Evolutionary Multitask Optimization With Adaptive Knowledge Transfer

Hao Xu, A. K. Qin, *Senior Member, IEEE*, and Siyu Xia, *Member, IEEE*

***Abstract*—Evolutionary multitask optimization (EMTO) studies how to simultaneously solve multiple optimization tasks via evolutionary algorithms (EAs) while making the useful knowledge acquired from solving one task to assist solving other tasks, aiming to improve the overall performance of solving each individual task. Recent years have seen a large body of EMTO works based on different kinds of EAs and studying one or more aspects in how to represent, extract, transfer, and reuse knowledge. A key challenge to EMTO is the occurrence of negative knowledge transfer between tasks, which becomes severer when the total number of tasks increases. To address this issue, we propose an adaptive EMTO (AEMTO) framework. This framework can adapt knowledge transfer frequency, knowledge source selection, and knowledge transfer intensity in a synergistic way to make the best use of knowledge transfer, especially when facing many tasks. We implement the proposed AEMTO framework and evaluate our implementation on three suites of MTO problems with 2, 10, and 50 tasks and one real-world MTO problem with 2000 tasks in comparison to several state-of-the-art EMTO methods with certain adaptation strategies regarding knowledge transfer and the single-task optimization counterpart of the proposed method. Experimental results have demonstrated the effectiveness of the adaptive knowledge transfer strategies used in AEMTO and the overall performance superiority of AEMTO.**

***Index Terms*—Adaptive knowledge transfer, evolutionary multitasking, multitask optimization, transfer optimization.**

I. Introduction

NOWADAYS, multitask optimization (MTO) [1], [2] has become a trending research topic in the field of optimization. It studies how to simultaneously solve multiple optimization tasks while making these task-solving processes to assist each other such that the overall performance of solving each task gets improved. MTO assumes some useful common knowledge exists for solving related tasks so that the useful knowledge acquired from solving one task may be used to help solving another task if these two tasks have certain relatedness. In fact, many real-world optimization tasks possess explicit or implicit relatedness [2], [3]. As such, high-quality candidate solutions acquired during solving one task, if transferred and reused, may facilitate the solving process of other related tasks. For example, optimizing key parameter settings in one engineering design task may lead to good solutions that directly function well in another engineering design task when these two tasks have certain commonality [4]. A deep learning model trained on one data set for classification, resulting in optimized network parameters, may perform well when being directly applied to another data set with similar properties to the one used in training [5].

As the most popular family of MTO techniques, evolutionary MTO (EMTO) [2], [6] employs evolutionary algorithms (EAs) as the task solver with knowledge transfer strategies designed under the paradigm of EAs. There exist different ways to represent, extract, transfer, and reuse knowledge and different choices of EAs, leading to different EMTO methods [6]–[9]. A key challenge to EMTO is negative knowledge transfer from unhelpful knowledge sources to a target task, degrading the optimization efficacy of the target task. This challenge becomes severer when the number of tasks involved in an MTO problem gets increased. Some recent works [10]–[13] have attempted to address this issue by designing various adaptive strategies from the aspects of knowledge transfer frequency, knowledge source selection, and knowledge transfer intensity. However, no existing works have considered these three aspects from a synergistic perspective.

In this work, we propose an adaptive EMTO (AEMTO) framework. In AEMTO, each task has an explicit task-specific population which is evolved by an EA for solving the task. As for knowledge transfer between tasks, each task can be regarded as a target task with the other tasks treated as source tasks from which knowledge could be extracted and transferred into the target task. The key innovation of our proposed framework is that it can leverage on *historical knowledge transfer behavior* to online estimate the degrees of helpfulness from different source tasks to a target task and utilize this estimation to:

1) adjust the knowledge transfer frequency w.r.t. a target task *to pursue a good balance between intratask self-evolution (intraSE) and intertask knowledge*

Manuscript received December 16, 2020; revised March 25, 2021 and June 28, 2021; accepted August 11, 2021. Date of publication August 24, 2021; date of current version March 31, 2022. This work was supported by the Australian Research Council (ARC) under Grant LP180100114 and Grant DP200102611. *(Corresponding authors: A. K. Qin; Siyu Xia.)*

Hao Xu was with the Department of Computing Technologies, Swinburne University of Technology, Hawthorn, VIC 3122, Australia. He is now with the School of Automation, Southeast University, Nanjing 210018, China, and also with the Key Laboratory of Measurement and Control of Complex Systems of Engineering, Ministry of Education, Nanjing 210096, Jiangsu, China (e-mail: haoxu@seu.edu.cn).

A. K. Qin is with the Department of Computing Technologies, Swinburne University of Technology, Hawthorn, VIC 3122, Australia (e-mail: kqin@swin.edu.au).

Siyu Xia is with the School of Automation, Southeast University, Nanjing 210018, China, and also with the Key Laboratory of Measurement and Control of Complex Systems of Engineering, Ministry of Education, Nanjing 210096, Jiangsu, China (e-mail: xsy@seu.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TEVC.2021.3107435.

Digital Object Identifier 10.1109/TEVC.2021.3107435

transfer (interKT), and once knowledge transfer occurs;
2) select the most helpful knowledge sources (from many candidates) for a target task *to make the best use of multiple available knowledge sources*;
3) determine the knowledge transfer intensities from those selected source tasks to the target task *to allow historically more helpful source tasks w.r.t. a target task to give stronger help to that target task*.

As such, AEMTO can adapt knowledge transfer frequency, knowledge source selection, and knowledge transfer intensity in a synergistic way, aiming to make the best use of knowledge transfer, especially when facing many tasks to be simultaneously solved. To the best of our knowledge, this has never been done in previous works.

We implement the proposed AEMTO framework as follows.
1) *Task Solver:* A classic differential evolution (DE) algorithm [14].
2) *Knowledge Representation and Reusing:* We use a binomial crossover operation [12] to reuse the knowledge acquired from a source task, where knowledge is represented as good candidate solutions generated in the task-solving process.
3) *Balancing Self-Evolution and Knowledge Transfer:* We use the population member update rates per generation due to intraSE and interKT as their rewards, respectively, estimate online the degree of the helpfulness of knowledge transfer as the ratio of the historically accumulated rewards from knowledge transfer and self-evolution, and use this estimate to adjust the transfer probability to adaptively control the frequency of knowledge transfer.
4) *Balancing Multiple Knowledge Sources:* We use the success rate of reusing the knowledge from a source task to help solving a target task as the reward, estimate online the degree of helpfulness from a source task to a target task as the historically accumulated reward by using the source task to help the target task, and apply the probability matching (PM) strategy [15] based on the estimated degrees of the helpfulness of all candidate source tasks w.r.t. a target task to calculate source task selection probabilities.
5) *Determining Transfer Intensity Per Knowledge Source:* We apply the stochastic universal selection (SUS) [16] based on source task selection probabilities to select source tasks for a target task and determine the amount of knowledge (i.e., the number of promising candidate solutions) to be acquired from each of the selected source tasks, apply the roulette wheel selection (RWS) [16] to select the determined number of promising candidate solutions from the population of each of the selected source tasks, and use all of the selected promising candidate solutions as the knowledge to be transferred to the target task.

We evaluate our implementation on three MTO test suites composed of MTO problems with 2, 10, and 50 component tasks as well as a real-world MTO problem with 2000 component tasks in comparison to several state-of-the-art EMTO methods which employ certain adaptation strategies regarding knowledge transfer. We also perform an ablation study on the proposed adaptive knowledge transfer strategies and conduct a parameter sensitivity analysis to provide an insight into the proposed method. Experimental results demonstrate the effectiveness of the proposed adaptive knowledge transfer strategies and the overall performance superiority of the proposed method over the compared methods.

The remainder of this article is organized as follows. Section II describes the background and related work. The proposed AEMTO framework and its implementation are elaborated in Section III. Section IV reports and discusses experimental results. The conclusions with some planned future work are given in Section V.

II. Background and Related Work

A. MTO Problem Definition

An MTO problem typically contains two or more optimization problems (i.e., component tasks) to be solved. In this work, we assume all component tasks are single-objective real-valued unconstrained minimization problems. Specifically, for an MTO problem with T component tasks, denoted by $\tau_i, i = 1, \ldots, T$, we aim at solving all T tasks simultaneously, denoted by $\boldsymbol{x}_i^* = \text{argmin}_{\boldsymbol{x} \in \boldsymbol{\mathcal{X}}_i} f_i(\boldsymbol{x})$, where $f_i : \boldsymbol{\mathcal{X}}_i \rightarrow \Re$ denotes the objective function of task τ_i, where $\boldsymbol{\mathcal{X}}_i$ represents the D_i-dimensional real-valued search space of task τ_i. Notably, if some component tasks have distinct search spaces, a unified search space [6], [17] can be used for encoding different search spaces into a common one in which knowledge transfer is conducted. When evaluating a candidate solution w.r.t. a specific task, its representation can be decoded from the unified search space to the task-specific one to enable evaluation. There exist some other ways to handle this. For example, Feng *et al.* [8] proposed using explicit representation mapping models, built on promising solutions from any two tasks with distinct search spaces, to transform candidate solutions between the two tasks so that the promising candidate solution from one task can be utilized, after representation mapping, by another task with a different search space.

B. EMTO

EMTO [2], [3] stands for a family of MTO approaches which employ EAs as the task solver and utilize knowledge transfer strategies designed under the paradigms of EAs. In terms of knowledge transfer strategies, existing EMTO approaches mainly differ in one or more of the following aspects.

1) How to Represent Knowledge: There exist different ways to represent the knowledge generated from a task-solving process. For example, a straightforward way is to represent as knowledge the promising solutions generated from when solving a task [6], [10], [12], [13], [18]–[20]. Such knowledge may directly facilitate solving some other tasks if the promising candidate solutions to these tasks and those represented as knowledge are somewhat similar. Another popular way is to build a generative model on promising solutions and use it to represent knowledge [21]–[23], which is usually more compact and powerful than the straightforward way mentioned above at the expense of additional computational cost for model building. Further, the effective search directions obtained during a task-solving process have emerged as

a new way of knowledge representation [9], [24], [25]. It is particularly useful when the global optima of two tasks are far away from each other and accordingly the first two ways may not work.

