swarm Evol. Comput.*, vol. 40, pp. 216–237, 2018.

[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, Oct. 2016.

[38] L. dos Santos Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1905–1913, 2008.

[39] Z. He, H. Peng, C. Deng, Y. Tan, Z. Wu, and S. Wu, "A spark-based gaussian bare-bones cuckoo search with dynamic parameter selection," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 1220–1227.

[40] L. Wei-Liem, "On latin hypercube sampling," *Ann. Stat.*, vol. 24, no. 5, pp. 2058–2080, 1996.

[41] L. R. de Farias and A. F. Araújo, "A decomposition-based many-objective evolutionary algorithm updating weights when required," *Swarm Evol. Comput.*, vol. 68, 2022, Art. no. 100980.

[42] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, *Scalable Test Problems for Evolutionary Multiobjective Optimization*. Berlin, Germany: Springer, 2005, pp. 105–145.

[43] R. Cheng et al., "A benchmark test suite for evolutionary many-objective optimization," *Complex Intell. Syst.*, vol. 3, pp. 67–81, 2017.

[44] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multi-objective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.

[45] Y. Sun, G. G. Yen, and Z. Yi, "IGD indicator-base d evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 173–187, Apr. 2019.

[46] Z. He and G. G. Yen, "Many-objective evolutionary algorithm: Objective space reduction and diversity improvement," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 145–160, Feb. 2016.

[47] F. Ming, W. Gong, and L. Wang, "A two-stage evolutionary algorithm with balanced convergence and diversity for many-objective optimization," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 10, pp. 6222–6234, Oct. 2022.

[48] J. Shen, P. Wang, and X. Wang, "A controlled strengthened dominance relation for evolutionary many-objective optimization," *IEEE Trans. Cybern.*, vol. 52, no. 5, pp. 3645–3657, May 2022.

[49] Z. Liang, T. Luo, K. Hu, X. Ma, and Z. Zhu, "An indicator-based many-objective evolutionary algorithm with boundary protection," *IEEE Trans. Cybern.*, vol. 51, no. 9, pp. 4553–4566, Sep. 2021.

[50] M. Wu, K. Li, S. Kwong, and Q. Zhang, "Evolutionary many-objective optimization based on adversarial decomposition," *IEEE Trans. Cybern.*, vol. 50, no. 2, pp. 753–764, Feb. 2020.

[51] W. J. Lim, A. B. Jambek, and S. C. Neoh, "Kursawe and ZDT functions optimization using hybrid micro genetic algorithm (HMGA)," *Soft Comput.*, vol. 19, no. 12, pp. 3571–3580, 2015.
[52] J. C. F. Cabrera and C. A. C. Coello, *Micro-MOPSO: A Multi-Objective Particle Swarm Optimizer That Uses a Very Small Population Size*.Berlin, Germany: Springer, 2010.
[53] M. Li, S. Yang, K. Li, and X. Liu, "Evolutionary algorithms with segment-based search for multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1295–1313, Aug. 2014.
[54] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Computation*, vol. 19, no. 1, pp. 45–76, 2011.
[55] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Inf.*, vol. 26, no. 4, pp. 30–45, 1996.
[56] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A matlab platform for evolutionary multi-objective optimization," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.
[57] M. Köppen and K. Yoshida, "Substitute distance assignments in NSGA-II for handling many-objective optimization problems," in *Proc. Evol. Multi-Criterion Optim.*, 2007, pp. 727–741.
[58] M. Li, C. Grosan, S. Yang, X. Liu, and X. Yao, "Multiline distance minimization: A visualized many-objective test problem suite," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 61–78, Feb. 2018.
[59] 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.
[60] C. He, Y. Tian, Y. Jin, X. Zhang, and L. Pan, "A radial space division based evolutionary algorithm for many-objective optimization," *Appl. Soft Comput.*, vol. 61, pp. 603–621, 2017.

Hu Peng received the B.S. degree in computer science and technology from Hunan Normal University, Changsha, China, in 2004, the M.S. degree in computer technology from the Huazhong University of Science and Technology, Wuhan, China, in 2008, and the Ph.D. degree from Wuhan University, Wuhan, China, in 2016. He is currently an Associate Professor with the School of Computer and Big Data Science, Jiujiang University, Jiujiang, China. His research interests include evolutionary algorithms and their applications. Dr. Peng was a Referee for more than 20 international journals, such as theIEEE TRANSACTIONS ON CYBERNETICS, *Information Sciences*, *Swarm and Evolutionary Computation*, *Knowledge-Based Systems*, *International Journal of Bio-Inspired Computation*, *Applied Soft Computing*, and *Soft Computing*.

Zhongtian Luo is currently working toward the B.S. degree with the School of Computer and Big Data Science, Jiujiang University, Jiujiang, China. His research interests include multiobjective optimization and its real-world applications.

Tian Fang is currently working toward the B.S. degree with the School of Computer and Big Data Science, Jiujiang University, Jiujiang, China. Her research interests include multiobjective optimization and its real-world applications.

Qingfu Zhang (Fellow, IEEE) received the B.Sc. degree in mathematics from Shanxi University, Taiyuan, China, in 1984, and the M.Sc. degree in applied mathematics and the Ph.D. degree in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively. He is a Chair Professor of computational intelligence with the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include evolutionary computation, optimization, and machine learning. Dr. Zhang is an Associate Editor for IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and IEEE TRANSACTIONS ON CYBERNETICS. He is a Web of Science Highly Cited Researcher of Computer Science for five consecutive years from 2016.