

# Learning Where, What and How to Transfer: A Multi-Role Reinforcement Learning Approach for Evolutionary Multitasking

Jiajun Zhan, Zeyuan Ma, Yue-Jiao Gong, *Senior Member, IEEE* and Kay Chen TAN, *Fellow, IEEE*

**Abstract**—Evolutionary multitasking (EMT) algorithms typically require tailored designs for knowledge transfer, in order to assure convergence and optimality in multitask optimization. In this paper, we explore designing a systematic and generalizable knowledge transfer policy through Reinforcement Learning. We first identify three major challenges: determining the task to transfer (where), the knowledge to be transferred (what) and the mechanism for the transfer (how). To address these challenges, we formulate a multi-role RL system where three (groups of) policy networks act as specialized agents: a task routing agent incorporates an attention-based similarity recognition module to determine source-target transfer pairs via attention scores; a knowledge control agent determines the proportion of elite solutions to transfer; and a group of strategy adaptation agents control transfer strength by dynamically controlling hyperparameters in the underlying EMT framework. Through pre-training all network modules end-to-end over an augmented multitask problem distribution, a generalizable meta-policy is obtained. Comprehensive validation experiments show state-of-the-art performance of our method against representative baselines. Further in-depth analysis not only reveals the rationale behind our proposal but also provide insightful interpretations on what the system have learned.

**Index Terms**—Evolutionary multitasking, Parameter control, Deep reinforcement learning, Knowledge transfer, Meta-Black-Box Optimization

## I. INTRODUCTION

E VOLUTIONARY Algorithms (EAs) are meta-heuristics that resemble Darwinian principle [1], where solution population goes through reproduction and natural selection to search for optima in given optimization problems. Traditional EAs, such as Genetic Algorithm (GA) [2], Differential Evolution (DE) [3] and their modern variants [4], [5], have long-standing reputation for addressing optimization problems in both industrial and scientific discovering fields [6]–[8]. Recently, a novel scenario termed as Multitask Optimization (MTO) was introduced by Gupta et al [9] to address the urgent need for efficient algorithms that can solve concurrent tasks in cloud computing industry. The complex nature of MTO poses new challenges to traditional EAs: how to realize effective knowledge transfer mechanism to accelerate the optimization process when handling multiple tasks simultaneously [10], [11]?

To address the above challenges, Evolutionary Multitasking (EMT) has emerged as a prominent solution, with its approaches generally falling into two branches according to their underlying paradigms. The first category is EMT algorithms with implicit knowledge transfer mechanism, first proposed by Gupta et al. [9] along with the proposal of MTO definition.

The proposed algorithm, termed as Multifactorial Evolution Algorithm (MFEA), maintains a single solution population for the sub-tasks in MTO, where each individual is indexed by its most specialized task. The knowledge transfer occurs within the reproduction (mutation & crossover) and selection of the evolution process. While such implicit parallelism facilitates efficient knowledge sharing and concurrent multitasking, it, to some extent, sacrifices the algorithmic flexibility in terms of customization for each sub-task [12]. This motivated the second category of researches that explore EMT algorithms with explicit knowledge transfer mechanisms. The paradigm in this line deploys a group of customized evolution processes for the sub-tasks in MTO. The knowledge transfer is governed by an explicit measure of inter-task similarity and an associated information exchange operator across different evolution processes. For example, in an early-stage study of Feng et al. [13], denoising autoencoders are trained for inter-task solution mapping to dynamically transfer solution-level knowledge. In this paper, we focus on the second paradigm of developing explicit EMT algorithms.

The development of explicit EMT algorithms centers on three fundamental questions regarding the knowledge transfer:

- **Where to Transfer:** Identifying which tasks should exchange information, typically by measuring inter-task similarity [14].
- **What to Transfer:** Determining the specific knowledge (e.g., proportion of solutions) to be conveyed from a source task to a target task [11].
- **How to Transfer:** Designing the precise mechanism for knowledge exchange between correlated tasks, such as the selection of evolutionary operators [15] or the control of transfer intensity [16].

