Ensemble Multifactorial Evolution with Biased Skill-Factor Inheritance for Many-task Optimization

Huynh Thi Thanh Binh, *Member, IEEE,* Le Van Cuong, Ta Bao Thang, Nguyen Hoang Long

***Abstract*—Current years have witnessed an increment in the number of research activities on improving the efficacy of multitasking algorithms for tackling challenging optimization problems. However, current approaches often present two potential problems. Firstly, although tasks may have different characteristics, existing literature usually utilizes only one search operator for all of them. Secondly, while multitasking environments comprise tasks of varying difficulty, previous proposals treat them equally. This paper proposes an algorithm named Ensemble Multifactorial Evolution with Biased Skill Factor Inheritance (EME-BI) for optimizing a large number of tasks simultaneously. In EME-BI, an effective parameter adaptation based on the knowledge transfer quality with Biased Skill-Factor Inheritance mechanism is designed to minimize negative transfer and allocate generated offspring to tasks that need resources. Besides, instead of using only one fixed search operator, EME-BI can automatically select the most appropriate one for each task at each evolutionary stage. Finally, the proposed algorithm is armed with a dynamically adjusted population size to promote exploitation. Empirical studies on various many-task benchmark problems and a real-world problem are conducted to verify the efficiency of EME-BI. The results portrayed that EME-BI achieves highly competitive performance compared to several state-of-the-art algorithms regarding the solution quality, convergence trend, and computation time. This proposal also won first prize at the CEC2021 Competition on Evolutionary Multitask Optimization, Multi-task single-objective optimization.**

***Index Terms*—Adaptive Transfer, Many-task Optimization, Multifactorial Evolutionary Algorithm, Parameter Adaptation.**

I. Introduction

In a swiftly transforming technological society, there is a growing body of literature that acknowledges the significance of optimization in every aspect of life. In fact, real-world optimization problems are frequently interconnected to a greater or less degree. Therefore, an essential requirement arises to handle them economically and reasonably by taking advantage of these synergies. Over the past few years, a novel research direction called Evolutionary Multitask Optimization (EMTO) has been inaugurated, and it immediately ushered in a new age for the intelligent computing sphere. Inspired by classic Evolutionary Algorithms (EAs), the EMTO paradigm focuses on ascertaining promising solutions for different problems simultaneously. Through optimizing multiple problems together, valuable genetic materials can be transferred between tasks, which often plays a crucial part in improving the quality of solutions, convergence characteristics, and resource usage compared to solving them separately.

The emergence of the cloud service and its series of optimization challenges [1] prompted Gupta *et al.* [2] to devise Multifactorial Optimization (MFO), which has currently matured into one of the most efficient and popular approaches when grappling with complex multitasking environments. Besides, an algorithm coined Multifactorial Evolutionary Algorithm (MFEA) was also introduced based on the concept of multifactorial inheritance from biology. With their potentialities, MFO and MFEA have become breakthroughs that have been successfully implemented in wide-ranging applications.

One of the first efforts to assess MFEA's ability to solve theoretical problems was also conducted by Gupta *et al.* [3] in 2015. In this study, the merging of MFEA with a bi-level solver resulted in a convergence of better solutions' quality with fewer evaluations than the conventional approach. A year later, Gupta *et al.* [4] proposed another method named multi-objective MFEA to work in the realm of Multi-Objective Optimization Problems (MOOPs). The authors evaluated their proposal using multiple MOOPs benchmarks in conjunction with an analysis of the composites manufacturing industry. This work greased the wheels for several other studies in the same field on the MOOPs [5], [6]. Concerning expensive problems, Min *et al.* [7] firstly defined the concept of Multi-Problem Surrogates (MPS), which enhances the optimization of a task using experience learned from addressing various preceded ones, and then applied this idea to their proposed Transfer Evolutionary Multi-Objective Optimization with MPS. Empirical experiments on synthetic benchmark functions and two practical case studies are undertaken to verify the proposal's efficacy. For reducing the calculation costs of expensive computational optimization problems, Ding *et al.* [8] proposed a new multitasking approach called MCEEA. MCEEA is equipped with decision variable translation and shuffling strategies, which allows for simultaneous optimization of computationally cheap problems with expensive ones.

On the other hand, MFEA is advantageous when it comes to solving several practical obstacles [9]. 2016 marked the first adoption of MFEA to software testing with more than two tasks [10]. The numerical results indicated that the proposed algorithm performs adequately in scenarios with resource constraints. In [11], Chandra *et al.* used MFEA to design a modular neural network, whose property is pre-

Manuscript... received... revised.... accepted.... This research was funded by Vingroup Innovation Foundation (VINIF) under project code VINIF.2022.DA00183. Corresponding author is Le Van Cuong. The contribution of each author is equal.

H. T. T. Binh, L. V. Cuong, and N. H. Long are with the School of Information and Communication Technology, Hanoi University of Science and Technology, Vietnam (e-mail: binhht@soict.hust.edu.vn, cuonglv.hust@gmail.com, longnh.mso@gmail.com).

T. B. Thang is an AI Engineer at Viettel Cyberspace Center, Viettel Group, Vietnam (email: thangtb3@viettel.com.vn).

serving knowledge in some modules even when the others are affected. In addition, there are many other studies that delved into the applications of MFEA in theoretical issues, such as combinatorial optimization [12], [13], and linkage-tree genetic algorithm [14], and also in practical ones, namely sparse reconstruction [15], fuzzy cognitive map learning [16], hyper-heuristic [17], job shop scheduling [18], and minimum routing cost clustered tree problem [19].

Despite its widespread adoption, the MFEA also has several flaws. Firstly, in MFEA, a fixed algorithmic configuration termed Probability of Random Mating (rmp) is used to regulate the degree of knowledge transfer and to balance between the algorithm's exploitation and exploration abilities. The rmp can be adjusted to 1 if tasks are foreseen to be closely related or to 0 otherwise. An rmp value close to 0 restricts inter-task knowledge exchange; each task only uses genetic information from its own population, promoting the exploitation of confined regions of the search space. A low rmp value also increases the likelihood of solutions getting trapped in local optima. On the contrary, an rmp value near 1 enables more knowledge transfer between tasks to occur, thereby enhancing the exploration of the entire search space, facilitating escape from local optima. Therefore, rmp must be selected sensibly to guarantee a good balance between exploiting specific regions and exploring the whole space. However, EAs generally deal with black-box optimization problems, in which neither the properties of the functions nor their relationship can be determined in advance. While problems in multitasking environments are not constantly similar, there is no assurance that the knowledge received would be helpful for a specific task. This phenomenon, denoted as the negative transfer, is a potential downside of MFEA. Secondly, due to this phenomenon, the MFEA shows ineffectiveness when the number of tasks increases. The existing MFEA is capable of well handling only two or three tasks at one time [20].

Several solutions hitherto have been proposed to deal with the negative transfer, usually followed in two main ways. The first is to adjust the rmp parameter, as used in Multifactorial Evolutionary Algorithm with Online Transfer Parameter Estimation (MFEA-II). Proposed by Bali *et al.* [21], MFEA-II, based on an online transfer parameter estimation, can adapt the extent of transfer using a mixture matrix of rmp. The second approach to minimize harmful interactions between tasks is to bring similar ones to knowledge transfer. One first proposal pursuing this strategy is Group-based MFEA (GMFEA) [22], which groups tasks of similar types and transfers knowledge within the same group. In 2019, Chen *et al.* [20] selected tasks for knowledge transfer through the feedback information of evolutionary process and Kullback-Leibler divergence, thereby introducing the Many-task Evolutionary Algorithm (MaTEA). Besides, Zhou *et al.* [23] investigated the effect of different crossover operators on knowledge transfer and proposed MFEA with Adaptive Knowledge Transfer (MFEA-AKT). MFEA-AKT can adaptively accommodate its crossover operator hinged on the information gathered during the evolution process. Lately, Liang *et al.* [24] proposed an Evolutionary Many-task Optimization Algorithm based on a Multi-source Knowledge Transfer (EMaTO-MKT). This algorithm has the ability to select multiple highly similar tasks to perform knowledge transfer based on Maximum Mean Discrepancy (MMD) as learning sources for each task.

Although a variety of methods have been developed in the EMTO field, they often treat easy and complex tasks equally. In fact, tasks do not always share the same difficulty. While some can be addressed effortlessly without transferring excessive knowledge, others with higher complexity will require more concentrated resources to solve. Therefore, equally allocating resources to solve tasks of different complexity as previous approaches can cause resource-consuming. To address this potential drawback, this paper proposes an algorithm named Ensemble Multifactorial Evolution with Biased Skill-Factor Inheritance (EME-BI) for optimizing more than three tasks simultaneously. Intending to tackle the detrimental interactions between tasks, EME-BI incorporates an rmp learning approach inspired by the adaptation of parameters in Differential Evolution (DE) [25]. The rationale behind this rmp learning method is *Rewarding the good and punishing the bad*, i.e., tweaking the rmp value of each task pair in accordance with the knowledge transfer quality. Notably, realizing that inter-task knowledge transfer is not always mutually beneficial, we integrate a novel mechanism called Biased Skill-Factor Inheritance (BI), which prioritizes assigning to generated offspring the tasks that they can perform more effectively, into our rmp learning scheme.

Apart from remedying the problem of fixed rmp value, our proposal also comprises a method for enhancing the exploitation ability. By and large, the selection of search operators plays a vital role in determining the quality of a solution in multitasking optimization. From the viewpoint of the *No Free Lunch* theorem, the most appropriate configuration is different depending on the concerned problem's characteristics. Accordingly, it is impossible to design a versatile algorithm to solve many tasks with the use of a single search method like most existing studies. Bearing these in mind, we endow the EME-BI with the ability to automatically select an efficient search operator for a specific problem via an ensemble strategy. In evolutionary computation, ensemble strategies are classified into two types, i.e., Low-level ensemble that combines multiple search strategies/operators/parameter values with different advantages and High-level ensemble where multiple evolutionary algorithms/variants are combined and run alternatively, or through multi-population [26]. This work employs a low-level ensemble strategy, in which two operators with different advantages are combined to tackle each task. Moreover, in this proposal, the size of the population is also reduced linearly to promote further exploitation, which is entirely compatible with the EAs' perspective. During the first generations, EA's population would need many individuals to scan the search space. If sufficient data has been gathered, only a few individuals are required to carry out the exploitation. To verify the efficacy of the proposed EME-BI, we adopted it on various many-task benchmark problems [20], [27] and a real-world problem and contrasted it with several state-of-the-art algorithms. The experimental results indicated that the EME-BI outperforms nearly all competing algorithms in terms of solution quality and search performance. The major

contributions of this work can be synthesized as follows:

- Propose a many-task evolutionary algorithm named EME-BI with the ability to assign the best search operator for each task separately.
- Propose a method to adjust the rmp value for each pair of tasks, featured by the Biased Skill-Factor Inheritance mechanism, which can suitably allocate generated offspring to tasks that need resources.
- Adopt the dynamic reduction mechanism in population size to promote the exploitation of the proposed EME-BI.
- Conduct experimental studies of EME-BI's efficacy on various many-task benchmark instances compared to state-of-the-art algorithms as well as on a real-world optimization problem called the planar kinematic arm.
- Investigate the influence and crucial role of each component in EME-BI by an ablation study.