2) How to Transfer Knowledge: Knowledge transfer typically involves the target task (that acquires knowledge) and the corresponding source tasks (that provide knowledge), where how to transfer knowledge between source and target tasks is a key research question. This question has two main aspects: 1) how much and how often knowledge will be transferred out from each source task and 2) how often knowledge will be transferred into the target task. Existing works have used the estimated degrees of helpfulness from different source tasks to a target task to determine how much (i.e., intensity) knowledge should be transferred out from each source. The degrees of helpfulness from different sources to a target may dynamically vary as the search proceeds. There exist different ways to estimate them, e.g., via the effectiveness of using knowledge from the source task to help solving the target task [13], [26]–[29] and via the relatedness between source and task tasks [10], [12], [21], [30], [31]. As for how often (i.e., frequency) knowledge will be transferred out from the source task or into the target task, there exist two commonly used ways in existing works, i.e., using a fixed value [12], [21], [29], [31] and adapting the transfer frequency on the fly [10], [13].

3) How to Extract and Reuse Knowledge: To extract a certain amount (defined via transfer intensity) of knowledge from a source task, different strategies have been used in existing works, e.g., randomly sampling promising candidate solutions from the current population, an archive of promising candidate solutions generated in the past or a generative model that represents knowledge [21], [26], [28]. For a target task, there exist different ways to reuse knowledge transferred from source tasks, i.e., incorporating knowledge into its own task-solving process. For example, knowledge, represented as promising candidate solutions, can be directly injected into the current population of the target task [13], [21], [26], [28]. Also, different evolutionary operations can be used to reuse knowledge by integrating it with the selected population members of the target task, e.g., crossover operators [10], [12], the dimension-wise search strategy [32], and the opposition-based learning with the simulated binary crossover operator [33]. Recently, Zhou *et al.* [34] proposed an adaptive configuration of the crossover operators method to pursue more effective knowledge reuse. Liang *et al.* [22] proposed a two-stage model-based adaptive knowledge transfer operation for better balancing exploration and exploitation during the search.

C. Evolutionary Many-Task Optimization

In many existing works, MTO problems under study typically contain two or three tasks. However, in practical scenarios, the number of optimization problems to be solved at the same time may go far beyond three. Recently, many-task optimization has emerged as a trending topic, which considers solving MTO problem that contains more than three [12], [26], [29] and even thousands of tasks [35]. In many-task optimization, negative knowledge transfer may more likely occur due to a high chance of the presence of less related tasks. Therefore, adaptive knowledge transfer becomes indispensable at the expense of the increased computational cost to learn adaptation-related parameters. In the following, we review some existing many-task optimization methods.

In some works, many-task optimization problems with more than three but up to ten component tasks were studied. For example, Chen *et al.* [12] and Huang *et al.* [30] estimated the intertask relatedness by calculating the Kullback–Leibler divergence between the generative models built on the populations of two tasks. Shang *et al.* [29] used the number of promising candidate solutions, transferred from a source task to a target task, which have higher quality than the average quality of the population members of the target task to estimate the degree of the helpfulness of the source task. In these works, the knowledge transfer frequency is set to a fixed value and only one source task with the highest relevance or helpfulness is selected from which knowledge is extracted and transferred, which prevents exploiting useful knowledge from multiple source tasks to the best.

Liaw and Ting [26] considered many-task optimization problems with 30 component tasks and proposed an evolution of biocoenosis through symbiosis (EBS) method to address them. In EBS, the knowledge transfer frequency is adapted according to the ratio of the times that the best-so-far solution is improved by the offspring of the task itself and the other tasks. The knowledge transfer intensities for different source tasks are set to be equal. Also, Liaw and Ting [13] proposed a symbiosis in biocoenosis optimization (SBO) method which treats each population as a biocoenosis and defines six types of symbiosis to determine the transfer rates of different source tasks. Tang *et al.* [18] proposed a group-based multifactorial EA (GMFEA) method which uses the bisecting K-means to cluster tasks into different groups based on the Manhattan distances between some solutions selected from any two tasks. Knowledge transfer is enabled for the tasks belonging to the same group. GMFEA was applied to solve many-task optimization problems with up to 100 tasks, showing good scalability.

Recently, Mouret and Maguire [35] proposed a multitask multidimensional archive of phenotypic elites (MAP-elites) method to deal with MTO problems with thousands of tasks. As indicated in [35], one of the major drawbacks of this method is that it requires the presence of at least a few hundreds of tasks. With fewer tasks, there lacks enough diversity to allow this method to work properly. Further, this method relies on some manually designed task descriptor to represent a task. The relatedness of two tasks is defined as the distance between the descriptors of two tasks. It may risk at an inaccurate estimation of task relatedness due to inappropriate task descriptors, leading to less helpfulness between two identified related tasks.

III. Proposed Method

A. Framework

The proposed AEMTO framework is illustrated in Fig. 1. In AEMTO, each component task in an MTO problem, denoted

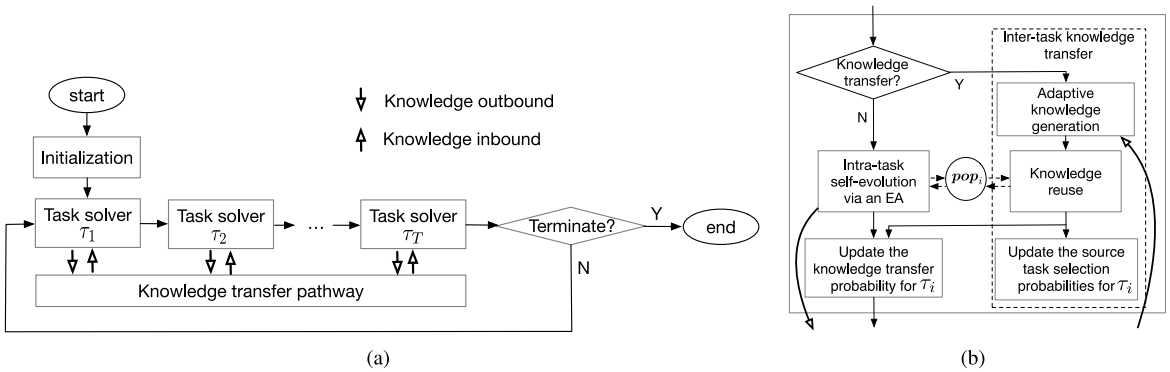


Fig. 1. Illustration of the AEMTO framework: (a) overall diagram and (b) individual task solver for each task $\tau_i, i \in \{1, \ldots, T\}$.

by $\tau_i, i \in \{1, \ldots, T\}$, has its own population which is evolved via intraSE facilitated by interKT to seek its solution $\boldsymbol{x}_i^*$. As shown in Fig. 1(a), AEMTO's problem-solving process starts from an initialization step, responsible for initializing the population and evaluating the quality of its members for each task as well as initializing various variables used in the algorithm. Then, all T tasks are solved in a generation-by-generation manner until some termination criterion is met. In each generation, every task is independently solved to some extent, one after another, under the help of the knowledge transferred in from the other tasks via the knowledge transfer pathway. Meanwhile, the knowledge generated from its own problem-solving process, represented as promising candidate solutions in this work, is transferred out via the knowledge transfer pathway to help solving the other tasks. It is worth noting that this work is focused on a centralized implementation of the AEMTO framework and thus we chose to solve all component tasks in sequence. However, the AEMTO framework may natively support a decentralized implementation which is one of our ongoing works mentioned in Section V.

Fig. 1(b) illustrates how each task solver works in every generation. A task solver has two key modules, i.e., intraSE and interKT, which are selectively executed, according to an adaptive knowledge transfer probability, to evolve a population to seek the solution of the corresponding task. The intraSE module employs an EA to evolve the population for one generation. The interKT module first generates knowledge by selecting from $T-1$ source tasks some most helpful ones based on each source task's historical behavior of helping the target task under consideration, measured via source task selection probabilities, and extracting a certain amount of knowledge from each of the selected source tasks in proportional to the degrees of their helpfulness, estimated via their selection probabilities. Then, the generated knowledge is utilized to update the current population of the target task via some knowledge reuse operation. After knowledge reuse, the effectiveness of a source task helping a target task in the current generation is quantitatively estimated and used to online update this source task's knowledge selection probability w.r.t. the target task. The effectiveness of applying intraSE or interKT to help the task-solving process in the current generation is quantitatively estimated and used to online update the knowledge transfer probability.

In the following, we will detail the strategies used for adapting the knowledge transfer probability and source task selection probabilities and describe our implementation of the AEMTO framework in this work.

B. Adaptation Strategies

As described above, each task solver has two kinds of probabilities to be adjusted on the fly as the number of generations increases, i.e., the knowledge transfer probability that pursues a good balance between intraSE and interKT, denoted by p_i^{tsf}, and source task selection probabilities that pursue making the best use of multiple available knowledge sources which are represented as a probability vector, denoted by $\boldsymbol{p}_i^{\text{sel}} = \{p_i^j | j = 1, \ldots, T; j \neq i\}$, with each element referring to one of the $T-1$ source tasks w.r.t. task τ_i. In this work, we employ the PM strategy [15] to adaptively update these probabilities, which has been widely used for adaptive operator allocation in EAs [36]. The basic idea of PM is to update the probabilities of selecting different operations to make them in proportion to their historically accumulated rewards received after being successfully applied, aiming to maximize the expected reward of operation selection.

1) Adaptation of the Knowledge Transfer Probability: In every generation, solving each task $\tau_i, i \in \{1, \ldots, T\}$ requires choosing to apply intraSE or interKT according to the current value of p_i^{tsf}. To adjust this probability via the PM strategy, we use q_i^s and q_i^o, with superscripts s and o denoting "self" and "other," for estimating the expected quality of applying intraSE and interKT, respectively. After either intraSE or interKT is applied, its effectiveness of helping the task-solving process in the current generation is measured via reward r which is then used to update q_i^* as follows:

$$q_i^* = \alpha q_i^* + (1-\alpha) r \tag{1}$$

where * indicates s or o and α denotes the quality update coefficient. There exist different ways to define r. In this work, it is defined as the population member update rate, i.e., the percentage of population members replaced by the candidate solutions generated by intraSE or interKT, in the following way:

$$r = \frac{|\boldsymbol{pop}_{i,g}| - |\boldsymbol{pop}_{i,g} \cap \boldsymbol{pop}_{i,g+1}|}{|\boldsymbol{pop}_{i,g}|} \tag{2}$$

where $|\cdot|$ denotes the cardinality of a set.