Following these research questions, a wide array of explicit EMT algorithms have been proposed [17], [18].

Though promising, the performance of existing EMT algorithms might be cursed by *no-free-lunch* theorem [19]. This theorem implies that an algorithm with fixed, manually-designed components, while excelling in some optimization environments, will inevitably struggle when faced with novel problems or unexpected shifts in the problem landscape. Consequently, adapting or redesigning such algorithms demands significant expertise and development effort, a challenge frequently highlighted in recent surveys on Meta-Black-Box Optimization (MetaBBO) [20]–[23]. The MetaBBO paradigm proposes automating the algorithm design workflow through meta learning [24], [25]. This is typically achieved by a bi-level architecture, where a meta-level policy is trained to configure or generate a low-level algorithm, in order to

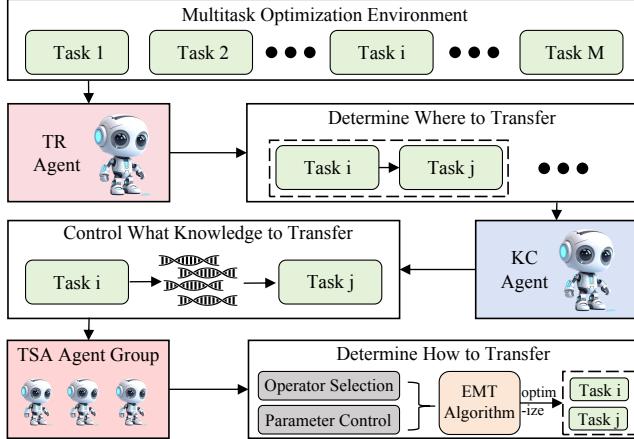


Fig. 1. Conceptual overview of **MetaMTO** framework.

boost the overall performance of the low-level optimization over a problem distribution, thereby generalizable to unseen problems. While the MetaBBO paradigm has been widely instantiated in several key avenues of EAs community such as single-objective optimization [26]–[28], multi-objective optimization [29], [30], and multi-modal optimization [31], [32], to the best of our knowledge, only two works [33], [34] explore the possibility of learning-boost EMT algorithms. While these initial works use reinforcement learning to address knowledge transfer, they fall short of providing holistic control, automating only isolated aspects of the “where, what and how” decisions. The lack of comprehensive and coordinated strategy leads to two critical issues: it not only results in suboptimal designs and performance bottlenecks, but it also fails to fully resolve the reliance on human expertise that MetaBBO aims to address.

To fill the research gap mentioned above, in this paper, we propose **MetaMTO**, a novel MetaBBO framework designed to learn systematic and generalizable EMT algorithms. Our approach is centered directly on the three fundamental questions of EMT: determining the source task (**where**), the knowledge to be transferred (**what**), and the method to transfer (**how**). To this end, in **MetaMTO**, we propose an end-to-end reinforcement learning (RL) [35] system to learn a cohesive policy that concurrently addresses all three interconnected questions. We illustrate our work’s novelty in Fig. 1, where RL agents with various roles are holistically organized together to facilitate fully automated control for knowledge transfer. Specifically, three types of agents are deployed to govern knowledge transfer in a cooperative fashion:

- **Task Routing (TR) Agent:** This agent addresses the “where to transfer” question. It processes status features extracted from all sub-tasks and uses an attention-based architecture to compute pairwise similarity scores. Based on the attention scores, it routes each target task with the most-matching source task for knowledge transfer.
- **Knowledge Control (KC) Agent:** This agent is responsible for the “what to transfer” decision. For each recognized source-target pair identified by the TR agent, the KC agent determines the quantity of knowledge

to transfer by selecting a specific proportion of elite solutions from the source task’s population.

- **Transfer Strategy Adaption (TSA) Agent Group:** These agents tackle the final “how to transfer” question. For each source-target pair and corresponding knowledge to be transferred, a TSA agent further suggest the desired transfer strategy, through controlling the key algorithm configurations in the underlying EMT framework.