The remainder of this paper can be described as follows. Section II covers the preliminary understanding of Multitask Optimization (MTO) and Many-task Optimization (MaTO) problems and related works. Section III elaborates on the proposed EME-BI. Section IV presents the experimental results on many-task benchmark problems and an ablation study of the proposed algorithm. Lastly, concluding remarks and discussions about future extensions are given in Section V.

II. Preliminaries

This section first briefly presents the fundamental knowledge on Multitask Optimization (MTO) as well as Many-task Optimization (MaTO) which works with more than three tasks. Then, the basic procedures of MFEA and related works are also outlined to give a better comprehension of our proposal.

A. Multitask Optimization and Many-task Optimization problems

In general, MTO intends to optimize multiple distinct tasks concurrently by entirely using a single search process. Consider a multitasking environment that contains K tasks to be minimized simultaneously. The j^{th} task, coined T_j, has a search space X_j, an objective function $f_j : X_j \rightarrow \mathbb{R}$, and can be constrained by one or several equality and/or inequality conditions. MTO determines a set of solutions $\{x_1, x_2, ..., x_{K-1}, x_K\}$ that satisfy:

$$x_i = argmin f_i(x), i = 1, 2, ..., K-1, K \tag{1}$$

Since the multitasking environment is composed of tasks with complicated relationships, MTO regularly functions with a limited number of tasks to avoid the occurrence of negative transfer.

When an MTO problem maintains more than three tasks, it is referred to as a MaTO problem. Recently, solving the MaTO has drawn significant attention from the computational intelligent society. The increased number of tasks challenges the design of algorithms, with the primary requirement being to leverage the relationships between them while avoiding their adverse transfers.

B. The basic Multifactorial Evolutionary Algorithm

Inspired by the bio-cultural concept of multifactorial inheritance, MFEA was the first algorithm to address the MTO problems. One of MFEA's innovations is its multitasking ability and scalability using a single population representation termed the Unified Search Space (USS). The design of MFO solvers in general or specifically in MFEA is based on the following four factors:

- *Factorial Cost f_j^i:* For each individual p_i in population P, factorial cost f_j^i is its cost value on task T_j.
- *Factorial Rank r_j^i:* The factorial rank r_j^i denotes the rank index of individual p_i in the task T_j according to the ascending order of factorial cost.
- *Scalar Fitness φ_i:* The scalar fitness of individual p_i is calculated as $\varphi_i = \frac{1}{min_{j=\{1,...,K\}} r_j^i}$. The larger the scalar fitness, the more important the individual's role in the unified population.
- *Skill Factor τ_i:* The skill factor τ_i indicates the task assigned to individual p_i, also known as the task that p_i can perform most effectively.

Throughout the evolution process, MFEA uses skill factors to classify the original population into separate subpopulations, each of which then undertakes the optimization of the corresponding task. Besides, MFEA is integrated with two cultural effects, i.e., the assortative mating and vertical cultural transmission, to facilitate knowledge transfer transpires between tasks, even if their skill factors are not the same.

Algorithm 1: The skeleton of MFEA

```
begin
    Generate an initital population P with N individuals;
    foreach individual p_i in P do
        Evaluate p_i on all tasks;
    Calculate skill factor and scalar fitness for each
      individual p_i in P;
    while Terminate conditions are not satisfied do
        O ← Generate offspring population with N
          individuals by using assortative mating on P;
        foreach individual o_i in O do
            Assign the skill factor τ_i by vertical cultural
              transmission;
            Evaluate o_i on the τ_i task only;
        P ← Select N fittest individuals from (P ∪ O) for
          the next generation;
```

The skeleton of MFEA is demonstrated in Algorithm 1. As mentioned above, the MFEA has not paid attention to the relationship of different optimization tasks. Therefore, it is only effective when encountering a small number of tasks due to the negative transfer phenomenon. In the proposed algorithm, instead of using a fixed rmp parameter, we introduce a mechanism to adjust this value based on the quality of knowledge transfer. Consequently, this approach helps our algorithm to improve its performance when solving a large number of tasks.

C. Related works

When faced with optimization problems modeled through the prism of EMTO, there are various approaches introduced to handle complex multitasking environments. Generally, two state-of-the-art techniques are often taken into account [28] in hopes of leveraging the inter-task complementary knowledge.

The first one uses a single search process on a population, similar to the MFO and MFEA. The most intricacy of this method is that it requires a suitable encoding and decoding design for merging all representatives of each optimization task into a USS. Within the USS, common knowledge between different tasks is implicitly transferred through search operators, mainly crossover functions. Recently, more attention is being given to concurrently tackling tasks based on their relatedness. In [29], a linearized domain adaptation strategy is combined into MFEA to transform the search space of a simple task to the search space similar to its constitutive complex task. Zheng *et al.* [30] introduced a Self-regulated Evolutionary Multitask Optimization algorithm (SREMTO) which automatically adapts to the intensity of knowledge transfer between assisted tasks based on their degree of relevance in the evolutionary process. The primary feature of SREMTO is the proper consideration of any individual's capability to solve each task thanks to its novel ability vectors, which substitute for skill factors in MFEA. In the same year, Bali *et al.* [21] proposed an MFEA-II that limits harmful interactions between optimization tasks by using an rmp learning scheme. Notwithstanding the effective rmp adaptation, its implementation also exposes a few shortcomings. Firstly, a firm reliance on probabilistic models is needed while adjusting the transfer intensity based on a mixture of them. Secondly, utilizing parent-centric crossover operators is required in MFEA-II. Finally, the runtime of MFEA-II when facing a large number of tasks is prolonged [19], [31] since sub-problems to be optimized are proportional to the square number of tasks.

The second typical approach to a multitasking scenario is to use more than one search process in parallel, analogous to Multipopulation-based Multitasking (MPM). Each task is assigned to a particular search process (or a population). These populations do not necessarily contain the same number of individuals or use the same search operator. In this case, the explicit knowledge transfer strategy is applied, that is, to exchange solutions between tasks periodically. In [32], Feng *et al.* made use of denoising autoencoder to explicitly transfer knowledge instead of transferring implicitly as existing MFEA does. However, the similarity between tasks is not taken into account, which may provoke negative transfer in the many-task environment. Together with limiting negative transfer phenomena, Shang *et al.* [33] first investigated the task selection for explicit EMTO algorithm in MaTO. Furthermore, the authors also presented a credit assignment strategy for this issue based on the feedback about the usefulness of transfer solutions. In [34], Thanh *et al.* adopt Multi-Armed Bandit (MAB) as the method to control the adaptive knowledge exchange between different tasks. Thanks to the effectively designed MAB agent, this algorithm has accurately learned the underlying relationship between tasks, but its execution time is relatively large as the number of tasks increases. Wang *et al.* [31] introduced the Multi-task Evolutionary Algorithm based on Anomaly Detection (MTEA-AD) for MaTO. To minimize the risk of negative transfer, MTEA-AD learns the relationship between tasks by an anomaly detection model armed with each of them. Besides, MTEA-AD is combined with a parameter adaptation strategy via elitism to adjust the degree of knowledge transfer and filter out individuals with negative information. Nevertheless, MTEA-AD can be implemented only for numerical optimization problems. Additionally, experience in just one preceding generation significantly influences the model parameter of this algorithm. Another research using MPM is an Evolutionary Many-task Optimization Algorithm based on a Multi-source Knowledge Transfer (EMaTO-MKT), proposed by Liang *et al.* [24]. An innovation of this algorithm is that it can perform knowledge transfer from multiple source tasks to one destination task at a time. In this study, multiple highly similar tasks are selected to perform knowledge transfer based on the maximum mean discrepancy. For each target task, populations of selected tasks are merged into a single population, which then is divided into c clusters using K-means. In each cluster, offspring are sampled from the pre-constructed probabilistic model. Although the efficiency of this technique has been proven in both single-objective and multi-objective MaTO test suites, EMaTO-MKT is also considered quite computationally expensive due to the cost of clustering and mapping between different spaces to measure the similarity of tasks. In addition, the selection of c is also a weakness of EMaTO-MKT. One of the most recently 2022 published studies by Wu *et al.* is Orthogonal transfer-based Multitasking Optimization (OTMTO) [35]. This paper mainly combats two existing problems in designing an efficient EMTO algorithm. The first problem is the difficulty in knowledge transfer between heterogeneous spaces, which comes up a lot in practice, caused by solved tasks' different dimensionalities. A cross-task mapping (CTM) strategy is proposed for this problem, which can map the global best individual of a task from its search space to the target task's search space. The second problem lies in the different degrees of similarity of tasks in different dimensions, handled by a novel orthogonal transfer method. Nonetheless, OTMTO may take a large computational time due to the use of nested DE algorithms, one for the CTM process, and one as the core optimization algorithm for each task.

Besides, there are other distinct approaches to lessening the harmful interactions between tasks and enhancing the MFEA's performance in many-task environments, notably managing multiple populations for implicit transfer. In [20], Chen *et al.* reckoned that it would be more practical to resolve tasks that arrive dynamically rather than simultaneously, as in MFEA. Therefore, the authors proposed an evolutionary framework MaTEA that can decide the best proper task for a given one based on their similarity and a reward strategy of knowledge transfer. For each task, MaTEA keeps an archive of individuals throughout several generations and then measures the Kullback-Leibler divergence of all archive pairs. Although this algorithm covers an attractive topic, its number of parameters still poses a challenge in applying it, especially in discrete

problems. Besides, the enormous stored population and the number of distance measurement operations to be performed likewise make this algorithm computationally expensive [34]. It is also worth noting the series of publications by Liaw *et al.* [36], [37] on the evolution of biocoenosis through symbiosis with two unique algorithms, including Evolution of Biocoenosis through Symbiosis (EBS) and Symbiosis in Biocoenosis Optimization (SBO). In brief, these studies were developed on the symbiosis in biocoenosis to balance the frequency and number of transfers based on the degree of symbiosis leveraged by the quality of information exchanged between tasks. Despite the creativity, their handcrafted adoptive transfer strategy fails to strike a balance between exploring potentially related task pairs and exploiting the captured frequency of successful transfer [34].

While various proposals have been developed for MTO with a small number of tasks, fewer efforts have been made to surmount the MaTO. Existing algorithms also have their weaknesses, and in general, there remains room for improving their performance and computational time. Therefore, we propose the EME-BI that works with the MaTO. To the best of our knowledge, this is the first study that investigates the effectiveness of ensemble search operators with adaptive selection and biased inheritance for many-tasking optimization. In addition, a performance comparison between the proposed algorithm and several state-of-the-art many-task algorithms, namely MFEA, MaTEA, EBS, SBO, MTEA-AD, and EMaTO-MKT, is also conducted.