At the end of each generation, p_i^{tsf} is recalculated, based on the updated q_i^s or q_i^o, as follows:

$$p_i^{\text{tsf}} = p_{lb}^{\text{tsf}} + \frac{q_i^o}{q_i^o + q_i^s + \epsilon}\left(p_{ub}^{\text{tsf}} - p_{lb}^{\text{tsf}}\right). \tag{3}$$

In the above equation, ϵ is a small positive number used to avoid division by zero. p_{lb}^{tsf} and p_{ub}^{tsf} define the lower and upper bounds of p_i^{tsf}. The lower bound is used to maintain the chance of a task being helped by other tasks via knowledge transfer throughout the task-solving process. The upper bound is used to make the population of each task to get reasonably evolved via interSE without overly relying on external knowledge. This may not only benefit a task's own solving process but also lead to the increased chance for a task to produce more useful knowledge to better help solving the other tasks.

2) Adaptation of Source Task Selection Probabilities: Each task $\tau_i, i \in \{1, \ldots, T\}$ has $T-1$ source tasks. In interKT, source task selection probabilities are used to select some most helpful source tasks from $T-1$ candidates and meanwhile determine the amount of knowledge to be extracted from the selected source tasks so that historically more helpful sources may offer stronger help by providing more knowledge. To adjust such probabilities via the PM strategy, we use a quality vector, denoted by $\boldsymbol{q}_i^{\text{sel}} = \{q_i^j | j = 1, \ldots, T; j \neq i\}$, to estimate the expected helpfulness (quality) of each source task w.r.t. target task τ_i.

After knowledge reuse is applied, the helpfulness from each source task to target task τ_i in the current generation is measured via reward r_i^j, received from using source task τ_j to help target task τ_i via knowledge reuse, which is then used to update q_i^j as follows:

$$q_i^j = \alpha q_i^j + (1-\alpha) r_i^j. \tag{4}$$

There exist different ways to define r_i^j. In this work, we define it as the percentage of knowledge from source task τ_j which succeeds in replacing the population member of target task τ_i, i.e., $r_i^j = ns_i^j / n_i^j$. Here, n_i^j and ns_i^j denote the total number of promising candidate solutions extracted as knowledge from τ_j and those of them contributing to generating candidate solutions via knowledge reuse, which replace some population members of τ_i, respectively. Notably, r_i^j is set to zero if τ_j is not selected for helping τ_i. After $\boldsymbol{q}_i^{\text{sel}}$ is updated, $\boldsymbol{p}_i^{\text{sel}} = \{p_i^j | j = 1, \ldots, T; j \neq i\}$ is updated as follows:

$$p_i^j = p_{\min} + (1 - (T-1)\cdot p_{\min}) \frac{q_i^j}{\sum_{l=1, l\neq i}^{T} q_i^l + \epsilon}. \tag{5}$$

In the above equation, ϵ is a small positive number used to avoid division by zero. $p_{\min}$ represents the minimum selection probability of each source task, which is used to ensure each source task to have a chance of being selected throughout the task-solving process.

C. Implementation

The implementation of the AEMTO framework in this work is described in Algorithm 1, where the details about interKT and intraSE modules are described in Algorithms 2 and 3, respectively.

Algorithm 1 AEMTO Method

Input:
$\{N_i\}_{i=1}^T$: The population size of each of T tasks $\tau_i, i = 1, \ldots T$.
α: The quality update coefficient.
$p_{lb}^{\text{tsf}}, p_{ub}^{\text{tsf}}$: The lower and upper bounds of the knowledge transfer probability.
$p_{\min}$: The minimum source task selection probability.
CR, F: The parameters used in intraSE.
$G_{\max}$: The maximum number of generations.
Output:
$\{\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \ldots, \boldsymbol{x}_T^*\}$: The solutions found for T tasks.

for $i = 1$ to T **do**
  Randomly initialize $\boldsymbol{pop}_{i,1}$ of size N_i for task τ_i.
  Evaluate each individual in $\boldsymbol{pop}_{i,1}$ on task τ_i.
  $\boldsymbol{q}_i^{\text{sel}} = \{q_i^j = 0 \mid j = 1, \ldots, T; j \neq i\}$.
  $\boldsymbol{p}_i^{\text{sel}} = \{p_i^j = \frac{1}{T-1} \mid j = 1, \ldots, T; j \neq i\}$.
  $q_i^s = 0, q_i^o = 0$.
  $p_i^{\text{tsf}} = \frac{p_{lb}^{\text{tsf}} + p_{ub}^{\text{tsf}}}{2}$.
end for
$g = 1$
while $g \leq G_{\max}$ **do**
  for $i = 1$ to T **do**
    if $rand(0,1) \leq p_i^{\text{tsf}}$ **then**
      $[\boldsymbol{pop}_{i,g+1}, \boldsymbol{q}_i^{\text{sel}}, \boldsymbol{p}_i^{\text{sel}}, r] \leftarrow$ interKT($\{\boldsymbol{pop}_{i,g}\}_{i=1}^T$, $\boldsymbol{q}_i^{\text{sel}}$, $\boldsymbol{p}_i^{\text{sel}}$, $p_{\min}$)# Alg.2
      $q_i^o = \alpha q_i^o + (1-\alpha) r$
    else
      $[\boldsymbol{pop}_{i,g+1}, r] \leftarrow$ intraSE($\boldsymbol{pop}_{i,g}$, CR, F)# Alg.3
      $q_i^s = \alpha q_i^s + (1-\alpha) r$
    end if
    # Update the knowledge transfer probability
    $p_i^{\text{tsf}} = p_{lb}^{\text{tsf}} + \frac{q_i^o}{q_i^o + q_i^s + \epsilon}(p_{ub}^{\text{tsf}} - p_{lb}^{\text{tsf}})$
  end for
  $g = g + 1$
end while
for $i = 1$ to T **do**
  Find the best individual $\boldsymbol{x}_i^*$ with the minimum objective function value in $\boldsymbol{pop}_{i,g}$.
end for

As described in Algorithm 1, the MTO task-solving process starts from initializing and evaluating a population of size N_i, denoted by $\boldsymbol{pop}_{i,1}$, as well as initializing several variables related to adaptation strategies for each task $\tau_i, i \in \{1, \ldots, T\}$. After that, all T tasks are iteratively solved generation by generation until a prespecified maximum number of generations, denoted by $G_{\max}$, is reached. In each generation $g, g \in \{1, \ldots, G_{\max}\}$, T tasks are addressed one by one. When solving τ_i, a random number uniformly distributed in $[0, 1]$ is generated and compared with p_i^{tsf} to determine whether intraSE or interKT is executed. After either of intraSE and interKT is executed, the population of τ_i gets updated to $\boldsymbol{pop}_{i,g+1}$ and the reward received from this execution is generated as r. For interKT, $\boldsymbol{q}_i^{\text{sel}}$ and $\boldsymbol{p}_i^{\text{sel}}$ also get updated. Then, q_i^o or q_i^s is updated, provided interSE or interKT is executed, which is used to update p_i^{tsf} at the end of the generation. Eventually, the best individual in the final population of each

Algorithm 2 interKT

Input:
$\{\boldsymbol{pop}_{i,g}\}_{i=1}^{T}$: The populations for T tasks at generation g, where the size of $\boldsymbol{pop}_{i,g}$ is N_i.
$\boldsymbol{q}_i^{\text{sel}}$: The source task selection quality vector for task τ_i.
$\boldsymbol{p}_i^{\text{sel}}$: The source task selection probability vector for task τ_i.
$p_{\min}$: The minimum source task selection probability.
Output:
$\boldsymbol{pop}_{i,g+1}$: The updated population for task τ_i.
$\boldsymbol{q}_i^{\text{sel}}$: The updated selection quality vector for task τ_i.
$\boldsymbol{p}_i^{\text{sel}}$: The updated selection probability vector for task τ_i.
r: The reward received from applying interKT.

Apply SUS based on $\boldsymbol{p}_i^{\text{sel}}$ to determine n_i^j promising solutions to be extracted from each source task $\tau_j, j \in \{1, ..., T\}$ and $j \neq i$, subjected to $\sum_{j=1;j\neq i}^{T} n_i^j = N_i$.
$\boldsymbol{K} = \emptyset$; # Initialize the knowledge pool
for $j = 1$ to T **do**
  if $j \neq i$ and $n_i^j \neq 0$ **then**
    Apply RWS based on objective function values to select n_i^j individuals of higher quality from $\boldsymbol{pop}_{j,g}$ and add them into $\boldsymbol{K}$.
  end if
end for
# Knowledge reuse
$ns_i^j = 0, j = 1, ..., T$ and $j \neq i$ # Initialize counters
$k = 1$
for $j = 1$ to T **do**
  if $j \neq i$ and $n_i^j \neq 0$ **then**
    for $s = 1$ to n_i^j **do**
      $\boldsymbol{v} = \boldsymbol{K}(k)$; $\boldsymbol{x} = \boldsymbol{pop}_{i,g}(k)$
      $\boldsymbol{c}$ = binomial_crossover($\boldsymbol{v}$, $\boldsymbol{x}$)
      if $f_i(\boldsymbol{c}) < f_i(\boldsymbol{x})$ **then**
        $\boldsymbol{pop}_{i,g+1}(k) = \boldsymbol{c}$
        $ns_i^j = ns_i^j + 1$
      else
        $\boldsymbol{pop}_{i,g+1}(k) = \boldsymbol{x}$
      end if
      $k = k + 1$
    end for
  end if
end for
# Update source task selection quality values
for $j = 1$ to T **do**
  if $j \neq i$ and $n_i^j \neq 0$ **then**
    $q_i^j = \alpha q_i^j + (1-\alpha)\frac{ns_i^j}{n_i^j}$
  end if
end for
# Update source task selection probabilities
for $j = 1$ to T **do**
  if $j \neq i$ **then**
    $p_i^j = p_{\min} + (1 - (T-1)p_{\min})\frac{q_i^j}{\sum_{l=1;l\neq i}^{T} q_i^l + \epsilon}$
  end if
end for
$r = \frac{N_i - |\boldsymbol{pop}_{i,g} \cap \boldsymbol{pop}_{i,g+1}|}{N_i}$ # Calculate the reward

task $\tau_i, i \in \{1, \ldots, T\}$, denoted by $\boldsymbol{x}_i^*$, is output as the solution to the MTO problem.