At its core, **MetaMTO** frames the EMT algorithm control as a Markov Decision Process (MDP) and employs a multi-role RL system to solve it, which enables a dynamic, per-step policy to control the entire knowledge transfer process for each MTO problem. To facilitate the training and evaluation, we further make following contributions to complement the overall systematic pipeline, including i) *Training Objective*: a novel reward scheme is designed to objectively measure how good **MetaMTO** is at designing per-step knowledge transfer strategy, kicking a balance between global convergence performance and transfer success rate. Then the training objective is maximizing the accumulated rewards along an optimization process; ii) *Training Diversity*: a novel MTO problem set is constructed by augmenting existing ones [36], which allows diversified problem distribution through efficient hierarchical composition. Meta-training **MetaMTO** over such training data leads to competitive generalization.

We have conducted rigorous comparative validation, where the results demonstrate state-of-the-art performance of our work against both human-crafted and learning-assisted baselines. Plentiful ablation studies and interpretation analysis further enhance our work’s scientific integrity.

The remaining of this paper is organized as follows. In Section II, we systematically review existing human-crafted EMT algorithms and learning-assisted ones, highlighting the research gap in existing research progress. Section III present a step-by-step derivation of **MetaMTO**, including both the mathematical modeling and specific technical designs. Thereafter, in Section IV, rigorous validation studies are conducted to empirically support our work’s significance, which include not only absolute performance comparison but also in-depth ablation and interpretability analysis. To conclude, the main contributions and findings of the paper are summarized in Section V, along with several promising directions we outline as important future works.

## II. RELATED WORK

### A. Evolutionary Multitasking

Evolutionary multitask optimization (EMT) is specifically designed to solve multitask optimization (MTO) problems [9]. It aims to solve multiple sub-tasks synergistically within a single optimization process. The objective for a given MTO problem instance  $\mathcal{I} := \{T_1, T_2, \dots, T_K\}$  can be mathematically defined as finding the set of optimal solutions  $\{x_1^*, x_2^*, \dots, x_K^*\}$ , where each solution is given by:

$$x_j^* = \operatorname{argmin}_{x \in R_j} f_j(x), \quad j = 1, 2, \dots, K, \quad (1)$$

where  $f_j$  and  $R_j$  denote the objective function and feasible region of sub-task  $T_j$ . MTO presents significant challenges to traditional EAs due to i) heterogeneous landscape properties of the objective functions  $\{f_j\}_{j=1}^K$  across sub-tasks; and ii) misaligned feasible decision variable regions  $\{R_j\}_{j=1}^K$ . Obviously, the similarity and dissimilarity between sub-tasks play a key role in MTO and should be addressed carefully. Motivated accordingly, existing EMT approaches mainly focus on designing effective *knowledge transfer* scheme. The fundamental rationale is that by distinguishing similar and dissimilar sub-tasks, we could assign computational resources properly to attain the optimality of MTO more efficiently. Following this idea, a wide range of EMT approaches have been proposed recently and they can be categorized into two branches according to their specific knowledge transfer paradigms [37].

1) *Implicit Knowledge Transfer*: In the implicit knowledge transfer scheme, a unified population is typically maintained to address all sub-tasks. Knowledge transfer is then implicitly conducted within the reproduction and selection of the evolution process. In the realm of implicit EMT algorithms, the pioneering work of Multifactorial Evolutionary Algorithm (MFEA) was first introduced by Gupta et al. [9]. In MFEA, each individual is assigned a skill factor to indicate its adept task. The occurrence of implicit knowledge transfer is controlled by a fixed random mating probability (*rmp*) between individuals with distinct skill factors. Recognizing that the fixed *rmp* in MFEA has limited capability in dynamically adjusting knowledge transfer probability and may lead to negative knowledge transfer, Bali et al. [38] introduced a data-driven evolutionary multitasking algorithm (MFEA-II). MFEA-II employs an adaptive *rmp* matrix to represent pairwise transfer probabilities between tasks, which can be updated online using real-time data. Furthermore, to more effectively control the occurrence of knowledge transfer, Zheng et al. [39] designed an ability vector for each individual to indicate its probability of being selected as transferred knowledge. Tang et al. [40] proposed a grouping strategy that clusters similar tasks together, permitting inter-task knowledge transfer only within the same group.