III. Ensemble Multifactorial Evolution With Biased Skill Factor Inheritance

This section introduces an algorithm, namely EME-BI, for solving many-task optimization problems. Overall, each generation of EME-BI includes two phases, as presented in Figure 1. The first phase is recombination, which employs a single population, i.e., a unified search space for K tasks. This phase facilitates exchanges of information between tasks via inter-task crossovers, thereby allowing them to explore the search space of the others. Notably, instead of allowing offspring to randomly imitate the skill factor of their parents, EME-BI prioritizes assigning them to the task in which they may have a higher performance. This mechanism, called biased skill-factor inheritance, helps maximize the efficiency of knowledge transfer and avoids wasting too much effort on easy tasks. After the recombination phase, all individuals are fed into a learning phase. Each task now focuses on exploiting its own search space and regions explored from other tasks in the previous phase. To achieve this, the current population is divided into K sub-populations corresponding to K tasks, each of which applies several search operators. Computational resources for each task are automatically allocated to the operators based on their effectiveness in the previous generation. Because each search operator has its pros and cons, using multiple ones simultaneously allows the algorithm to select an appropriate operator for each task in each stage of the evolution process. Moreover, balancing exploration and exploitation resulting from different search behaviors can reduce premature convergence and stagnation. Finally, the population size in EME-BI is dynamically reduced to boost the convergence with fewer computational resources. More details of the proposed algorithm are described in the following subsections.

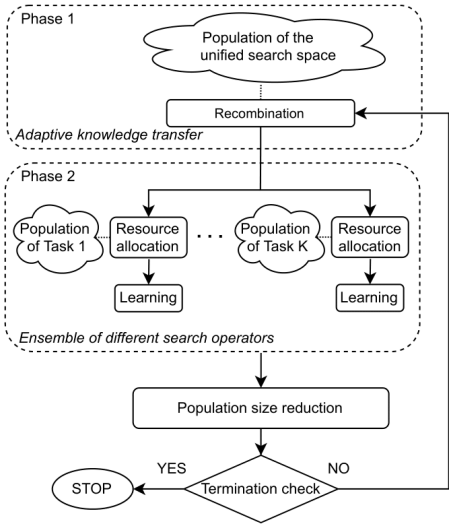


Fig. 1. Structure of the proposed EME-BI.

A. Adaptive Knowledge Transfer with Biased Skill-Factor Inheritance

Rather than using a fixed scalar parameter rmp like in the canonical MFEA, this paper proposes an adaptive mechanism to reduce the harmful interactions among tasks. In EME-BI, RMP takes the form of a non-symmetric $K \times K$ matrix (for the case of K tasks), in which each RMP_{ij} is the mean of a probability model that controls knowledge transfers from the j^{th} task to the i^{th} task. During the evolution process, the RMP matrix is adjusted dynamically according to a simple rule: *Rewarding the good and punishing the bad*. Specifically, RMP_{ij} is increased (rewarding) if at least one inter-task crossover between the i^{th} and j^{th} tasks improves a solution of the i^{th} task (successful inter-task crossover). In contrast, RMP_{ij} is decreased (punishing) to minimize the harmful interactions.

Algorithm 2 presents the inter-task crossover in EME-BI. Consider two individuals a and b with different skill factors τ_a and τ_b that are selected as parents. The random mating probability between a and b is generated from a normal distribution with the mean equals to the maximum of $\{RMP_{\tau_a\tau_b}, RMP_{\tau_b\tau_a}\}$ and the standard deviation of 0.1:

$$rmp = rand_n(max(RMP_{\tau_a\tau_b}, RMP_{\tau_b\tau_a}),\ 0.1). \quad (2)$$

All rmp values generated from Eq. (2) that lead to successful inter-task crossovers are recorded to update the RMP matrix. As a result, they can guide to generate rmp in the next generations.

At the beginning of the search, since there is no information about the similarity between tasks, all entries in the RMP matrix are initialized to 0.3 as the best-experimented value

Algorithm 2: Inter-task crossover with the proposed biased skill factor inheritance mechanism

Input: Two parents a and b with $\tau_a \neq \tau_b$.
Output: Two offspring c_1 and c_2.

1 **begin**
2 $rmp \leftarrow rand_n(\max(RMP_{\tau_a\tau_b}, RMP_{\tau_b\tau_a}), 0.1)$;
3 **if** $rand(0,1) \leq rmp$ **then**
4 $C \leftarrow$ *Intra-task crossover* between a and b;
5 **foreach** *offspring* c_i *in* C **do**
6 **if** $rand(0,1) \leq \frac{RMP_{\tau_a\tau_b}}{RMP_{\tau_a\tau_b}+RMP_{\tau_b\tau_a}}$ **then**
7 Assign to c_i skill factor τ_a;
8 Evaluate c_i for task τ_a only;
9 **if** $f_{\tau_a}(c_i) < f_{\tau_a}(a)$ **then**
10 $S_{\tau_a\tau_b} \leftarrow S_{\tau_a\tau_b} \bigcup \{rmp\}$;
11 $\Delta_{\tau_a\tau_b} \leftarrow \Delta_{\tau_a\tau_b} \bigcup \{f_{\tau_a}(a) - f_{\tau_a}(c_i)\}$;
12 **else**
13 Assign to c_i skill factor τ_b;
14 Evaluate c_i for task τ_b only;
15 **if** $f_{\tau_b}(c_i) < f_{\tau_b}(b)$ **then**
16 $S_{\tau_b\tau_a} \leftarrow S_{\tau_b\tau_a} \bigcup \{rmp\}$;
17 $\Delta_{\tau_b\tau_a} \leftarrow \Delta_{\tau_b\tau_a} \bigcup \{f_{\tau_b}(b) - f_{\tau_b}(c_i)\}$;
18 **else**
19 Randomly select a' with skill factor τ_a;
20 $c_1 \leftarrow$ *Intra-task crossover* between a and a';
21 Assign offspring c_1 skill factor τ_a;
22 Randomly select b' with skill factor τ_b;
23 $c_2 \leftarrow$ *Intra-task crossover* between b and b';
24 Assign offspring c_2 skill factor τ_b;
25 Evaluate c_1, c_2 for their assigned skill factors only;

in MFEA [2]. After each generation, $RMP_{\tau_a\tau_b}$ is updated as Eq. (3):

$$RMP_{\tau_a\tau_b} = \begin{cases} RMP_{\tau_a\tau_b} + c \times \text{mean}_{WL}(S_{\tau_a\tau_b}), & \text{if } S_{\tau_a\tau_b} \neq \emptyset, \\ (1-c) \times RMP_{\tau_a\tau_b}, & \text{otherwise,} \end{cases} \tag{3}$$

where c is the RMP updating rate, $S_{\tau_a\tau_b}$ is the collection of all rmp values that improve solutions of task τ_a. $\text{mean}_{WL}(\cdot)$ is a weighted Lehmer mean where weights are calculated based on the amount of objective improvements as in Eq. (4) and (5):

$$\text{mean}_{WL}(S_{\tau_a\tau_b}) = \frac{\sum_{i=1}^{|S_{\tau_a\tau_b}|} w_i \times (S^i_{\tau_a\tau_b})^2}{\sum_{i=1}^{|S_{\tau_a\tau_b}|} w_i \times S^i_{\tau_a\tau_b}}, \tag{4}$$

$$w_i = \frac{\Delta^i_{\tau_a\tau_b}}{\sum_{i'=1}^{|\Delta_{\tau_a\tau_b}|} \Delta^{i'}_{\tau_a\tau_b}}, \tag{5}$$

where $S^i_{\tau_a\tau_b}$ is the i^{th} element of $S_{\tau_a\tau_b}$ and $\Delta^i_{\tau_a\tau_b}$ is the i^{th} element of $\Delta_{\tau_a\tau_b}$. In other words, $\Delta^i_{\tau_a\tau_b}$ is the objective improvement of a solution of task τ_a resulting from a successful inter-task crossover between tasks τ_a and τ_b that occurred with the rmp value $S^i_{\tau_a\tau_b}$. The rmp value, which increases the solution quality significantly, will have a higher effect on future rmp values.

Moreover, an important observation is that the inter-task interaction is not always beneficial to both tasks. This is also supported by a remark that individuals generated by the inter-task crossover are unlikely to be high performing across all tasks [2]. Furthermore, since different tasks have different complex levels, devoting the same effort to all tasks may lead to wasting resources. For example, with the inter-task crossover between two tasks τ_a and τ_b above, assume that τ_b is supported by an easy task τ_a. If τ_a has found the optimal solution, assigning offspring to this task will not yield any improvement.

Based on these observations, we design a mechanism called Biased skill-factor Inheritance (BI), in which the probability that each child inherits the skill factor from the parent a or b is proportional to $RMP_{\tau_a\tau_b}$ and $RMP_{\tau_b\tau_a}$ (see Algorithm 2, lines 6-17). In the case $RMP_{\tau_a\tau_b}$ is greater than $RMP_{\tau_b\tau_a}$, assigning a child to task τ_a with higher probability can get a higher success transfer rate. Besides, it can help the algorithm put different efforts into different tasks at each evolution stage. Offspring can be assigned more to easy tasks at early stages because they can be solved easily with a large improvement percent. However, when easy tasks are converged, the generated offspring will be prioritized to assign to more difficult tasks, which are still room for improvement. This helps multitasking algorithms avoid wasting too much effort on easy tasks. Besides, the quality of transferred solutions from easy tasks to complex tasks is now enhanced, leading to more successful transfers.

It is worth mentioning that the proposed mechanism here is designed based on the comparison of objective values. Thus, it is less expensive in terms of memory space and time complexity than the existing population's distribution-based approaches. Moreover, the method can also be adapted to discrete optimization problems.

B. Learning Phase with an Ensemble of Search Operators

In this phase, individuals in the unified population are divided into K sub-populations according to their skill factors. After that, an ensemble of search operators is applied to each sub-population to improve the self-evolution of each task. Considering a sub-population $P_k(t)$ of the k^{th} task at generation t, to promote the appropriate operator and reduce the effects of unsuitable ones, the computational resources (represented by individuals of $P_k(t)$) are allocated to each search operator depending on their performance in the previous generation $t-1$. Besides, an improved Boltzmann method is adopted to enhance the global search ability of EME-BI. Algorithm 3 presents the skeleton of the learning phase.

Algorithm 3: Learning phase on sub-population $P_k(t)$

1 **begin**
2 Randomly divide $P_k(t)$ into m sets with a ratio of γ;
3 Assign search operators to the sets based on their performance ▷ See Subsect. III-B1;
4 **foreach** *individual* p *in* $P_k(t)$ **do**
5 $p' \longleftarrow$ Generate a neighbor individual using the assigned search operator;
6 Assign to p' skill factor τ_p and evalute p' in task τ_p;
7 Improve the global search ability with the improved Boltzmann method ▷ See Subsect. III-B2;

1) Resource Allocation Method: Suppose that m operators are used together in the learning phase. We randomly divide $P_k(t)$ into m sets, where $m-1$ among these are of equal size $\gamma \times N$, and the remaining one is of larger size $(1-(m-1)\times\gamma)\times N$, with γ is a population division rate and N is the size of $P_k(t)$. The γ value must be less than $\frac{1}{m}$ to ensure the existence of a set whose size is larger than the rest. After that, m search operators are assigned to these sets, where the most effective operator is linked to the largest one. All individuals in each set then undergo the learning phase with the assigned search operator. As a result, the most successful operator will obtain more computational resources.