Algorithm 2 details the implementation of interKT for target task $\tau_i, i \in \{1, \ldots, T\}$, in generation $g, g \in \{1, \ldots, G_{\max}\}$. First, the SUS [16] based on the current $\boldsymbol{p}_i^{\text{sel}}$ is applied to determine the amount of knowledge, denoted by n_i^j (representing the number of promising candidate solutions), to be extracted from each source task $\tau_j, j \in \{1, \ldots, T\}$ and $j \neq i$, subjected to $\sum_{j=1;j\neq i}^{T} n_i^j = N_i$, as shown in Fig. 2. By doing so, historically more helpful source tasks that correspond to higher selection probabilities may provide stronger help (via more promising candidate solutions) to the target. Then, for each of the selected source tasks with nonzero n_i^j, the RWS [16] based on objective function values is applied to select n_i^j individuals of higher quality from the population of τ_j, and add them into a knowledge pool denoted by $\boldsymbol{K}$. Next, knowledge reuse is carried out by applying a binomial crossover operation [12] to $\boldsymbol{K}$ and $\boldsymbol{pop}_{i,g}$. If any resulting offspring $\boldsymbol{c}$ replaces its parent in $\boldsymbol{pop}_{i,g}$ and thus enter $\boldsymbol{pop}_{i,g+1}$, then the success counter w.r.t. source task τ_j, denoted by ns_i^j, from which another parent of $\boldsymbol{c}$ comes is increased by one. After knowledge reuse is executed, rewards $r_i^j, j = 1, \ldots, T$, and $j \neq i$ are calculated and used to update $\boldsymbol{q}_i^{\text{sel}}$ and then $\boldsymbol{p}_i^{\text{sel}}$. Finally, the reward r received from applying interKT is calculated.

Algorithm 3 intraSE

Input:
$\boldsymbol{pop}_{i,g}$: The population of size N_i for task τ_i at generation g.
CR, F: The parameters in DE.
Output:
$\boldsymbol{pop}_{i,g+1}$: The updated population for task τ_i.
r: The reward received from applying intraSE.

Apply DE/rand/1/bin [14] algorithm
$\boldsymbol{pop}_{i,g+1} \leftarrow$ DE($\boldsymbol{pop}_{i,g}$, CR, F)
$r = \frac{N_i - |\boldsymbol{pop}_{i,g} \cap \boldsymbol{pop}_{i,g+1}|}{N_i}$ # Calculate the reward

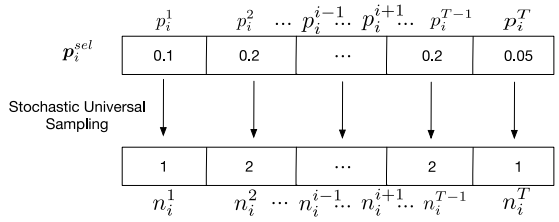


Fig. 2. Illustration of selecting knowledge from source tasks $\tau_j, j = 1, \ldots, T$, and $j \neq i$ to target task τ_i by applying the SUS based on $\boldsymbol{p}_i^{\text{sel}}$, resulting in n_i^j candidate solutions selected from each source task $\tau_j, j \in \{1, \ldots, T\}$ and $j \neq i$.

Algorithm 3 details the implementation of intraSE for target task $\tau_i, i \in \{1, \ldots, T\}$, in generation $g, g \in \{1, \ldots, G_{\max}\}$. In this work, we employ a classic DE algorithm, i.e., DE/rand/1/bin [14] with two key parameters CR and F, as the EA. After it is applied to evolve the population of τ_i from the current generation to the next one, the reward r of applying intraSE is calculated.

In our implemented AEMTO method, we employ a unified real-valued search space of dimension size $\max\{D_i\}_{i=1}^{T}$ to accommodate different component tasks with their real-valued search space of different dimension sizes, i.e., $D_i, i = 1, \ldots, T$, as did in [6]. We set the range of the unified search space to $[0, 1]^{D_{\max}}$, and use a linear transformation to enable the knowledge transfer in the unified search space

and the evaluation in the task-specific one, as described in Section II-A. For the binomial crossover used in knowledge reuse, the crossover rate is set to a random number uniformly distributed in [0.1, 0.9], as did in [12]. Among the four parameters related to the adaptation strategies, two less sensitive ones revealed by the experimental study, i.e., α and p_{lb}^{tsf}, are set to the fixed values of 0.3 and 0.05, respectively. The other two, i.e., p_{ub}^{tsf} and $p_{\min}$, need to be manually specified with sensitivity analysis on them provided in Section IV-G.

D. Time Complexity Analysis

The worst-case time complexity of AEMTO is $O(T*N*D_{\max})+O(T*N*\log N+N*T^2)$ per generation, where T and N denote the total number of tasks and the equal population size of each task, respectively. The first part mainly corresponds to intraSE and the second part mainly for interKT.

In contrast, the SBO [13] method has slightly lower worst-case time complexity, i.e., $O(T*N*D_{\max})+O(T*N*\log N+T^2)$ per generation. However, as the task number T increases while the equal population size of each task N is fixed at a certain value, AEMTO and SBO tend to have the same time complexity. We do not compare AEMTO with multifactorial EA (MFEA) [6] and its extended version MFEA2 [10] which are two well-known EMTO methods. Different from AEMTO, both MFEA and MFEA2 employ the genetic algorithm (GA) as the task solver and also make use of a single population to deal with all tasks. Further, MFEA2 natively suffers from high time complexity due to its computationally expensive model learning component. Notably, there is a significant gain in terms of optimality guaranty in MFEA2 that other EMTO methods with heuristic adaptation strategies cannot provide. In practice, we could reduce the computation time of MFEA2 by decreasing its model learning frequency.

IV. Experimental Study

We evaluate the proposed AEMTO method on three suites of MTO problems with 2, 10, and 50 component tasks, respectively, and a real-world MTO problem with 2000 component tasks to demonstrate that:

1) controlling the balance between intraSE and interKT via transfer frequency adaptation leads to the improved performance of solving each component task (Section IV-C);
2) selecting the most helpful source tasks and determining the amount of knowledge to be acquired from the selected sources to best help a target task via transfer source and intensity adaptation leads to performance improvement (Section IV-D);
3) AEMTO outperforms several state-of-the-art EMTO methods with certain adaptation strategies (including MaTDE [12], SBO [13], and MFEA2 [10]) and its single-task optimization counterpart, i.e., AEMTO with knowledge transfer disabled, denoted by STO (Sections IV-E and IV-F);
4) AEMTO has some sensitive parameters in its adaptation strategies which should be set in a problem-dependent way (Section IV-G).

In the following, we will first describe the MTO problems and experimental setup used in our study, and then report and discuss experimental results.

A. Test Problems

Test suite 1 consists of nine MTO problems with two component tasks per problem, proposed in [37], where the two component tasks in an MTO problem may have different dimension sizes (i.e., 25D and 50D). The degree of task relatedness between two component tasks, measured by the degree of global optima intersection and the intertask similarity, varies across different MTO problems. For example, "CI," "PI," and "NI" denote the complete, partial, and no intersection of the global optima of two component tasks, respectively. "HS," "MS," and "LS" denote the high, medium, and low intertask similarity between two component tasks, respectively. More details about this test suite can be found in the online supplementary material.

Test suite 2 contains five MTO problems, where each MTO problem has ten component tasks with some purposely designed task relatedness. The ten component tasks in an MTO problem may have different dimension sizes (i.e., 25D and 50D). The first problem was proposed in [12] for studying the MaTDE method, denoted by the MaTDE-problem. It contains four easy tasks (denoted by E-task) and six complex tasks (denoted by C-task), where four out of the six C-tasks are designed to be able to be assisted by some E-task(s) via the intersection of their global optima. More details about this MTO problem can be found in the online supplementary material. The other four problems are designed by us following the way described in [13] and [18]. Specifically, we employ five commonly used test problems at 50D, i.e., Rosenbrock, Ackley, Schwefel, Griewank, and Rastrgin, to create two groups of component tasks. Each group contains all five test problems with their global optima generated via a random perturbation from a common centroid, where global optimum shifting is applied to move the original global optimum of a test problem to the newly generated one. The centroids of two groups are separated away from each other. By varying the amount of perturbation from zero and small to medium and large, four MTO problems are generated, denoted by ManyTask10-zero, ManyTask10-small, ManyTask10-medium, and ManyTask10-large. As such, the tasks in the same group are more related to each other than to those from different groups. Meanwhile, the within-group task relatedness decreases as the perturbation amount increases. More details about these four problems can be found in the online supplementary material.

Test suite 3 contains six MTO problems with 50 component tasks at 50D, taken from the test bed used in the CEC 2019 Competition on EMTO.[1] Each of the six MTO problems is generated on top of a unique basic test problem, where 50 component tasks therein are created by applying different global optimum shifting vectors and search space rotation

[1] http://www.bdsc.site/websites/MTO_competiton_2019/MTO_Competition_CEC_2019.html

matrices to the basic test problem. More details about this test suite can be found in the online supplementary material.