The aforementioned works consider “where” and “what” to transfer in implicit EMT, whereas some other efforts pay attention to “how” to transfer the knowledge. Bali et al. [41] proposed a linear domain adaptation strategy to transform the search space of a simple source task to the complex target task. Furthermore, Zhou et al. [42] proposed an adaptive knowledge transfer approach, which assigns an operator indicator for each individual to adaptively select the most suitable crossover operators. Li et al. [43] incorporated a team learning strategy into the MFEA. They divides the population into one elite and three ordinary teams, with each team conducting knowledge transfer by a distinct approach.

2) *Explicit Knowledge Transfer*: While implicit EMT algorithms have shown promising results in diverse optimization domains [44], [45], they still face some limitations, such as the insufficient flexibility for per-task adaptation and the limited control over dynamic resource allocation. The explicit knowledge transfer was then proposed by Feng et al. [13]

and has since become an active area of research. Specifically, the method is characterized by its multi-population structure, where each sub-task is optimized by a dedicated population. This allows more flexible adaptation within the separated sub-task domain, while also enabling explicit transfer operations between them through well-defined operations like solution exchange and transformation. To explore the where to transfer question, population distribution-based similarity measurement approach is often used. Chen et al. [14] proposed to employ an archive for each task to record previous and current population and select the auxiliary task by measuring the Kullback-Leibler divergence [46] for task similarity between paired task archives. Along the line, Wang et al. [47] introduced an anomaly detection model fitted by the population of each task to detect candidate transferred individuals. Gong et al. [11] proposed MTO-DRA, which incorporates dynamic resource allocation into explicit knowledge transfer. This approach dynamically assigns more computational resources to more challenging tasks, thereby regulating the amount of transferred knowledge.

Finally, the question of how to transfer has been approached. Lin et al. [15] focused primarily on the selection of information exchange operators. They applied an indicator vector to dynamically choose domain adaptation approaches for learned mappings between different task domains. In parallel, Wu et al. [16] studied the transfer strength control and introduced an approach termed TRADE. This approach transfers valid DE parameter settings from auxiliary to target tasks as explicit knowledge, thereby enhancing the search efficiency. From another perspective, Jiang et al. [48] proposed a block-based mechanism and enabled explicit knowledge transfer to occur in non-aligned dimensions of individuals of the same or different tasks. In addition, some recent researches develop integrated approaches. Xu et al. [17] proposed an adaptive EMT framework that coordinates knowledge transfer frequency, auxiliary task selection and transfer intensity in a synergistic manner. In a similar vein, Liang et al. [18] introduced a multi-source knowledge transfer mechanism, which integrates novel designs for adjusting transfer probability, identifying auxiliary tasks, and facilitating the knowledge transfer process.

The EMT approaches reviewed above have shown promising results in real-world MTO scenarios [49], e.g., the vehicle routing problem [50], [51], the itinerary planning [52] and the fuzzy system design [53].

### B. Learning-assisted Evolutionary Multitasking

Though promising, a key limitation of human-crafted EMT approaches is that: being designed and tuned for a specific group of problems, they unavoidably require labor-intensive fine-tuning or even re-design to adapt for new problems. This issue has been well summarized as the “*no-free-lunch*” curse in recent learning-assisted optimization literature, where MetaBBO are suggested as effective tools to approach generalized optimization [20], [21], [26], [54]–[59]. Motivated by this perspective, Chen et al. [60] and Xue et al. [61] both employed neural networks to model transfer mappings between task pairs. While Chen et al. used zero padding to align dimensions