At the first generation, since the effectiveness of search operators in dealing with the k^{th} task is unknown, m operators are randomly assigned to m sets. In later generations, the performance of each operator is evaluated based on the accumulated objective improvement and the consumed computational resources as follows:

$$\eta_k^i(t) = \frac{\Delta f_k^i(t-1)}{FE_k^i(t-1)}, \tag{6}$$

where $\eta_k^i(t)$ indicates the effectiveness of the i^{th} operator on the k^{th} task at the generation t. $\Delta f_k^i(t-1)$ is the accumulated objective improvement of the k^{th} task that contributed by the i^{th} operator, and $FE_k^i(t-1)$ is the number of function evaluations of the k^{th} task consumed by the i^{th} operator at the previous generation $t-1$. A higher value of $\eta_k^i(t)$ indicates that the i^{th} operator is more effective for the k^{th} task.

In the case of EME-BI, two operators with different search behaviors are implemented in the learning phase. The first is *DE/p*best*/1/bin* operator from the Differential Evolution (DE) family. This operator has been demonstrated to have a fast convergence by incorporating best solutions' information [25]. Hence, *DE/p*best*/1/bin* is appropriate for exploiting in tasks with unimodal landscapes or when the global basin was discovered in a multimodal landscape. From a solution $p = \{x^1, x^2, \ldots, x^D\}$, a new solution $p' = \{y^1, y^2, \ldots, y^D\}$ is generated by the *DE/p*best*/1/bin* operator as follows:

$$y^j = \begin{cases} x^j_{p\text{best}} + F \times (x^j_{r_1} - x^j_{r_2}), & \text{if } rand(0,1) \le CR \\ & \text{or } j = j_{rand}, \\ x^j, & \text{otherwise,} \end{cases} \tag{7}$$

where $x_{p\text{best}} = \{x^1_{p\text{best}}, x^2_{p\text{best}}, \ldots, x^D_{p\text{best}}\}$ is chosen from the top 10% best individuals of the current population, $x_{r_1} = \{x^1_{r_1}, x^2_{r_1}, \ldots, x^D_{r_1}\}$ and $x_{r_2} = \{x^1_{r_2}, x^2_{r_2}, \ldots, x^D_{r_2}\}$ are randomly selected from the current population. j_{rand} is a random index in [1, D] that ensures at least one difference between p and p'. The scale factor F and the crossover rate CR are two parameters of the *DE/p*best*/1/bin* operator, which are adapted dynamically as in SHADE [38].

The second operator is the Gaussian Mutation [39], which is adopted in many research due to its exemplary demonstration in global search behavior and ability to escape from the trap of local optima [40]. For this reason, Gaussian Mutation is a promising option for solving complex optimization tasks, especially those with rugged fitness landscapes.

With two operators ($m = 2$), the value of γ is now in the half-closed interval (0, 0.5]. The smaller the value of γ, the larger the difference in resources allocated to the operators. Moreover, the random division of $P_k(t)$ into two sets in each generation allows exchanges of information and experiences between search operators. For example, when $P_k(t)$ converges into a local optimum, the Gaussian Mutation can help the population escape from the local trap and explore new search regions. The new regions are then shared with *DE/p*best*/1/bin*, and thus, they can be rapidly exploited thanks to the strengths of this operator.

2) Enhance the Global Search Ability with Improved Boltzmann Method: The Boltzmann method is adopted in the learning phase of EME-BI to preserve the population's diversity and enhance the capability of escaping from local optima. Specifically, if the learning phase generates a better solution, the original individual p is always replaced. In contrast, if the learning phase fails to generate a better solution, p may be replaced by the less-fit solution p' with a probability that depends on the range of objective values and the population's diversity as follows:

$$P_{replace}(p,p') = \begin{cases} 1, & \text{if } \Delta f > 0, \\ \sigma_{\tau_p} \times exp(\frac{\Delta f}{f_{max} - f_{min}}), & \text{otherwise,} \end{cases} \tag{8}$$

where $\Delta f = f_{\tau_p}(p) - f_{\tau_p}(p')$, f_{max} and f_{min} are the maximum and minimum objective values of task τ_p in the current population, and σ_{τ_p} is an indicator that inversely proportions to the population's diversity of task τ_p:

$$\sigma_{\tau_p} = \frac{D_{\tau_p}(0) - D_{\tau_p}(t)}{D_{\tau_p}(0)}, \tag{9}$$

where $D_{\tau_p}(t)$ is the weighted deviation of individuals of task τ_p from the current best solution, $D_{\tau_p}(0)$ is the value of $D_{\tau_p}(t)$ at the initial generation $t = 0$. The value of $D_{\tau_p}(t)$ is determined as:

$$D_{\tau_p}(t) = \sum_{\substack{x \in P_{\tau_p}(t), \\ x \neq x_{best}}} w_x \times dis(x, x_{best}), \tag{10}$$

where $dis(x, x_{best})$ is the Euclidean distance between a solution x and the best solution x_{best} of $P_{\tau_p}(t)$, w_x is the weight of x that calculated based on objective values:

$$w_x = 1 - \frac{f_{\tau_p}(x)}{\sum_{x' \in P_{\tau_p}(t)} f_{\tau_p}(x')} \tag{11}$$

The rationale behind the calculations of σ_{τ_p} is that in most cases of Evolutionary Algorithms, it is well established that the population tends to converge toward the best solution. In addition, with the elitism selection, fitter individuals tend to survive longer and produce more offspring. Therefore, these members will influence the population's diversity more than the less-fit individuals. A small value of $D_{\tau_p}(t)$ indicates that individuals of $P_{\tau_p}(t)$ are being clustered in a small region around the best solution. This will lead to a high value of σ_{τ_p}, with an assumption that the initial population is the most diverse (following Eq. (9)). As a result, p' will have more chance to replace the original solution p, helping the population to escape the local optima (following the Eq. (8)).

C. Dynamic Population Size Reduction

The last feature of EME-BI is the dynamic population size reduction (DPSR) during the evolutionary process, which speeds up the convergence and avoids burning computational resources. The reason is that at the beginning of the search, more exploration is required to discover all promising regions as soon as possible. Therefore, the genetic pool needs to be more expansive, and the population needs to be large enough to cover the whole search space. As time continues, the individuals will aggregate around a particular region in the landscape. Thus, it is economical to have a small population at this point. Moreover, with the implementation of elitism selection, removing the worst individuals in the population gives the elites more chances to participate in the mating process. The exploitation around these members is promoted, and the convergence is sped up.

Based on these observations, the size of the unified population $P(t)$ is changed dynamically using the linear reduction method as Eq. (12):

$$e\_size(t) = K \times \left[N_{max} + FEs \times \left(\frac{N_{min} - N_{max}}{MAX\_FEs} \right) \right] \quad (12)$$

where N_{max} is the population size of each task at the first generation, and N_{min} is the minimum population size of each task at the end of the search. MAX_FEs and FEs are the total and the consumed computational budget. At the end of each generation, if the current population size is larger than the expected size $e\_size(t)$ calculated by Eq. (12), the worst individuals, according to scalar fitness values, are eliminated.

According to the descriptions above, we come to the framework of EME-BI as given in Algorithm 4.

Algorithm 4: Framework of EME-BI

```
begin
    P(0) ⟵ Randomly generate K × N individuals;
    Assign skill factor and evaluate each individual in P(0);
    Set t = 1;
    while termination condititions are not satisfied do
        /* START RECOMBINATION PHASE */
        P(t) ← Select top N × K/2 individuals based on scalar fitness;
        Initialize the offspring set P_c(t) ← ∅;
        while |P_c(t)| < N × K do
            Randomly select two individuals a and b from P(t);
            if τ_a = τ_b then
                [c1, c2] ⟵ Intra-task crossover (a, b);
                Assign offspring c1 and c2 skill factor τ_a;
                Evaluate c1, c2 for skill factors τ_a only;
            else
                [c1, c2] ⟵ Inter-task crossover (a, b)   ▷ See Algorithm 2;
            P_c(t) ← P_c(t) ⋃ {c1, c2};
        P(t) ← Select top N × K individuals from P(t) ⋃ P_c(t);
        Update the RMP matrix;
        /* END RECOMBINATION PHASE */
        /* START LEARNING PHASE */
        Divide P(t) into K sub-populations for K tasks;
        foreach sub-population P_k(t) do
            Perform the learning phase on P_k(t)   ▷ See Algorithm 3;
        Merge all sub-populations into P(t);
        /* END LEARNING PHASE */
        Reduce the population size of P(t);
        N ← |P(t)|/K;
        t ← t + 1;
```

D. Computational complexity

Algorithm 4 shows that the computational cost of each generation in EME-BI comes from the recombination phase, the learning phase, and the population reduction. The recombination phase consists of three main steps: generating $K \times N$ individuals for K tasks by the crossover requires $O(KND)$, updating scalar fitness with quicksort requires $O(KN\log N)$, and updating the RMP matrix requires $O(K^2)$. Therefore, the complexity of this phase is $O(KND + KN\log N + K^2)$. The learning phase employs the Gaussian mutation and the *DE/p*best*/1/bin* operator. The complexity of both operators is O(D), therefore, applying them to all individuals requires $O(KND)$. Finally, the population reduction step with quicksort requires $O(KN\log N)$. Overall, the time complexity of one generation in EME-BI is $O(2KND + 2KN\log N + K^2)$.

IV. Computational Results

A. Experimental Setting

To verify the effectiveness of the proposed algorithm, we conduct numerical experiments on three recent MaTO datasets, which are CEC'17 [27], [20], GECCO'20 [41], and a real-word application dataset [42], [31].

The CEC'17 dataset has been used widely in many state-of-the-art many-task algorithms. The CEC'17 consists of a 10-task benchmark problem, including four easy tasks (called E-task) and six complex tasks (called C-task). Each C-task has some assisted tasks that are highly similar and can support solving C-task more quickly and effectively. This dataset can verify whether the multitasking algorithm can take advantage of knowledge transfers between similar tasks and avoid harmful transfers between unrelated tasks to give better solutions than single task algorithms. The detail of the CEC'17 dataset is presented in Table I.

TABLE I
THE DETAIL OF THE CEC'17 DATASET

Task	Function	Search Space	Category	Assisted Tasks
T_1	$Sphere_1$	$[-100, 100]^{50}$	E-task	None
T_2	$Sphere_2$	$[-100, 100]^{50}$	E-task	None
T_3	$Sphere_3$	$[-100, 100]^{50}$	E-task	None
T_4	$Weierstrass_1$	$[-0.5, 0.5]^{25}$	E-task	None
T_5	$Rosenbrock$	$[-50, 50]^{50}$	C-task	T_1
T_6	$Ackley$	$[-50, 50]^{50}$	C-task	T_2
T_7	$Weierstrass_2$	$[-0.5, 0.5]^{50}$	C-task	T_3, T_4
T_8	$Schewefel$	$[-500, 500]^{50}$	C-task	None
T_9	$Griewank$	$[-100, 100]^{50}$	C-task	T_4
T_{10}	$Rastrigin$	$[-50, 50]^{50}$	C-task	None

The second dataset, GECCO'20, is the latest benchmark dataset used in the GECCO 2020 and CEC 2021 Competition on Multitask Optimization [41]. The dataset consists of ten 50-task benchmark problems. Each problem includes 50 complex optimization functions with many different shift and rotation

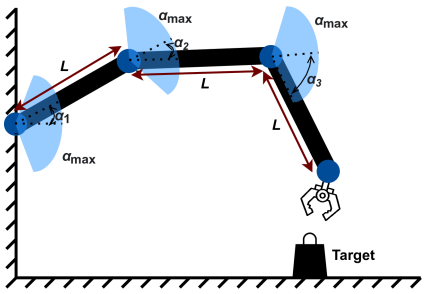


Fig. 2. An illustration of a planar arm with three links.

operators. The GECCO'20 is used to verify the effectiveness of algorithms when solving a large number of complex tasks simultaneously.