A real-word MTO problem with 2000 component tasks at 50D is created under a planar kinematic arm control problem scenario [35], as illustrated in Fig. 7(a). This control problem aims at finding the optimal angle of each joint in the arm, denoted by $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_D\}$, to minimize the distance between the tip position $\boldsymbol{p}_d$ and the target position $\boldsymbol{T}$, where the total number of joints, denoted by D, corresponds to the problem dimension size. The total length of the arm is L and each segment between two adjacent joints has the equal length. The objective function of this optimization (minimization) problem is defined as $f(\boldsymbol{\alpha}, [L, \alpha_{\max}]) = ||\boldsymbol{p}_D - \boldsymbol{T}||$, where $||.||$ denotes the Euclidean distance calculation and $\alpha_{\max}/D$ denotes the upper bound of the angle of each joint. Here, each decision variable $\alpha_i, i \in \{1, \ldots, D\}$ takes a real value in [0, 1]. By setting L and $\alpha_{\max}$ to different values, we can obtain different problems. The details of the problem definition can be referred to in [35]. In this work, we set the planar target $\boldsymbol{T}$ at [0.5, 0.5] and create 2000 problems with their L and $\alpha_{\max}$ values evenly sampled in [0, 1] via centroidal Voronoi tesselation [38] as did in [35].

B. Experimental Setup

The parameter settings for the AEMTO method are summarized as follows.

1) The population size of each task: $N_i = 100, i = 1, \ldots, T$.
2) The quality update coefficient: $\alpha = 0.3$.
3) The lower and upper bounds of the knowledge transfer probability: $p_{lb}^{\text{tsf}} = 0.05$ and $p_{ub}^{\text{tsf}} = 0.7$.
4) The minimum source task selection probability: $p_{\min} = p_{\text{base}}/(T-1)$ with $p_{\text{base}} = 0.3$.
5) DE (rand/1/bin) parameters: $CR = 0.9$ and $F = 0.5$.
6) The maximum number of generations: $G_{\max} = 1000$.

To make a fair comparison, MaTDE and SBO use the same setting for the population size of each task. Also, they use DE (rand/1/bin) as the basic task solver with relevant parameter settings kept same to those used in the AEMTO method. The other parameter settings for MaTDE and SBO are the same as those used in their original papers [12] and [13]. For MFEA2, the population size is set to $100 * T$ because it employs a single population to deal with T tasks. The parameter settings for its GA-based task solver are the same as those used in [10]. For the single-task optimization counterpart of AEMTO, i.e., STO, all of its parameter settings are the same as those used in AEMTO besides the knowledge transfer probability which is set to zero. For all compared methods, the maximum number of generations is set to 1000.

In our experiments, all component tasks in an MTO problem are started to be solved simultaneously. Each method in comparison is executed for 20 independent runs with different random seeds. All methods are implemented in C++ and run on a Linux server with an Intel CPU at 2.30 GHz. The mean and standard deviation of the best-achieved objective function error values (FEVs) over 20 runs are used for evaluating the accuracy of solving a component task. Here, the FEV is defined as the difference between the objective function value of the best solution found so far and that of the global optimum. The convergence map in terms of the mean of the best-achieved FEVs over 20 runs is used for evaluating the time efficiency of solving a component task. We employ Wilcoxon's rank-sum test [39] at a significant level of 0.05 to compare the proposed method with each of the other methods and report the statistical comparison results in the form of "tie/win/lose" (abbrev. "t/w/l").

C. Effectiveness of Transfer Frequency Adaptation

To evaluate the effectiveness of AEMTO's transfer frequency adaptation strategy which aims to seek a good balance between intraSE and interKT, we compare AEMTO with AEMTO w/o adaptive knowledge transfer (aTsf) which fixes the transfer probability for each task at a certain value, e.g., $p_i^{\text{tsf}} = 0.3, i = 1, \ldots, T$, in this study. The comparison is made on test suite 1, where there is only one source task for any target task in each of its MTO problems and thus the influence from the selection and utilization of multiple source tasks can be avoided. The comparison results w.r.t. each MTO problem in test suite 1, measured by the mean and standard deviation of the best-achieved FEVs over 20 runs for each component task, are reported in Table S5 in the online supplementary material. It can be observed that AEMTO performs better than AEMTO (w/o aTsf) on 12 out of 18 component tasks in total, similar on five tasks and worse on only one task.

Fig. 3 illustrates how the adaptation strategy performs on two selected MTO problems in test suite 1, i.e., problem 1 (CI+HS) and problem 6 (PI+LS) shown in the first and second rows, respectively. The figures in the first column compare the changing curves of the mean transfer probability value (averaged over 20 runs) as the number of generations increases in fixed-value and adaptive cases. The figures in the second and third columns illustrate the changing curves of the accumulated population member update rate as the number of generations increases w.r.t. AEMTO, AEMTO (w/o aTsf), and STO for two component tasks, respectively. The accumulated population member update rate in generation g for task τ_i is defined by $\sum_{k=1}^{g} (\#\text{updated\_pop\_member}_k^i)/(G_{\max} * N_i)$. Such a curve depicts the degree of population evolution, reflecting the effect of adapting the transfer probability on facilitating the optimization process. The figures in the last column compare the convergence curves of AEMTO, AEMTO (w/o aTsf), and STO for each component task.

For problem 1, its two component tasks are highly related. It can be observed from Fig. 3(b) and (c) that both AEMTO and AEMTO (w/o aTsf) have their accumulated population member update rate curves surpassing that of STO, indicating that knowledge transfer can facilitate the optimization process. Also, the curve of AEMTO surpasses that of AEMTO (w/o aTsf), revealing the advantage of adaptive knowledge transfer. The convergence curves of AEMTO, AEMTO (w/o aTsf), and STO for each component task, as shown in Fig. 3(d), further confirm the superiority of AEMTO. There is an interesting observation from Fig. 3(a) that even though the two component tasks are highly related, the transfer probability for each

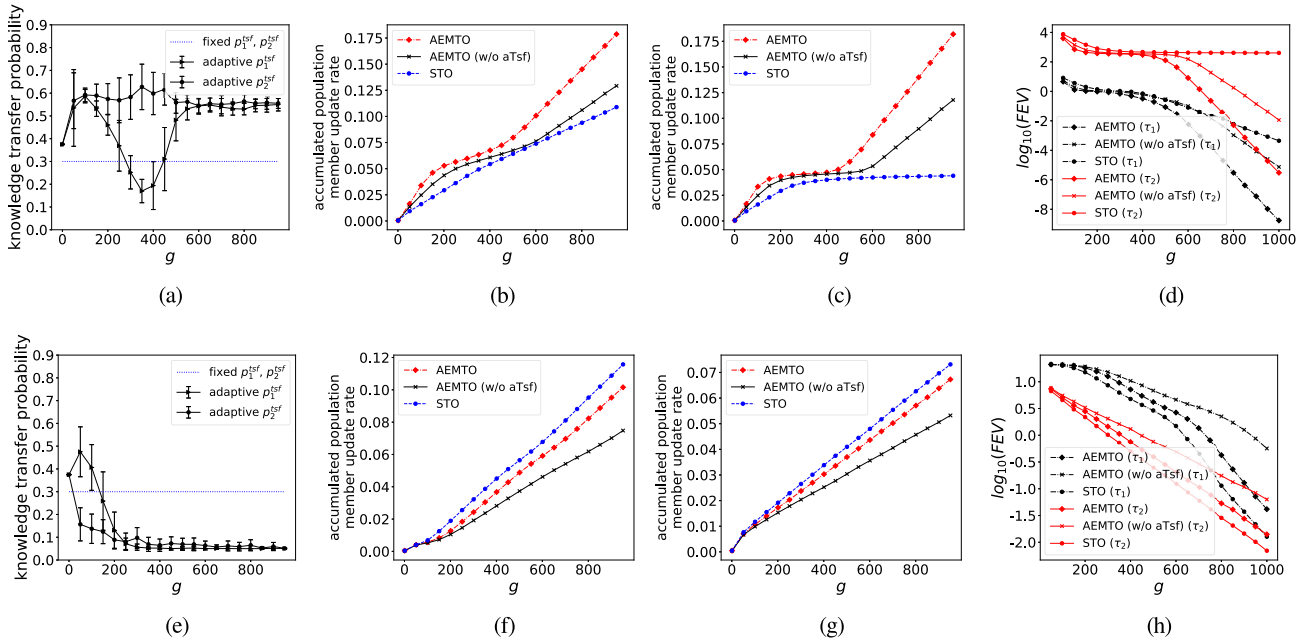


Fig. 3. Behavior of knowledge transfer probability adaptation on MTO problems 1 (CI+HS; 1st row) and 6 (PI+LS; 2nd row) in test suite 1: (a) and (e) comparison of the changing curves of the mean transfer probability value (averaged over 20 runs) as the number of generations g increases in fixed (at the value of 0.3) and adaptive cases; (b) and (f) comparison of the changing curves of the accumulated population member update rate as g increases w.r.t. AEMTO, AEMTO (w/o aTsf), and STO for component task τ_1, and (c) and (g) for component task τ_2; and (d) and (h) comparison of the convergence curves of AEMTO, AEMTO (w/o aTsf), and STO on τ_1 and τ_2, respectively.

TABLE I
COMPARISON OF AEMTO (W/O ASEL) WITH AEMTO (CONTROL METHOD) ON EACH OF THE FIVE MTO PROBLEMS IN TEST SUITE 2, I.E., MATDE-PROBLEM, MANYTASK10-ZERO, MANYTASK10-SMALL, MANYTASK10-MEDIUM, AND MANYTASK10-LARGE, IN TERMS OF THE "T/W/L" SCORES (OBTAINED VIA WILCOXON'S RANK-SUM TEST) OVER THE TEN COMPONENT TASKS

Problem	MaTDE-problem	ManyTask10 (zero)	ManyTask10 (small)	ManyTask10 (medium)	ManyTask10 (large)
AEMTO (w/o aSel)	2/0/8	0/0/10	0/1/9	0/0/10	3/1/6

task may not always maintain high during optimization. This is because different tasks may have distinct difficulty levels and thus the different times needed for finding useful knowledge. During some period of optimization, particularly in the early stage of the search, the promising solutions found by one task may be far away from its own global optimum and thus can hardly help the other task even though these two tasks are highly related, in terms of having the same global optimum. Accordingly, the transfer probability for the other task may decrease. However, when the promising solutions found by one task get closer to its global optimum as optimization proceeds, they may more likely help the other task, leading to the increased transfer probability for the other task. Such a pattern is shown in Fig. 3(a).