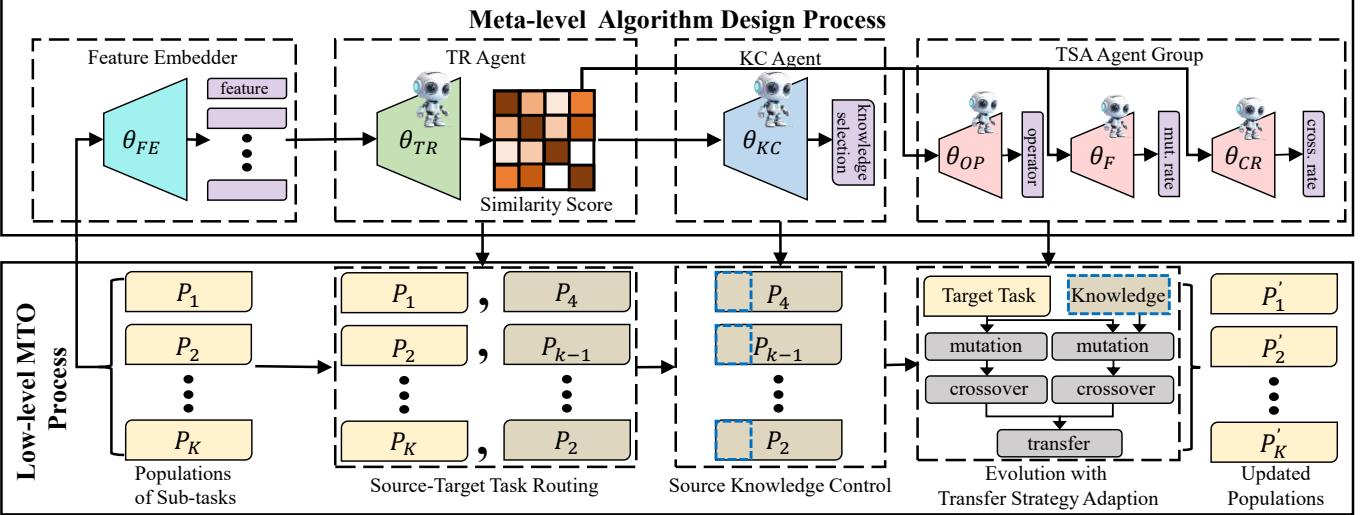


Fig. 2. A detailed illustration of the overall workflow within **MetaMTO**. Following the bi-level architecture in MetaBBO, a multi-role RL system is deployed at the meta-level to control the knowledge transfer scheme at the low-level evolution. A single evolution step is presented for simplicity.

across tasks, Xue et al. directly mapped solutions even between tasks with different dimensions. On the other hand, to mitigate the computational burden caused by additional function evaluations in multitask optimization, surrogate-assisted EMT algorithms have been developed. Huang et al. [62] proposed a novel surrogate-assisted task selection approach. In this approach, a surrogate model is trained to determine the skill factor of each offspring and identify the most promising candidates for real and precise evaluation.

The above studies represent significant progress at the component level, by designing sophisticated transfer mechanisms and efficient heuristics. However, a higher-level approach is required to shift the focus from designing the transfer mechanisms to learning the meta-level policy that governs them. This aligns with the essential goal of the MetaBBO paradigm, which, to our knowledge, has received little attention so far. Li et al. [33] introduced an adaptive strategy using the Q-learning method to dynamically optimize the *rmp* value for each task. Wu et al. [34] developed a learning-to-transfer (L2T) framework that autonomously regulates the knowledge transfer process and effectively addresses the issues of when and how to transfer across tasks.

Despite their promise, existing learning-assisted EMT works are still limited at the early exploratory stage. Taking L2T [34] as an example, on the one hand, its rigid architecture is tied to a fixed number of tasks, which hinders generalization. On the other hand, the provided control is in-holistic, leaving the “where” to transfer mechanism unexplored. This motivates us to propose **MetaMTO** for more systematic and generalized learning to holistically address “where”, “what” and “how” questions in an end-to-end manner.

### III. METHODOLOGY

As illustrated in Fig. 2, **MetaMTO** aligns with the bi-level MetaBBO paradigm [20], while equipped with novel designs tailored for EMT backbone to solve MTO problems. We begin with modeling the interplay between the meta-level policy and

the low-level EMT process and defining the corresponding learning objective. Without loss of generality, we assume that all sub-tasks in an MTO problem hold minimization objectives, and the underlying EMT approach could be any off-the-shelf multi-population EAs, which in this paper, is a primitive multi-population DE architecture that resembles initial EMT researches [12], [14] with explicit knowledge transfer schemes.