Besides, to further validate the efficiency of the proposed algorithm in dealing with real-world optimization problems, we apply EME-BI algorithm to solve the planar kinematic arm (PKA) problems [42], [31]. The PKA problem (Figure 2) aims to find an angle of each joint $\alpha = (\alpha_1, \ldots, \alpha_d)$ so that the tip of the arm is as close as possible to a target T. The search space dimension is thus defined by the number of joints d. Each task is defined by one arm and is parameterized by the length of the links L and the maximum angle for each joint α_{max} (same for all joints). Both these two values are normalized by the dimension d so that all arms have the same settings regardless of the dimension d:

$$L' = \frac{L}{d}, \tag{13}$$

$$\alpha'_i = \frac{1}{d} \cdot (\alpha_i - 0.5) \cdot \alpha_{max} \cdot 2\pi \tag{14}$$

In this experiment, the position of the target T is set to (1, 1) as in previous works [42], [31]. The kinematics of the arm is used to calculate the tip positions of links as follows:

$$M_i = M_{i-1} \cdot \begin{pmatrix} \cos(\alpha'_i) & -\sin(\alpha'_i) & 0 & L' \\ \sin(\alpha'_i) & \cos(\alpha'_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

$$p_i = M_i \cdot (0, 0, 0, 1)^T, \tag{16}$$

where $M_0 = I$ is the identity matrix, and p_i is the tip position of the ith link. The objective value of a given solution α is then defined by the Euclidean distance from the tip of the last link to the target T as follows:

$$f(\alpha, [L, \alpha_{max}]) = ||p_d - T|| \tag{17}$$

We construct 500, 1000, 1500 and 2000 tasks with $d = 10$ using a centroidal Voronoi tesselation [43] for this experiment.

The proposed EME-BI is compared with six state-of-the-art many-tasking algorithms, which are MaTEA [20], EMaTO-MKT [24], MTEA-AD [31], SBO [37], EBS [36], and the basic MFEA [2]. Because compared algorithms outperformed single-task algorithms in previous experiments, we do not compare the proposed algorithm with single-task algorithms in this paper. All algorithms are equipped with the same Genetic Algorithm solver and renamed MaTGA, EMaTO-MKT, MTEA-AD, SBGA, EBSGA, and MFEA. We do not use the Covariance matrix adaptation evolution strategy (CMA-ES) [37] solver because CMA-ES cannot be integrated into most compared algorithms. It is hard to verify the effectiveness of adaptation strategies with different solvers fairly. All experiments are run 30 independent times with different random seeds on an Intel Core i7-8700 CPU 3.20GHz and 16GB RAM computer. All algorithms, except MTEA-AD and EMaTO-MKT, are developed in Java programming language. MTEA-AD and EMaTO-MKT results are executed from the author's official Matlab code. The parameter settings of algorithms are as follows:

- Parameters of the compared algorithms are set up as their original paper to make fair comparisons.
- For CEC'17 and GECCO'20, the maximum number of function evaluations is $K \times 100000$, with K being the number of tasks. For complex planar arms dataset, the maximum number of function evaluations is $K \times 4000$.
- We use the Simulated Binary Crossover (SBX) with $\eta_c = 2$ for the intra-task crossover.
- To select parameters c, γ, and N_{min}, we experimented EME-BI with multiple parameter values on the CEC'17 dataset, shown in the supplementary material file. The results showed that $c = 0.06$, $\gamma = 0.3$, and $N_{min} = N_{max}/5$ is the most suitable values. Therefore, we use these values for all later experiments.

B. Experimental Scenario

To clarify the effectiveness of the proposed algorithm, we conduct five experimental scenarios as follows:

- Firstly, we conduct comprehensive comparisons between EME-BI and several state-of-the-art algorithms in terms of the solution quality, convergence trend, and computational time to prove its complete superiority.
- Secondly, we analyze the effect of the main components on the proposed algorithm.
- Thirdly, we compare the effectiveness of the ensemble operator method and single operator methods.
- Fourthly, we analyze the learned rmp values to verify whether the proposed algorithm has correctly learned the similarities between tasks to give effective transfers.
- Finally, we explain the effectiveness of the proposed biased skill-factor inheritance compared to the random inheritance.

C. Results and Discussion

1) Comprehensive Comparisons between the Proposed Algorithm and Several State-of-the-art Algorithms: In this section, our EME-BI is compared with several state-of-the-art many-task algorithms regarding the solution quality, computational time, and convergence trend.

The obtained average results by our proposal and other compared algorithms on the CEC'17 dataset are presented in Table II, where the results are averaged over 30 independent runs. Symbols $+, \approx, -$ indicate the compared algorithm

TABLE II
THE OBTAINED AVERAGE RESULTS OVER 30 INDEPENDENT RUNS OF EACH ALGORITHM ON EACH TASK IN THE CEC'17 DATASET. SYMBOLS $+$, $\approx$, AND $-$ INDICATE THAT THE CORRESPONDING ALGORITHM WAS STATISTICALLY BETTER, SIMILAR, AND WORSE THAN THE PROPOSED ALGORITHM ACCORDING TO WILCOXON SIGNED-RANK TEST AT $\alpha = 0.05$

Task	EME-BI	MFEA	EBSGA	SBGA	MaTGA	MTEA-AD	EMaTO-MKT
T_1	**0.00E+00**	1.32E+00 ($-$)	5.58E-04 ($-$)	1.43E-05 ($-$)	2.45E-04 ($-$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)
T_2	**0.00E+00**	1.27E+00 ($-$)	6.11E-05 ($-$)	2.28E-06 ($-$)	4.75E-04 ($-$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)
T_3	**0.00E+00**	1.17E+00 ($-$)	2.07E-06 ($-$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)
T_4	**0.00E+00**	3.37E+00 ($-$)	1.64E-05 ($-$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)	2.95E-01 ($-$)
T_5	**0.00E+00**	8.16E+02 ($-$)	3.29E-02 ($-$)	1.99E-03 ($-$)	2.16E-02 ($-$)	1.17E-05 ($-$)	5.67E+01 ($-$)
T_6	**0.00E+00**	2.00E+01 ($-$)	8.59E-04 ($-$)	2.00E+00 ($-$)	3.62E-03 ($-$)	1.49E-05 ($-$)	2.00E+01 ($-$)
T_7	**0.00E+00**	1.06E+01 ($-$)	1.61E-03 ($-$)	3.37E-03 ($-$)	5.57E-04 ($-$)	5.11E-04 ($-$)	**0.00E+00** ($\approx$)
T_8	**6.36E-04**	2.34E+03 ($-$)	4.73E-02 ($-$)	6.53E-04 ($-$)	3.40E-02 ($-$)	**6.37E-04** ($\approx$)	2.69E+03 ($-$)
T_9	**0.00E+00**	4.11E-02 ($-$)	4.58E-03 ($-$)	**0.00E+00** ($\approx$)	4.53E-03 ($-$)	4.93E-04 ($-$)	**0.00E+00** ($\approx$)
T_{10}	**5.54E+00**	1.55E+01 ($-$)	3.41E+01 ($-$)	3.09E+01 ($-$)	1.56E+01 ($-$)	1.76E+01 ($-$)	1.62E+01 ($-$)
$-/\approx/+$		10/0/0	10/0/0	7/3/0	8/2/0	5/5/0	5/5/0

is statistically significantly better, equal, or worse than our proposed algorithm according to Wilcoxon signed-rank test with $\alpha = 0.05$. The results showed that the proposed algorithm EME-BI achieved outstanding performance on all tasks. Specifically, EME-BI outperformed MFEA and EBSGA on 10 out of 10 tasks. In comparison to SBGA, MaTGA, EME-BI achieved better solution quality on 7, 8, and 5 out of 10 tasks, respectively, and gave similar quality on other tasks. Finally, our proposal exceeded MTEA-AD and EMaTO-MKT on 5 out of 10 tasks and matched on other tasks.

TABLE III
A WILCOXON SIGNED-RANK TEST WITH $\alpha = 0.05$ ON THE OBTAINED RESULTS ON THE GECCO'20 DATASET. $Better$, $Equal$, AND $Worse$ INDICATE THE NUMBER OF TASKS THAT THE PROPOSED ALGORITHM WAS BETTER, SIMILAR, AND WORSE THAN THE CORRESPONDING ALGORITHM

Algorithms	Better	Equal	Worse	p-value	Decision
MFEA	**500**	0	0	0.00	**+**
EBSGA	**440**	0	60	0.00	**+**
SBGA	**403**	0	97	0.00	**+**
MaTGA	**364**	0	136	4.00E-08	**+**
MTEA-AD	**499**	0	1	0.00	**+**
EMaTO-MKT	125	**199**	176	3.45E-3	**–**

To examine the effectiveness of the proposed algorithm in solving MaTO problems with a more significant number of high-dimensional tasks, we conducted more experiments and a Wilcoxon signed-rank test with $\alpha = 0.05$ on the latest dataset GECCO'20, which contains ten benchmarks. Each of them contains 50 tasks to be solved at the same time. The results shown in Table III indicated that the proposed EME-BI outperformed MFEA, EBSGA, SBGA, MaTGA, and MTEA-AD on most tasks. Besides, the obtained p-values are less than 0.05, confirming a statistically significant difference between these results. However, in comparison to EMaTO-MKT, our algorithm only gave better results on 125 tasks, worse on 179 tasks, and similar results on the remaining 199 tasks.

The effectiveness of our algorithms and compared algorithms are also investigated on a real-world dataset with 500, 1000, 1500, and 2000 planar kinematic arm tasks solved simultaneously. A Wilcoxon signed-rank test with $\alpha = 0.05$ is conducted on the obtained results as in Table IV. The results also showed that our algorithm is better than all algorithms on most tasks, and all obtained p-values are less than 0.05. This confirmed our algorithm's efficiency over current studies when solving a large number of tasks simultaneously. Notably, when increasing the number of tasks from 50 tasks to 500 tasks, EMaTO-MKT performs worse than our proposed algorithm. Because when facing a large number of tasks, using a fixed number of clusters and a fixed number of transferred tasks in EMaTO-MKT can cause a significant reduction in its performance. For example, when tasks are highly similar, clustering and only internally exchanging can restrict beneficial knowledge transfers between tasks. In contrast, when tasks have poor similarity, the difference between tasks in each cluster is significant, and sample offspring from the distribution on these clusters may affect their quality.