For problem 6, the relatedness of its two component tasks is low. Accordingly, it can be observed from Fig. 3(f)–(h) that both AEMTO and AEMTO (w/o aTsf) perform worse than STO, indicating the occurrence of negative knowledge transfer. In comparison, AEMTO performs better than AEMTO (w/o aTsf) due to its ability to somewhat prevent ineffective knowledge transfer. Fig. 3(e) shows that the transfer probabilities for both component tasks eventually decrease to a very small value due to low task relatedness. The fluctuation in the early stage for τ_1 is a reasonable phenomenon due to the dynamics of the optimization process in which the promising solutions generated from when solving one task during some period may temporally help solving the other task and thus lead to the occasionally increased transfer probability.

D. Effectiveness of Transfer Source and Intensity Adaptation

To evaluate the effectiveness of AEMTO's transfer source and intensity adaptation strategy that aims at making the best use of multiple candidate source tasks to allow historically more helpful source tasks to provide stronger help, we compare the performances of AEMTO and AEMTO w/o adaptive source task selection (aSel) which fixes the selection probability of each source task at $1/(T-1)$ on test suite 2 which contains five MTO problems with ten component tasks. The comparison results in terms of the mean and standard deviation of the best-achieved FEVs over 20 runs for each component task in each MTO problem are reported in Tables S6 and S7 in the online supplementary material and summarized in Table I. It can be observed from the "t/w/l" scores that AEMTO consistently outperforms AEMTO (w/o aSel) across all five MTO problems.

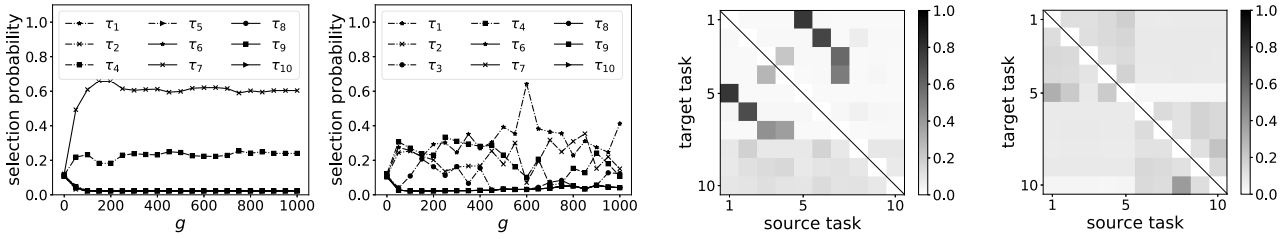


Fig. 4. Behavior of source task selection probability adaptation. (a) and (b) Comparison of the changing curves of the selection probabilities of different source tasks for component task τ_3 in MaTDE-problem and component task τ_5 in ManyTask10-medium, respectively, as the number of generations g increases. (c) and (d) Heatmap representation of the mean selection probability averaged over all generations and all 20 runs w.r.t. each source task for each component task in the MaTDE-problem and ManyTask10-medium, respectively. Note: A target task does not treat itself as its own source task and thus the diagonals in (c) and (d) are crossed out.

Fig. 4 shows the behavior of source task selection probability adaptation. Fig. 4(a) and (b) illustrates the changing curves, as the number of generations increases, of the selection probabilities of different source tasks for component task τ_3 in MaTDE-problem and component task τ_5 in ManyTask10-medium, respectively. For τ_3 in MaTDE-problem, the two tasks τ_4 and τ_7 which are known to be highly related to τ_3, maintain high selection probabilities during optimization while all of the other seven tasks have very low selection probabilities. Also, τ_4 with its global optimum partially intersected with that of τ_3 always has lower selection probabilities than τ_7 with the same global optimum that τ_3 has in the unified search space. For τ_5 in ManyTask10-medium, the tasks in the same group as τ_5, i.e., τ_1, τ_2, τ_3, and τ_4, have reasonably large selection probabilities while all of the other tasks in another group maintain very low selection probabilities. The fluctuation of selection probabilities for τ_1, τ_2, τ_3, and τ_4 reflects their varying relatedness to τ_5, caused by the population traversing the different regions of the search space.

Fig. 4(c) and (d) illustrates the heatmap representation of the mean selection probability averaged over all generations and all 20 runs w.r.t. each source task for each component target task in MaTDE-problem and ManyTask10-medium, respectively. In both problems, those source tasks which are known to have higher relatedness to the target task correspond to higher mean selection probabilities. Particularly, Fig. 4(d) shows two blocks of tasks with higher selection probabilities, corresponding to the two predefined task groups with high intragroup and low intergroup task relatedness.

The above observations from Fig. 4 verify the fact that the adaptive source task selection strategy can successfully identify and make use of the tasks related to a target task as its source tasks. To further demonstrate both transfer frequency adaptation and transfer source and intensity adaptation are essential in AEMTO, we further compare AEMTO with AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel) as well as STO on four selected component tasks from the MTO problems in test suite 2, i.e., τ_2 and τ_8 in MaTDE-problem, τ_5 in ManyTask10-zero, and τ_5 in ManyTask10-medium, in terms of the changing curve of the accumulated population member update rate and the convergence curve of the average best-achieved FEV. As shown in Fig. 5, AEMTO consistently outperforms all of the other compared methods for both curves. Also, further exclusion of transfer frequency adaptation leads to severer performance degradation in all cases. Compared to STO, the performances of AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel) are sometimes better and sometimes worse. For example, for τ_2 in MaTDE-problem, as shown in Fig. 5(a)–(e), STO outperforms AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel). This is because the studied component task τ_2 in this case is easy to solve via STO and therefore if knowledge transfer is not properly controlled, like in AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel), it will cause the negative effect on solving τ_2. For τ_5 in ManyTask10-zero and τ_5 in ManyTask10-medium, knowledge transfer causes the positive effect on solving them in both adaptive and nonadaptive cases, as shown in Fig. 5(b)–(f) and 5(c) and (g). For τ_5 in ManyTask10-medium, AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel) outperform STO in Fig. 5(d) and vice versa in Fig. 5(h). This is because the population member update rate is related but not directly corresponds to the effectiveness of finding the global optimum. It may happen that the population is noticeably evolving, but the best found candidate solution is not improved like in this case.

E. Comparison on MTO Problems With 2, 10, and 50 Tasks

In this section, we compare AEMTO with several state-of-the-art EMTO methods which employ certain adaptation strategies regarding knowledge transfer, i.e., MaTDE, SBO, and MFEA2, on test suites 1, 2, and 3 which contains multiple MTO problems with 2, 10, and 50 component tasks, respectively, where AEMTO (w/o aTsf and aSel) and STO are also included for comparison as contrast.

The comparison results in terms of the mean and standard deviation of the best-achieved FEVs over 20 runs for each component task in each MTO problem from the three test suites are reported in Tables S8–S19 in the online supplementary material and summarized in Table II. It can be observed from the "t/w/l" scores in Table II that AEMTO consistently outperforms the other methods on most of the component tasks from all MTO problems in the three test suites. The last row in Table II indicates the average ranking of each method in comparison over all 368 component tasks, where all of the compared methods are ranked on the basis of

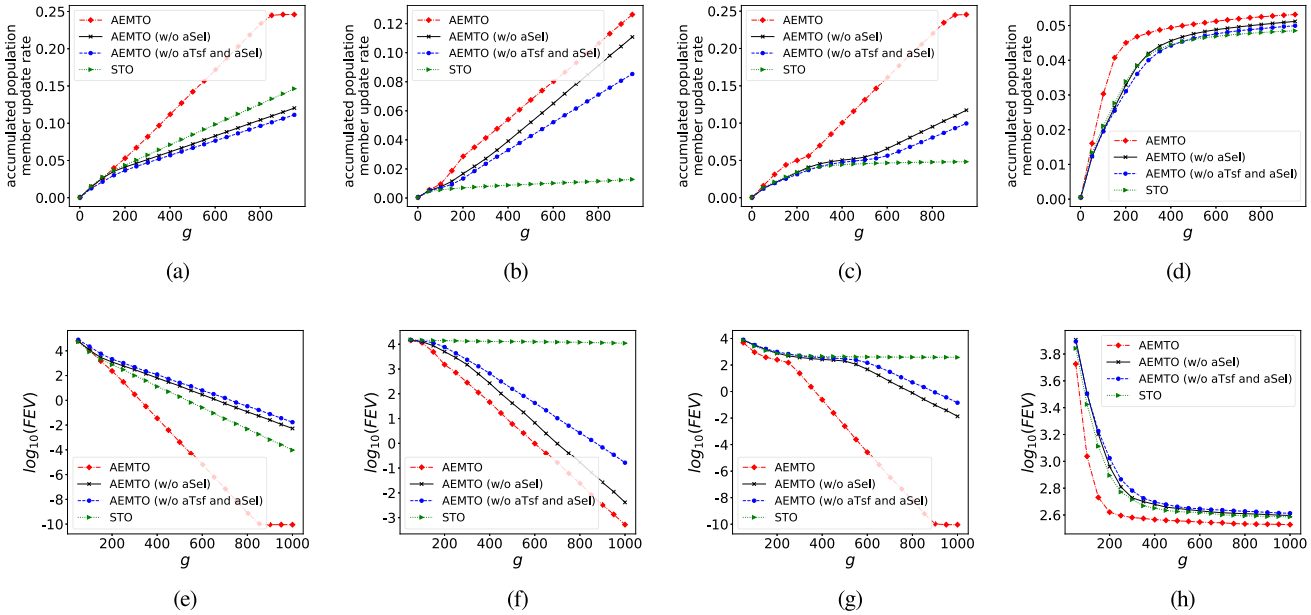


Fig. 5. Comparison of AEMTO with AEMTO (w/o aSel) and AEMTO (w/o aTsf and aSel) as well as STO on four selected component tasks from the MTO problems in test suite 2, i.e., (a) and (e) for τ_2 in MaTDE-problem, (b) and (f) for τ_8 in MaTDE-problem, (c) and (g) for τ_5 in ManyTask10-zero, and (d) and (h) for τ_5 in ManyTask10-medium, in terms of the changing curve of the accumulated population member update rate (the first row) and the convergence curve of the average best-achieved FEV (the second row) as the number of generations g increases.