#### A. Problem Modeling

Suppose we have a MTO problem distribution  $\mathbb{D}$  from which diverse MTO problem instances can be sampled. Each MTO instance  $\mathcal{I}$  sampled from  $\mathbb{D}$  comprises  $K$  sub-tasks  $T_1 \sim T_K$ . Our aim is to learn an algorithm design policy  $\pi_\theta$ , where  $\theta$  are some learnable parameters, to control the evolutionary optimization process of a given EMT approach  $\mathcal{A}$ , so that it achieves desired optimization performance on each  $\mathcal{I} \in \mathbb{D}$ . We now clarify the specific mathematics behind via diving into the low-level evolutionary process. During the optimization of each  $\mathcal{I}$  by the EMT approach  $\mathcal{A}$ , we first define  $\mathcal{R}_{\mathcal{I}}^t$  as a scalar performance metric capable of profiling the optimization progress information as of  $t$ -th optimization step across all  $K$  sub-tasks in  $\mathcal{I}$  (larger is better). In existing MetaBBO approaches tailored for single-task optimization, the design of  $\mathcal{R}_{\mathcal{I}}^t$  majorly focuses on objective’s optimality. However, in this work, since we aim to address MTO problems, the design has to consider optimality in each sub-task’s objective space and also the knowledge transfer status, which we detail in Sec III-B4. Following this design principle, at each time step  $t$ , a state function  $sf(\cdot)$  maps the meta information (e.g., populations and objective values for sub-tasks, knowledge transfer history, etc.) into a state representation  $s^t$ . The policy  $\pi_\theta$  then conditions on state  $s^t$  to output a knowledge transfer action  $a^t$  for  $\mathcal{A}$ . By using  $a^t$  to optimize  $\mathcal{I}$ , we advance the evolutionary process by one step and record the performance metric  $\mathcal{R}_{\mathcal{I}}^t$ . The learning objective of **MetaMTO** is to find a

TABLE I  
STATE DEFINITION IN **METAMTO** FOR EACH SUB-TASK POPULATION.

State ID	Definition	Type	Description
$s_1$	$\text{mean}(\text{std}(X))$	[0, 1]	diversity in decision space
$s_2$	$\text{std}(f(X))$	[0, 1]	convergence in objective space
$s_3$	$\frac{G_{\text{sta}}}{G}$	[0, 1]	stagnation status in optimization
$s_4$	$I(f(X^{t+1}) < f(X^t))$	bool	new best-so-far solution checking
$s_5$	$\frac{N_{\text{sur}}}{N_{\text{tra}}}$	[0, 1]	survival rate of transferred solutions

TABLE II  
ACTION GROUPS IN **METAMTO** FOR EFFECTIVE KNOWLEDGE TRANSFER.

Group	Action ID	Range	Description
TASK ROUTING	$\{a_{(1),j}\}_{j=1}^K$	$\{1, 2, \dots, K\}^K$	assign source task for the recipient task
KNOWLEDGE SELECTION	$\{a_{(2),j}\}_{j=1}^K$	$[0, 0.5]^K$	determine the elite portion to transfer
TRANSFER STRATEGY ADAPTION	$\{a_{(3.1),j}\}_{j=1}^K$	$\{1, 2, 3, 4\}^K$	dynamic mutation operator selection
	$\{a_{(3.2),j}\}_{j=1}^K$	$[0, 1]^K$	control mutation strength
	$\{a_{(3.3),j}\}_{j=1}^K$	$[0, 1]^K$	control crossover strength

policy that maximizes the expectation of accumulated  $\mathcal{R}_{\mathcal{I}}^t$  over the MTO problem distribution  $\mathbb{D}$ :

$$\mathcal{J}(\theta) = \mathbb{E}_{\mathcal{I} \in \mathbb{D}} \left[ \sum_{t=1}^G \mathcal{R}_{\mathcal{I}}^t \right] \quad (2)$$

where  $G$  denotes the number of allowed maximal optimization steps for each problem instance. With this problem modeling, we next detail designs in **MetaMTO** for each mathematical component above.