TABLE IV
A WILCOXON SIGNED-RANK TEST WITH $\alpha = 0.05$ ON THE REAL-WORLD DATASET. $+$, $=$, AND $-$ INDICATE THE NUMBER OF TASKS THAT THE CORRESPONDING ALGORITHM WAS BETTER, SIMILAR, AND WORSE THAN THE PROPOSED ALGORITHM

# Tasks	500	1000	1500	2000
Algorithm	-/=/+ (p-value)	-/=/+ (p-value)	-/=/+ (p-value)	-/=/+ (p-value)
MFEA	345/0/155 (**0.0**)	998/2/0 (**0.0**)	1500/0/0 (**0.0**)	1996/4/0 (**0.0**)
EBSGA	333/58/109 (**0.0**)	996/2/2 (**0.0**)	1499/0/1 (**0.0**)	1994/5/1 (**0.0**)
SBGA	304/89/107 (**0.0**)	971/27/2 (**0.0**)	1485/0/15 (**0.0**)	1957/5/38 (**0.0**)
MaTGA	287/0/213 (**0.04**)	795/295/0 (**0.0**)	1213/0/287 (**0.0**)	**
MTEA-AD	500/0/0 (**0.0**)	911/89/0 (**0.0**)	1378/122/0 (**0.0**)	1835/165/0 (**0.0**)
EMaTO-MKT	336/0/164 (**0.0**)	*	*	*

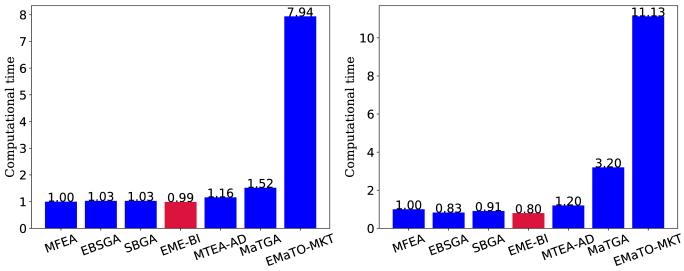
** indicates out of memory, * indicates massive time-consumption

Next, we compare the computational time of our EME-BI with other algorithms. All algorithms are run independently on the same computer. Except for MTEA-AD and EMaTO-MKT, which are obtained from the official Matlab code of the authors, all other algorithms are reimplemented by Java language. Therefore, to make fair comparisons, we implemented the MFEA in both languages and estimated the running time of MTEA-AD and EMaTO-MKT as follows:

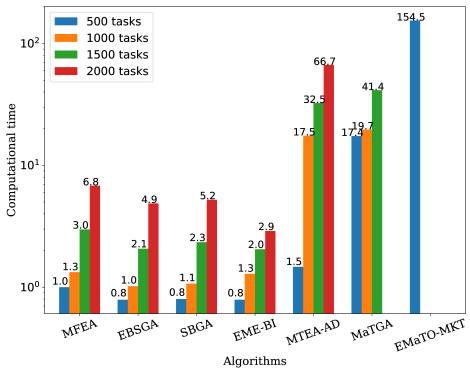
$$Time_J(\text{A}) \approx \frac{Time_M(\text{A}) \times Time_J(\text{MFEA})}{Time_M(\text{MFEA})} \tag{18}$$

where $Time_M(A)$ and $Time_J(A)$ are the running time of algorithm A in $Matlab$ and $Java$, respectively. The running time of all algorithms in comparison to MFEA on both the 10-task CEC'17, 50-task GECCO'20 benchmark, and real-world-task datasets are shown in Figure 3. The results showed that our EME-BI had a similar running time as MFEA, EBSGA, and SBGA on the datasets with small number of tasks. EME-BI ran 1.2-to-23 times faster than MTEA-AD. Compared

to MaTGA, our proposal got a much faster running time, approximately 1.5-to-21 times. The more the number of tasks, the more significant the running time improvement of our algorithm. Notably, although the EMaTO-MKT algorithm gave good quality solutions, it had an extremely high computational cost. When solving 10, 50, and 500 tasks, the computational time of the EMaTO-MKT went up 7.94, 11.13, and 154.5 times more than the MFEA due to the time consuming of clustering, mapping, and Kullback-Leibler divergence based similarities. Huge computational time makes EMaTO-MKT significantly reduce its competitiveness in practical settings.



(a) 10-task CEC'17 (b) 50-task GECCO'20



(c) Complex Planar Arms

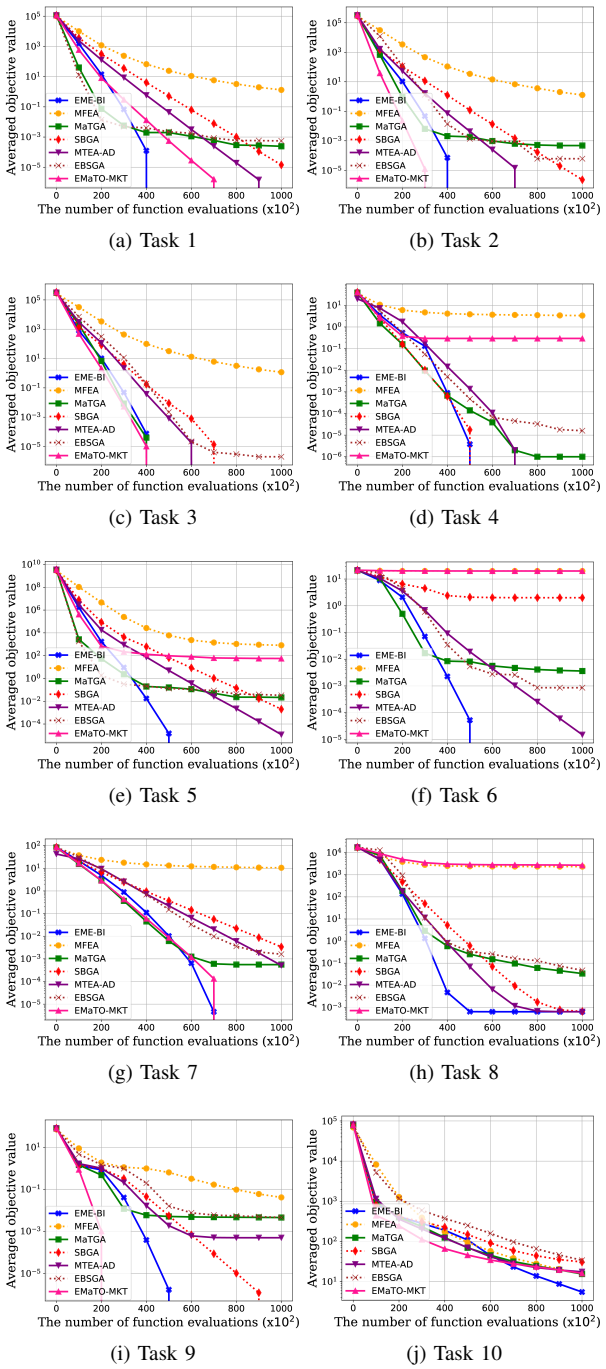
Fig. 3. Running time of all algorithms.

Finally, we evaluated the convergence trend of the EME-BI and compared algorithms. The objective values of each task at each generation on the CEC'17 dataset are averaged over all runs, as shown in Figure 4. The findings showed that EME-BI converged faster than all other algorithms except MaTGA and EMaTO-MKT. In the first 300 generations, EME-BI was slower than MaTGA on some tasks, but in later generations, our algorithm showed a superior convergence trend. In comparison to EMaTO-MKT, our algorithm only converged slower on tasks T_2 and T_9, similar on task T_3, and faster on all other tasks. Our EME-BI reached the optimal solution on 9 out of 10 tasks (except T_{10}) at the 400^{th} generation on E-tasks and the 700^{th} generation on C-tasks.

Additionally, our proposal also won first prize at the CEC'21 Competition on Evolutionary Multi-task Optimization, Multi-task Single-Objective Optimization (MTSOO)[1].

Overall, the EME-BI proved a solid ability to find high-quality solutions on both a small and large number of tasks.

[1]https://bit.ly/MTOCompetitionCEC2021



(a) Task 1 (b) Task 2

(c) Task 3 (d) Task 4

(e) Task 5 (f) Task 6

(g) Task 7 (h) Task 8

(i) Task 9 (j) Task 10

Fig. 4. Convergence trend of all algorithms on the CEC'17 MaTO dataset.

Our proposed algorithm also has a good convergence trend and a very competitive running time compared to state-of-the-art many-task algorithms.

2) Analyze the Effect of Main Components: We analyze the effect of three main components: RMP Adaptation, Dynamic Population Size Reduction (DPSR), and Learning phase in improving the performance of the proposed algorithm. Three variants of the proposed algorithm are implemented: EME-BI-without-RMP-Adaptation, EME-BI-without-DPSR, and EME-BI-without-Learning-Phase. These algorithms contain all components of EME-BI but are not equipped with the proposed

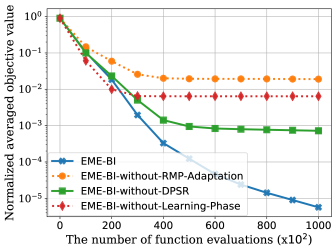
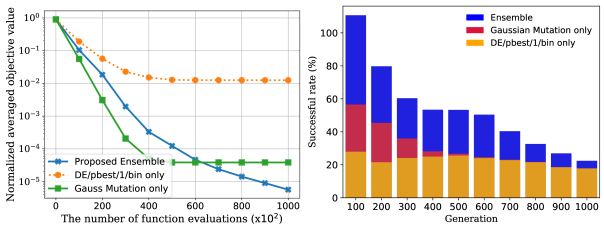
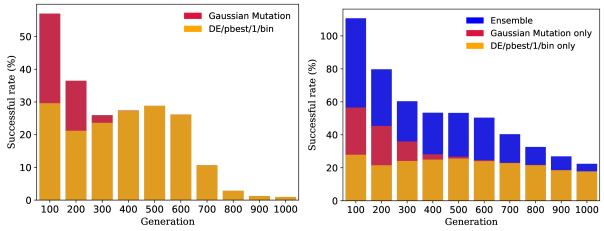


Fig. 5. Analyze the influence of the main components.



(a) Convergence trend (b) Successful rate

Fig. 6. Analyze the effectiveness of the proposed ensemble method



(a) E-task (b) C-task

Fig. 7. Each operator's success rate in the proposed ensemble method.

RMP adaptation, DPSR, and Learning phase, respectively.

The obtained results in each generation are normalized and averaged on all tasks, as shown in Figure 5. The findings showed that the performance of EME-BI is reduced if all three components are not integrated. If EME-BI does not include the proposed RMP adaptation, it shows the most significant drop in algorithm quality. The reason is that the algorithm cannot avoid negative transfers between unrelated tasks and promote to exchange of beneficial information between highly similar ones if using a fixed transfer parameter. The algorithm wastes more chances for ineffective transfers and cannot explore more promising regions through knowledge exchange between similar tasks.