TABLE II
COMPARISON OF AEMTO WITH MATDE, SBO, AND MFEA2 AS WELL AS AEMTO (W/O ATSF AND ASEL) AND STO ON EACH OF THE THREE TEST SUITES IN TERMS OF THE "T/W/L" SCORES (OBTAINED VIA WILCOXON'S RANK-SUM TEST) OVER ALL COMPONENT TASKS FROM ALL MTO PROBLEMS IN A SPECIFIC TEST SUITE. THE LAST ROW INDICATES THE AVERAGE RANKING OF EACH COMPARED METHOD OVER ALL 368 COMPONENT TASKS FROM ALL THREE TEST SUITES

Test suite	AEMTO	AEMTO (w/o aTsf and aSel)	MaTDE	SBO	MFEA2	STO
1	–	4/3/11	7/2/9	4/3/11	0/6/12	6/4/8
2	–	3/3/44	6/0/44	5/1/44	0/7/43	8/3/39
3	–	56/7/237	131/9/160	53/11/236	0/100/200	51/6/243
Average ranking	1.78	3.90	2.77	4.39	3.59	4.56

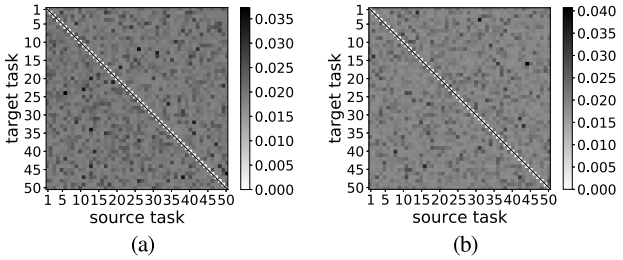


Fig. 6. Heatmap representation of the mean selection probability averaged over all generations and all 20 runs w.r.t. each source task for each component (target) task in two selected MTO problems in test suite 3, i.e., (a) problem 2 and (b) problem 6.

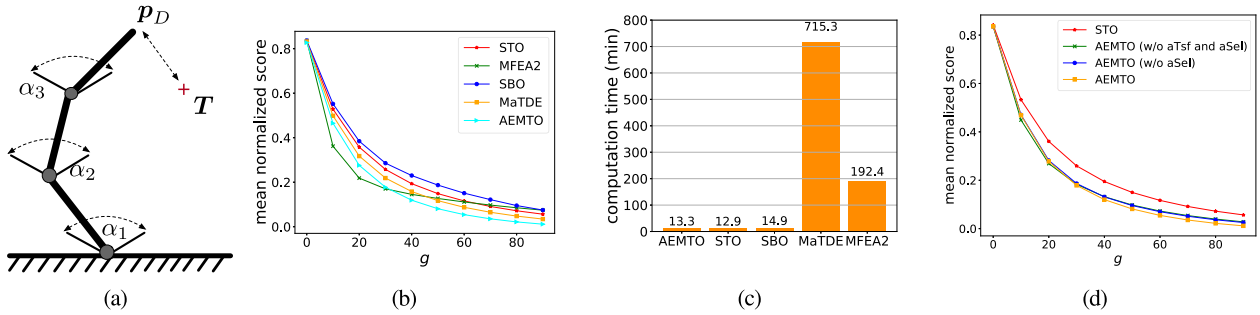
the average best-achieved FEV in the ascending order on each component task. The ranking superiority of AEMTO over the other methods is noticeable.

To further reveal the source task selection behavior in AEMTO when dealing with many tasks, Fig. 6 illustrates the heatmap representation of the mean selection probability averaged over all generations and all 20 runs w.r.t. each source task for each component task (as the target task) in two selected MTO problems in test suite 3, i.e., problems 2 and 6. It can be observed that for each target task there exist multiple source tasks with higher selection probabilities which have offered help to solve the target task.

F. Comparison on Real-World MTO Problem With 2000 Tasks

We further evaluate the performance of AEMTO for dealing with a real-world MTO problem with 2000 component tasks, as detailed in Section IV-A, in comparison to MaTDE, SBO, and MFEA2 as well as AEMTO (w/o aSel), AEMTO (w/o aTsf and aSel), and STO. Different from the previous experiments, the population size for each component task is set to 20 and the maximum number of generations is set to 100 for each method in comparison. The other parameter settings are all kept the same as those specified in Section IV-B.

Fig. 7(b) compares AEMTO with MaTDE, SBO, MFEA2, and STO in terms of the convergence curve of the mean normalized score, defined by $\bar{f} = 1/T \sum_{i=1}^{T}([f_i - f_i^{\min}]/[f_i^{\max} - f_i^{\min}])$, where $f_i^{\min}$ and $f_i^{\max}$ represent the minimum and maximum best-achieved FEVs on component task i, respectively, among all compared methods and all execution runs, which was proposed in [6] to evaluate



(a) (b) (c) (d)

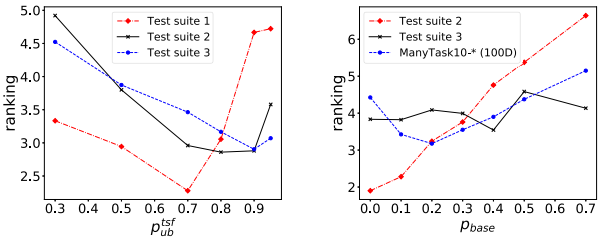
Fig. 7. (a) Illustration of the planar kinematic arm control problem [35]. (b) Comparison of AEMTO with MaTDE, SBO, MFEA2, and STO in terms of the convergence curve of the mean normalized score as the number of generations g increases. (c) Comparison of the computation times (in minutes) of AEMTO, MaTDE, SBO, MFEA2, and STO. (d) Comparison of AEMTO with AEMTO (w/o aSel), AEMTO (w/o aTsf and aSel), and STO in terms of the convergence curve of the mean normalized score.

the overall performance of a method across many component tasks. It can be observed that AEMTO outperforms the other methods in comparison. Fig. 7(c) compares the computation time (in minutes) of the methods compared in Fig. 7(b). It can be observed that AMETO has the smallest computation time among all EMTO methods in comparison, which is only slightly larger than that of STO. The computation time of SBO is very close to that of AEMTO, matching the analysis for the scenario where the population size for each task is far smaller than the total number of competent tasks. Both MaTDE and MFEA2 have significantly larger computation times due to their involved model learning components, where MaTDE is more computationally expensive because its model learning process is more sophisticated. Fig. 7(d) compares AEMTO with AEMTO (w/o aSel), AEMTO (w/o aTsf and aSel), and STO in terms of the convergence curve of the mean normalized score, revealing that source task selection probability adaptation greatly contributes to the performance advantage of AEMTO and transfer probability adaptation contributes little when dealing with a large number of tasks.

G. Parameter Sensitivity Analysis

In this section, we analyze the sensitivity of two parameters used in AEMTO's adaptation strategies, i.e., p_{ub}^{tsf} and p_{base} that is equivalent to $(T-1)*p_{\min}$. The other two parameters used in AEMTO's adaptation strategies are prefixed with no need of being manually specified, as discussed in Section III-C. Specifically, we comprehensively evaluate the performance of AEMTO on different MTO problems under varying parameter settings, and find an overall best parameter setting, i.e., $p_{ub}^{\text{tsf}} = 0.7$ and $p_{\text{base}} = 0.3$. Then, we fix one of them at the found optimal value and vary another to study its sensitivity, where p_{ub}^{tsf} and p_{base} are tested under $\{0.3, 0.5, 0.7, 0.8, 0.9, 0.95\}$ and $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7\}$, respectively.

Fig. 8(a) shows the average rankings of different tested parameter values of p_{ub}^{tsf} over all component tasks from all MTO problems in test suites 1, 2, and 3, respectively, where the performances of AEMTO under $p_{ub}^{\text{tsf}} = \{0.3, 0.5, 0.7, 0.8, 0.9, 0.95\}$ are ranked on the basis of the average best-achieved FEV in the ascending order on each component task. For test suites 1 and 2, the best setting of p_{ub}^{tsf} is 0.7. For test suite 3, the best setting of p_{ub}^{tsf} is 0.9. This



(a) (b)

Fig. 8. Parameter sensitivity analysis results: (a) upper bound of knowledge transfer probability p_{ub}^{tsf} on test suites 1, 2, and 3, and (b) minimum source task selection probability p_{base} on test suites 2, 3, and ManyTask10-* (100D) which denotes the collection of ManyTask10-zero, ManyTask10-small, ManyTask10-medium, and ManyTask10-large problems from test suite 2 with all component tasks augmented to 100D. The average rankings of different tested parameter values over all component tasks from a specific set of MTO problems are used for the analysis.

observation indicates that setting p_{ub}^{tsf} to a value below 1.0 is favored by all test cases and a reasonable value range to try for p_{ub}^{tsf} is between 0.7 and 0.9.

Fig. 8(b) shows the average rankings of different tested parameter values of p_{base} over all component tasks from all MTO problems in test suite 2, test suite 3, and ManyTask10-* (100D), respectively. ManyTask10-* (100D) denotes the collection of ManyTask10-zero, ManyTask10-small, ManyTask10-medium, and ManyTask10-large problems from test suite 2 with all component tasks augmented to 100D. It is created as a test set with increased complexity for component tasks due to dimensionality augmentation. Here, test suite 1 is not used because the MTO problems therein contain only two component tasks and thus no source task selection is needed. It can be observed that the best setting of p_{base} varies much in different test cases, i.e., 0.0 for test suite 2, 0.2 for ManyTask10-*, and 0.4 for test suite 3. Since MTO problems in test suite 2 have purposely defined component task relatedness which is relatively easy to be identified, the selection probabilities of those tasks highly related to a target task may grow high in the early stage of optimization without risking at premature vanishing. As such, setting p_{base} to a nonzero value may cause negative knowledge transfer from unrelated tasks, especially in the late stage of optimization, leading to performance degradation. However, it is more often in practice

that intrinsic task relatedness may not be easily identified, e.g., due to high task complexity like ManyTask10-* (100D) or large problem scale-like test suite 3. In such cases, setting p_{base} to a nonzero value may prevent the selection probabilities of those tasks highly related to a target task from premature vanishing so that they may have the chance of gradually increasing during optimization. In general, a reasonable value range to try for p_{base} is between 0.1 and 0.4.