Besides, suppose EME-BI is not equipped with the proposed Learning phase. In that case, it causes premature convergence to sub-optimal solutions on several complex tasks due to the strong exploitation ability of the DPSR method.

In general, combining these components helps to solve each other's limitations. For example, the RMP adaptation helps tasks make better use of knowledge from other tasks. Meanwhile, the DPSR and Learning phase help improve the self-evolution of each task. When the self-evolution of each task falls into a local trap, it can automatically select strong exploration operators thanks to the Learning phase or receive knowledge information from other support tasks to escape and explore new search regions. In contrast, it may use good exploitation operators to obtain better convergence if the convergence is accelerating.

3) Analyze the Effectiveness of the Proposed Ensemble Method: To explain the effectiveness of the proposed ensemble method, we implemented variants of the EME-BI, which use only one of the operators separately in the learning phase. The obtained results from these variants are shown in Figure 6a. The findings showed that using multiple operators together is better than only using one in terms of solution quality. This phenomenon can be explained as follows. Using the $DE/p\text{best}/1/bin$ operator is often a good choice for unimodal optimization problems. However, it is unsuitable for complex tasks with many local traps. In such cases, it may lead to a reduction in the exploration of the algorithm. Besides, function types of tasks in multitasking environments are unknown and very diverse, especially when the number of tasks becomes very large. Combining multiple operators gives multitask algorithms more chances to choose the most suitable operator for each task rather than only one fixed operator for all tasks.

Figure 6b describes the success rate of using multiple operators in the form of a stacked bar chart. The results showed that using the various operators is better than using only one operator in the first 600 generations. However, the ensemble gave worse results in later stages than "$DE/p\text{best}/1/bin$ only". The reason is that the ensemble method found global optima in 9 out of 10 tasks after the first 600-700 generations. Therefore, the success rate of the ensemble at later stages decreases because it cannot bring about as much improvement as before. The results also showed that "$DE/p\text{best}/1/bin$ only" had a better success rate than "Gaussian Mutation only". The reason is that the $DE/p\text{best}/1/bin$ operator has a strong exploitation ability thanks to incorporating the best solutions' information. As a result, the $DE/p\text{best}/1/bin$ operator can more thoroughly and effectively exploit the current search region of each task and regions discovered by knowledge exchange between tasks. Meanwhile, the Gaussian Mutation operator only uses small random mutations. Therefore, it gave "$DE/p\text{best}/1/bin$ only" a better success rate.

Finally, to further explain obtained results by using multiple operators in the learning phase, we analyze the actual success rates of each operator on easy tasks (E-task) and complex tasks (C-tasks). Stacked bar charts in Figure 7 indicated that Gaussian Mutation was used more successfully in the first 200 or 300 generations, especially on complex tasks to avoid local traps and explore more new potential search spaces. In contrast, $DE/p\text{best}/1/bin$ was used more successfully in final generations, especially on easy tasks to exploit promising regions found in previous stages. Overall, we can conclude that the proposed ensemble of multiple operators is an effective approach to solving many-task optimization problems.

4) Analyze the learned rmp values: This section analyzes the actual learned rmp values to verify whether the proposed algorithm can benefit from fruitful knowledge exchange and

avoid harmful interactions among tasks.

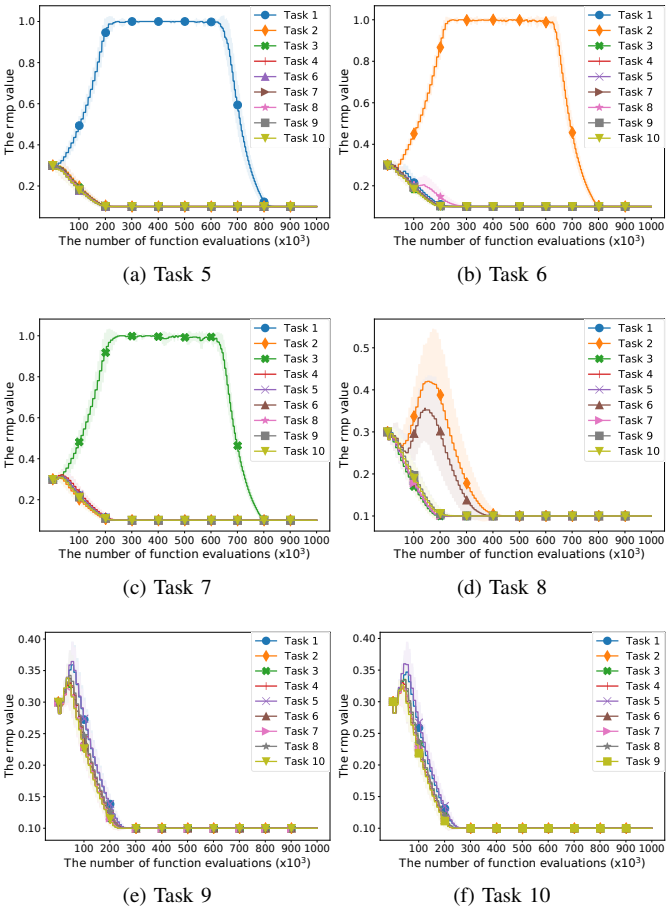
Figure 8 shows generated rmp values between each complex task and other tasks in each generation. For tasks T_5, T_6, and T_7, the learned rmp values between these tasks and their assisted tasks tend to increase to 1 after the first 200 generations. The higher rmp values, the more knowledge exchanges between them are encouraged. Hence, it can help complex tasks to achieve good solutions more quickly. However, from the 700^{th} generation onwards, these rmp values tend to decrease because these complex tasks reached the optimal solution, as shown in Figure 4. Knowledge transfers now cannot bring any improvement in solution quality. Therefore, the rmp is decreased to avoid wasting computational resources on ineffective transfers.

It's also worth noting that although T_3 and T_4 reported in Table I are assisted tasks for T_7, our experiments showed that most of the support for T_7 is only contributed by T_3. T_4 only supported T_7 in the first 50-60 generations with a reasonably low transfer success rate. Furthermore, although T_4 is the assisted task of T_9, its rmp value is quite similar to the remaining tasks. Recent studies [20], [31] also indicated this phenomenon when the successful transfer times between T_9 and other tasks are quite similar. The successful transfer times between T_9 and T_4 are relatively small compared to other assisted task pairs. The reason is that T_4 only has a 25-dimensional search space while all other complex tasks have a 50-dimensional search space. Meanwhile, at the beginning of the optimization process, multitasking algorithms build a unified search space for all tasks with the maximum dimension. Therefore, T_4's solutions are padded with rear random 25 elements, which are not optimized in the objective function of T_4. When performing knowledge transfer crossover between T4 and other tasks, these unoptimized rear 25 elements may affect the performance of new offspring. As a result, choosing T_4 to be an assisted task is not as prominent as the other E-tasks. Therefore, although T_7 has two assisted tasks T_3 and T_4, T_3 is more prioritized than T_4.

In contrast, rmp values of unrelated tasks also slightly increase in the first 100 generations but drop immediately after a small number of generations. Small rmp values are learned for unrelated tasks to reduce the probability of choosing these task pairs for knowledge transfer. As a result, the negative interactions between them are minimized. The results also showed that T_2 and T_6 have a higher learned rmp to T_8 than other tasks in the first 200 generations. However, the learned rmp values only increased slightly from 0.3 to 0.35-0.4 and dropped soon after. Slightly increasing the rmp values between tasks at early generations helps encourage common knowledge exchange and create a diverse initial population for each task, allowing the algorithm to explore and exploit more potential search areas.

Overall, the results confirmed that the proposed algorithm promoted exchanges between highly similar tasks and limited interactions in pairs of dissimilar tasks.

5) Analyze the Effectiveness of Biased Skill-Factor Inheritance Method: To validate the effectiveness of the proposed inheritance method, we implemented a variant of the proposed algorithm called EME-RI that utilizes random inheritance. The



(a) Task 5 (b) Task 6

(c) Task 7 (d) Task 8

(e) Task 9 (f) Task 10

Fig. 8. The learned mean rmp values.

obtained results from EME-RI on the CEC'17 dataset are presented in Table S-II in the supplementary material. The results showed that EME-BI is better than EME-RI on two complex tasks, T_5 and T_{10}, and similar on other tasks. The biased inheritance method got a 32.8% relative improvement on T_{10} and reached the optimal solution in T_5. This proved the effectiveness of our inheritance method.

Besides, we investigate how the proposed inheritance method works to get good results. A heatmap depicting actual transfer times between tasks if using biased inheritance is shown in Figure 9. The more intense color in the cells, the larger the number of transfers between these pairs of tasks. The results showed that our method promoted transfers from easy tasks to complex tasks rather than vice versa. This helps our algorithms focus more on complex tasks to get better results. The statistics in Table V also showed that our inheritance method reduced the total number of transfers but got more successful transfers in both relative and absolute values. Besides, our inheritance method helps increase 4.6% relative improvement on the successful transfers from easy to complex tasks. It can be concluded that the proposed inheritance method is effective in reducing redundant transfers and promoting successful transfers from easy tasks to complex tasks. It is suitable for practical multitasking applications where tasks vary and some particular tasks need more attention.

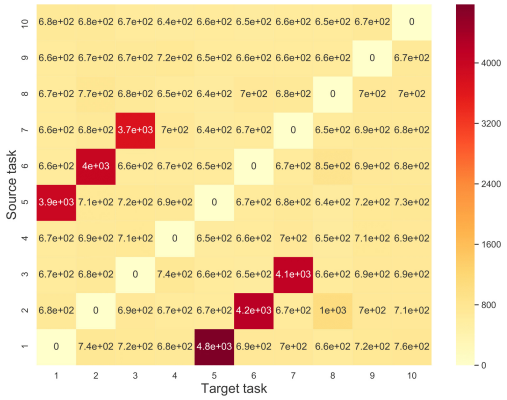


Fig. 9. Actual transfer times between tasks if using biased inheritance.

TABLE V
THE TOTAL NUMBER OF ACTUAL TRANSFERS AND SUCCESSFUL TRANSFERS IF USING TWO DIFFERENT INHERITANCE METHODS

Algorithms	Actual	Successful	Successful E-task-to-C-task
EME-RI	83046	3777	2413
EME-BI	82458(−1%)	3850(+1.9%)	2522(+4.6%)

V. CONCLUSION

This paper proposes an algorithm named EME-BI to solve the many-task optimization problems. In an attempt to avoid the negative transfer phenomenon in existing MFEA, EME-BI adapts the rmp of each task pair based on the quality of the received knowledge. Notably, we design a novel mechanism named BI and incorporate it into the rmp learning scheme. With the ability to prioritize assigning generated offspring to tasks in which they may be higher performing, this mechanism solves the problem of wasting resources existing in previous approaches when tasks with different difficulties are treated equally. Additionally, EME-BI is equipped with ensemble search operators and an adaptive selection; hence it can automatically select the best search operator for a specific task during its evolution. Finally, linear population size reduction is included in EME-BI to enhance the exploitation capacity. To analyze the efficacy of the proposed algorithm, we conduct experiments on various many-task benchmark problems and on a real-world optimization problem called the planar kinematic arm, in combination with an ablation study. Several state-of-the-art many-task algorithms are also compared against EME-BI in terms of solution quality, convergence trend, and computation time. The results illustrated that thanks to the efficiency of its components, EME-BI outperforms other algorithms in all cases.

There are some possible future directions open to investigate, especially when MaTO is one of the most blooming research areas hitherto. Instead of using two search operators, we can employ a versatile pool of these or even ensemble of multiple many-tasking algorithms to drive the evolutionary search of tasks with unknown characteristics. Besides, it can be observed that there are few public real-world many task datasets. Therefore, design and evaluation of existing algorithms seem to be based mainly on artificial benchmarks. We expect that more efforts will be invested on contributing many-task datasets, including not only continuous functions but also real-world discrete/combinatorial problems.

ACKNOWLEDGMENT

This research was funded by Vingroup Innovation Foundation (VINIF) under project code VINIF.2022.DA00183.

REFERENCES

[1] L. Bao, Y. Qi, M. Shen, X. Bu, J. Yu, Q. Li, and P. Chen, "An evolutionary multitasking algorithm for cloud computing service composition," in *World congress on services*. Springer, 2018, pp. 130–144.

[2] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: toward evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2015.

[3] A. Gupta, J. Mańdziuk, and Y.-S. Ong, "Evolutionary multitasking in bi-level optimization," *Complex & Intelligent Systems*, vol. 1, no. 1-4, pp. 83–95, 2015.

[4] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE transactions on cybernetics*, vol. 47, no. 7, pp. 1652–1665, 2016.

[5] J. Lin, H.-L. Liu, K. C. Tan, and F. Gu, "An effective knowledge transfer approach for multiobjective multitasking optimization," *IEEE transactions on cybernetics*, 2020.

[6] K. K. Bali, A. Gupta, Y.-S. Ong, and P. S. Tan, "Cognizant multitasking in multiobjective multifactorial evolution: Mo-mfea-ii," *IEEE transactions on cybernetics*, 2020.

[7] A. T. W. Min, Y.-S. Ong, A. Gupta, and C.-K. Goh, "Multiproblem surrogates: Transfer evolutionary multiobjective optimization of computationally expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 1, pp. 15–28, 2017.

[8] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 1, pp. 44–58, 2017.

[9] A. Gupta, L. Zhou, Y.-S. Ong, Z. Chen, and Y. Hou, "Half a dozen real-world applications of evolutionary multitasking, and more," *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 49–66, 2022.

[10] C. Yang, J. Ding, Y. Jin, and T. Chai, "Offline data-driven multiobjective optimization: Knowledge transfer between surrogates and generation of final solutions," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 3, pp. 409–423, 2019.

[11] R. Chandra, A. Gupta, Y.-S. Ong, and C.-K. Goh, "Evolutionary multi-task learning for modular training of feedforward neural networks," in *International Conference on Neural Information Processing*. Springer, 2016, pp. 37–46.

[12] L. Feng, Y. Huang, L. Zhou, J. Zhong, A. Gupta, K. Tang, and K. C. Tan, "Explicit evolutionary multitasking for combinatorial optimization: A case study on capacitated vehicle routing problem," *IEEE transactions on cybernetics*, 2020.

[13] F. Zhang, Y. Mei, S. Nguyen, K. C. Tan, and M. Zhang, "Multitask genetic programming-based generative hyperheuristics: A case study in dynamic scheduling," *IEEE Transactions on Cybernetics*, 2021.

[14] T. T. B. Huynh, D. T. Pham, B. T. Tran, C. T. Le, M. H. P. Le, A. Swami, and T. L. Bui, "A multifactorial optimization paradigm for linkage tree genetic algorithm," *Information Sciences*, vol. 540, pp. 325–344, 2020.

[15] H. Li, Y.-S. Ong, M. Gong, and Z. Wang, "Evolutionary multitasking sparse reconstruction: Framework and case study," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 733–747, 2018.

[16] F. Shen, J. Liu, and K. Wu, "Evolutionary multitasking fuzzy cognitive map learning," *Knowledge-Based Systems*, vol. 192, p. 105294, 2020.

[17] X. Hao, R. Qu, and J. Liu, "A unified framework of graph-based evolutionary multitasking hyper-heuristic," *IEEE Transactions on Evolutionary Computation*, 2020.

[18] F. Zhang, Y. Mei, S. Nguyen, M. Zhang, and K. C. Tan, "Surrogate-assisted evolutionary multitask genetic programming for dynamic flexible job shop scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 651–665, 2021.

[19] T. B. Thang, N. B. Long, N. V. Hoang, and H. T. T. Binh, "Adaptive knowledge transfer in multifactorial evolutionary algorithm for the clustered minimum routing cost problem," *Applied Soft Computing*, p. 107253, 2021.

[20] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An adaptive archive-based evolutionary framework for many-task optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 3, pp. 369–384, 2019.

[21] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: Mfea-ii," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 69–83, 2019.

[22] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm." in *IJCAI*, 2018, pp. 3870–3876.

[23] L. Zhou, L. Feng, K. C. Tan, J. Zhong, Z. Zhu, K. Liu, and C. Chen, "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE transactions on cybernetics*, vol. 51, no. 5, pp. 2563–2576, 2020.

[24] Z. Liang, X. Xu, L. Liu, Y. Tu, and Z. Zhu, "Evolutionary many-task optimization based on multi-source knowledge transfer," *IEEE Transactions on Evolutionary Computation*, 2021.

[25] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *IEEE Transactions on evolutionary computation*, vol. 13, no. 5, pp. 945–958, 2009.

[26] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms–a survey," *Swarm and evolutionary computation*, vol. 44, pp. 695–711, 2019.

[27] B. Da, Y.-S. Ong, L. Feng, A. K. Qin, A. Gupta, Z. Zhu, C.-K. Ting, K. Tang, and X. Yao, "Evolutionary multitasking for single-objective continuous optimization: Benchmark problems, performance metric, and baseline results," *arXiv preprint arXiv:1706.03470*, 2017.

[28] E. Osaba, A. D. Martinez, and J. Del Ser, "Evolutionary multitask optimization: a methodological overview, challenges and future research directions," *arXiv preprint arXiv:2102.02558*, 2021.

[29] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, "Linearized domain adaptation in evolutionary multitasking," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 1295–1302.

[30] X. Zheng, A. K. Qin, M. Gong, and D. Zhou, "Self-regulated evolutionary multitask optimization," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 16–28, 2019.

[31] C. Wang, J. Liu, K. Wu, and Z. Wu, "Solving multi-task optimization problems with adaptive knowledge transfer via anomaly detection," *IEEE Transactions on Evolutionary Computation*, 2021.

[32] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y.-S. Ong, K.-C. Tan, and A. K. Qin, "Evolutionary multitasking via explicit autoencoding," *IEEE transactions on cybernetics*, vol. 49, no. 9, pp. 3457–3470, 2018.

[33] Q. Shang, L. Zhang, L. Feng, Y. Hou, J. Zhong, A. Gupta, K. C. Tan, and H.-L. Liu, "A preliminary study of adaptive task selection in explicit evolutionary many-tasking," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 2153–2159.

[34] T. L. Thanh, L. V. Cuong, T. B. Thang, and H. T. T. Binh, "Multi-armed bandits for many-task evolutionary optimization," in *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2021, pp. 1664–1671.

[35] S.-H. Wu, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Orthogonal transfer for multitask optimization," *IEEE Transactions on Evolutionary Computation*, 2022.

[36] R.-T. Liaw and C.-K. Ting, "Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 2266–2273.

[37] ——, "Evolutionary manytasking optimization based on symbiosis in biocoenosis," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 4295–4303.

[38] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *2014 IEEE congress on evolutionary computation (CEC)*. IEEE, 2014, pp. 1658–1665.

[39] R. Hinterding, "Gaussian mutation and self-adaption for numeric genetic algorithms," in *Proceedings of 1995 IEEE International Conference on Evolutionary Computation*, vol. 1. IEEE, 1995, p. 384.

[40] X. Zhang, D. Wang, Z. Fu, S. Liu, W. Mao, G. Liu, Y. Jiang, and S. Li, "Novel biogeography-based optimization algorithm with hybrid migration and global-best gaussian mutation," *Applied Mathematical Modelling*, vol. 86, pp. 74–91, 2020.

[41] K. Q. Liang Feng, Y. Y. Abhishek Gupta, Y.-S. O. Eric Scott, and X. Chi, "New mto benchmarks for gecco 2020 competition on evolutionary multi-task optimization," July 2020, available at: http://www.bdsc.site/websites/MTO/index.html.

[42] J.-B. Mouret and G. Maguire, "Quality diversity for multi-task optimization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 2020, pp. 121–129.

[43] V. Vassiliades, K. Chatzilygeroudis, and J.-B. Mouret, "Using centroidal voronoi tessellations to scale up the multidimensional archive of phenotypic elites algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 623–630, 2017.

Huynh Thi Thanh Binh is Associate Professor and Vice Dean of the School of Information and Communication Technology, Hanoi University of Science and Technology. She is Head of Modeling, Simulation and Optimization Lab. Her current research interests include – evolutionary computation, computational intelligence, evolutionary multitasking. She is Associate Editor of the Engineering Applications of Artificial Intelligence Journal, IEEE Transactions on Emerging Topics in Computational Intelligence. She has served as a regular reviewer, a program committee member of numerous prestigious academic journals and conferences such as Applied Soft Computing, Memetic Computing, IEEE Access, IEEE Congress on Evolutionary Computation, Swarm and Evolutionary Computation. . . She is member of IEEE Computational Intelligence Society – Women in Computational Intelligence Committee; Chair of IEEE Computational Intelligence Society Vietnam Chapter; IEEE Asia Pacific Executive committee member; IEEE Asia Pacific Student Activities Committee Chair (2019, 2020).

Le Van Cuong is a Research Assistant and currently working toward the master's degree at the School of Information and Communication Technology, Hanoi University of Science and Technology. His current research interests include computational intelligence, evolutionary multi-tasking. He and his team recently won the First Prize at the CEC2021 Competition on Evolutionary Multi-task Optimization, Multi-task single-objective optimization (MTSOO) and ranked fourth in the CEC2021 competition on Single Objective Bound Constrained Optimization.

Ta Bao Thang is an AI Engineer at Viettel Cyberspace Center, Viettel Group, Vietnam. His current research interests include Evolutionary Computation and Transfer Learning. He has published about 20 papers in prestigious journals and conferences in evolutionary computation and transfer learning. In 2020, he won Second Prize in the Multi-Task Single-Objective Optimization Competition and the Third prize in Electric Vehicle Routing Optimization organized by IEEE WCCI 2020. He and his team recently won the First Prize at the CEC2021 Competition on Evolutionary Multi-task single-objective optimization.

Nguyen Hoang Long is a Research Assistant at the School of Information and Communication Technology, Hanoi University of Science and Technology, Vietnam. His current research interests include evolutionary computation, intelligent computation, and evolutionary multitasking. Long had several publications at prestigious journals and conferences such as Memetic Computing and Congress on Evolutionary Computation. Recently, he and his team won the First Prize at the CEC2021 Competition on Evolutionary Multi-task single-objective optimization.