V. Conclusion and Future Work

We proposed an AEMTO framework capable of adaptively adjusting knowledge transfer frequency to pursue a good balance between intraSE and interKT, selecting the most helpful knowledge sources among many candidates, and determining the transfer intensities of the selected sources based on the estimated degrees of their helpfulness. The proposed AEMTO framework was implemented and evaluated on three test suites of MTO problems with the small and medium numbers of component tasks as well as a real-world MTO problem with 2000 component tasks. We performed an ablation study to demonstrate the effectiveness of transfer frequency adaptation and transfer source and intensity adaptation in AEMTO, compared the performance of AEMTO with several state-of-the-art EMTO methods which employ certain adaptation strategies as well as the single-task optimization counterpart of AEMTO to demonstrate the superiority of AEMTO, and analyzed the sensitivity of two key parameters used in AEMTO's adaptation strategies.

Currently, we are investigating a decentralized implementation of AEMTO,where each task is solved on a unique computing unit and interKT is enabled via information communication among different computing units, aiming at improving scalability so that the running time of AEMTO may no longer significantly increase with the increased task number. Further, we will study the effects of different knowledge representation methods on AEMTO. Moreover, we plan to investigate using AEMTO to solve multiobjective optimization problems [40].

References

[1] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task Bayesian optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2004–2012.

[2] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on transfer optimization: Because experience is the best teacher," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 51–64, Feb. 2018.

[3] Y.-S. Ong and A. Gupta, "Evolutionary multitasking: A computer science view of cognitive multitasking," *Cogn. Comput.*, vol. 8, no. 2, pp. 125–142, 2016.

[4] J. Liang *et al.*, "Evolutionary multi-task optimization for parameters extraction of photovoltaic models," *Energy Convers. Manag.*, vol. 207, Mar. 2020, Art. no. 112509.

[5] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Proc. Int. Conf. Artif. Neural Netw.*, 2018, pp. 270–279.

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

[7] L. Feng *et al.*, "An empirical study of multifactorial PSO and multifactorial DE," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 921–928.

[8] L. Feng *et al.*, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3457–3470, Sep. 2019.

[9] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, "Linearized domain adaptation in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 1295–1302.

[10] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 69–83, Feb. 2020.

[11] Y.-W. Wen and C.-K. Ting, "Parting ways and reallocating resources in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 2404–2411.

[12] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An adaptive archive-based evolutionary framework for many-task optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 3, pp. 3870–3876, Jun. 2020.

[13] R.-T. Liaw and C.-K. Ting, "Evolutionary manytasking optimization based on symbiosis in biocoenosis," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 4295–4303.

[14] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[15] D. Thierens, "An adaptive pursuit strategy for allocating operator probabilities," in *Proc. 7th Annu. Conf. Genet. Evol. Comput.*, 2005, pp. 1539–1546.

[16] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proc. 2nd Int. Conf. Genet. Algorithms*, vol. 206, 1987, pp. 14–21.

[17] Y. Yuan, Y.-S. Ong, A. Gupta, P. S. Tan, and H. Xu, "Evolutionary multitasking in permutation-based combinatorial optimization problems: Realization with TSP, QAP, LOP, and JSP," in *Proc. IEEE Region 10 Conf. (TENCON)*, 2016, pp. 3157–3164.

[18] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm," in *Proc. IJCAI*, vol. 4, 2018, pp. 369–384.

[19] X. Zheng, A. K. Qin, M. Gong, and D. Zhou, "Self-regulated evolutionary multi-task optimization," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 16–28, Feb. 2020.

[20] Q. Chen, X. Ma, Z. Zhu, and Y. Sun, "Evolutionary multi-tasking single-objective optimization based on cooperative co-evolutionary memetic algorithm," in *Proc. IEEE 13th Int. Conf. Comput. Intell. Security (CIS)*, 2017, pp. 197–201.

[21] B. Da, A. Gupta, and Y.-S. Ong, "Curbing negative influences online for seamless transfer evolutionary optimization," *IEEE Trans. Cybern.*, vol. 49, no. 12, pp. 4365–4378, Dec. 2018.

[22] Z. Liang, W. Liang, X. Xu, L. Liu, and Z. Zhu, "A two stage adaptive knowledge transfer evolutionary multi-tasking based on population distribution for multi/many-objective optimization," 2020. [Online]. Available: arXiv:2001.00810.

[23] A. Gupta and Y.-S. Ong, "Multitask knowledge transfer across problems," in *Memetic Computation: The Mainspring of Knowledge Transfer in a Data-Driven Optimization Era*. Cham, Switzerland: Springer, 2019, pp. 83–92.

[24] J. Yin, A. Zhu, Z. Zhu, Y. Yu, and X. Ma, "Multifactorial evolutionary algorithm enhanced with cross-task search direction," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2019, pp. 2244–2251.

[25] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 44–58, Feb. 2019.

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

[27] E. Osaba, A. D. Martinez, A. Galvez, A. Iglesias, and J. Del Ser, "dMFEA-II: An adaptive multifactorial evolutionary algorithm for permutation-based discrete optimization problems," 2020. [Online]. Available: arXiv:2004.06559.

[28] M. Gong, Z. Tang, H. Li, and J. Zhang, "Evolutionary multitasking with dynamic resource allocating strategy," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 858–869, Oct. 2019.

[29] Q. Shang *et al.*, "A preliminary study of adaptive task selection in explicit evolutionary many-tasking," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2019, pp. 2153–2159.

[30] S. Huang, J. Zhong, and W. Yu, "Surrogate-assisted evolutionary framework with adaptive knowledge transfer for multi-task optimization," *IEEE Trans. Emerg. Topics Comput.*, early access, Oct. 10, 2019, doi: 10.1109/TETC.2019.2945775.

[31] J. Zhang, W. Zhou, X. Chen, W. Yao, and L. Cao, "Multisource selective transfer framework in multiobjective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 424–438, Jun. 2020.

[32] X. Ma, Q. Chen, Y. Yu, Y. Sun, L. Ma, and Z. Zhu, "A two-level transfer learning algorithm for evolutionary multitasking," *Front. Neurosci.*, vol. 13, p. 1408, Jan. 2020.
[33] Y. Yu, A. Zhu, Z. Zhu, Q. Lin, J. Yin, and X. Ma, "Multifactorial differential evolution with opposition-based learning for multi-tasking optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2019, pp. 1898–1905.
[34] L. Zhou *et al.*, "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2563–2576, May 2021.
[35] J.-B. Mouret and G. Maguire, "Quality diversity for multi-task optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2020, pp. 121–129.
[36] C. Igel and M. Kreutz, "Operator adaptation in evolutionary computation and its application to structure optimization of neural networks," *Neurocomputing*, vol. 55, nos. 1–2, pp. 347–361, 2003.
[37] B. Da *et al.*, "Evolutionary multitasking for single-objective continuous optimization: Benchmark problems, performance metric, and baseline results," 2017. [Online]. Available: arXiv:1706.03470.
[38] V. Vassiliades, K. Chatzilygeroudis, and J.-B. Mouret, "Using centroidal voronoi tessellations to scale up the multi-dimensional archive of phenotypic elites algorithm," 2016. [Online]. Available: arXiv:1610.05729.
[39] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[40] Y. Yuan *et al.*, "Evolutionary multitasking for multiobjective continuous optimization: Benchmark problems, performance metrics and baseline results," 2017. [Online]. Available: arxiv.abs/1706.02766.

Hao Xu received the B.Eng. degree in automation and the M.Eng. degree in pattern recognition and intelligence system from Southeast University, Nanjing, China, in 2018 and 2021, respectively.

His current research interests include multitask optimization, machine learning, and parallel computing.

A. K. Qin (Senior Member, IEEE) received the B.Eng. degree from Southeast University, Nanjing, China, in 2001, and the Ph.D. degree from Nanyang Technology University, Singapore, in 2007.

From 2007 to 2017, he was with the University of Waterloo, Waterloo, ON, Canada; INRIA Grenoble Rhône-Alpes, Montbonnot-Saint-Martin, France; and RMIT University, Melbourne, VIC, Australia. He joined Swinburne University of Technology, Hawthorn, VIC, Australia, in 2017, where he is currently a Professor. He is currently the Director of Swinburne Intelligent Data Analytics Lab, the Deputy Director of Swinburne Space Technology and Industry Institute, and the Program Lead of Swinburne Data Science Research Institute. His major research interests include machine learning, evolutionary computation, computer vision, remote sensing, services computing, and pervasive computing.

Dr. Qin was a recipient of the 2012 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award. He is currently the Chair of the IEEE Computational Intelligence Society (CIS) Neural Networks Technical Committee and the Vice-Chair of the IEEE CIS Emergent Technologies Task Force on "Multitask Learning and Multitask Optimization."

Siyu Xia (Member, IEEE) received the B.E. and M.S. degrees in automation engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2000 and 2003, respectively, and the Ph.D. degree in pattern recognition and intelligence system from Southeast University, Nanjing, in 2006.

He is currently an Associate Professor with the School of Automation, Southeast University. His research interests include object detection, applied machine learning, social media analysis, and intelligent vision systems.

Dr. Xia received the Outstanding Reviewer Award for *Neurocomputing* in 2016. He was a recipient of the Science Research Famous Achievement Award from the Higher Institution of China in 2015. He has served as a Reviewer for many journals, including IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and *Neurocomputing*. He has also served on the PC/SPC for the conferences, including AAAI, ACM MM, MICCAI, and IJCAI. He is a member of ACM.