### B. MDP Formulation

We formally model the above meta-learning problem as a Markov Decision Process (MDP) [63], where the formalized  $s^t$  denotes state, the configuration decision  $a^t$  denotes action, and the immediate performance metric  $\mathcal{R}_{\mathcal{I}}^t$  denotes the reward. We detail their concrete designs as follows.

1) *State*: At each optimization step  $t$  of the underlying EMT process, given the populations  $\{(X_j^t, f_j(X_j^t))\}_{j=1}^K$  for each of the  $K$  sub-tasks, we construct for each sub-task a 5-dimensional feature vector  $s_j^t$  to describe the optimization state information. We present the mathematical definition of these feature logits in Table I, where we omit subscripts  $j$  and superscript  $t$  wherever appropriate. Note that such state feature design not only coheres with the up-to-date MetaBBO researches [34], [55], [56], [58], [64] but also further incorporates the optimization information of knowledge transfer. Specifically, according to recent landscape feature analysis literature [54], [65]–[67], we first distill a concise set of features to avoid redundancy. This results in four representative features ( $s_1$  to  $s_4$ ) that cover the diversity, convergence, stagnation and update process. Then, inspired by L2T [34], we purposely add a key state feature ( $s_5$ ) that quantifies the positive impact of the knowledge transferred to the recipient sub-task's evolution. The incorporation of  $s_5$  enables the agent to characterize both task-specific and cross-task optimization dynamics, thereby facilitating comprehensive and coordinated action-making for knowledge transfer. The complete state feature is hence the union of state features  $s^t := \{s_j^t\}_{j=1}^K$  from all  $K$  sub-tasks. We leave the computation details of these features at Appendix I.A.

TABLE III  
FOUR DIFFERENT OPERATORS IN THE MUTATION OPERATOR POOL.

Operator ID	Formulation
1	$\mathcal{V} = X_{\text{target,best}} + F \cdot (X_{\text{source},r_1} - X_{\text{source},r_2})$
2	$\mathcal{V} = X_{\text{target},r_1} + F \cdot (X_{\text{source},r_2} - X_{\text{source},r_3})$
3	$\mathcal{V} = X_{\text{source},r_1} + F \cdot (X_{\text{target},r_2} - X_{\text{target},r_3})$
4	$\mathcal{V} = X_{\text{source,best}} + F \cdot (X_{\text{target},r_1} - X_{\text{target},r_2})$

2) *Action*: To provide a comprehensive and sophisticated control, an agent must answer three fundamental questions of where, what, and how to transfer. Listed in Table II, we structure our action space by defining three distinct action groups, each responsible for one of these decisions. The action space is formulated as  $\mathbf{A} := \{a_{(1)}, a_{(2)}, a_{(3.1)}, a_{(3.2)}, a_{(3.3)}\}$ , where action groups  $a_{(1)}$ ,  $a_{(2)}$  and  $a_{(3)} := \{a_{(3.1)}, a_{(3.2)}, a_{(3.3)}\}$  respectively correspond to task routing, knowledge control and transfer strategy adaption:

- **Task Routing (TR)**: In this group, each of the  $K$  sub-tasks is regarded as a target task and to be assigned with a source task. Then the overall action is  $a_{(1)} := \{a_{(1),j}\}_{j=1}^K$ , with each  $a_{(1),j}$  is an integer index selected from  $\{1, 2, \dots, K\} \setminus \{j\}$  (for  $j$ -th sub-task, index  $j$  is not allowed to be selected). This action group is responsible for determining where to transfer knowledge.
- **Knowledge Control (KC)**: Once the source task is assigned, we then determine what knowledge to transfer. This is achieved by determining a proportion  $a_{(2),j} \in [0, 0.5]$ , and then selecting that proportion of elite solutions from source task's population. The overall action is hence  $a_{(2)} := \{a_{(2),j}\}_{j=1}^K$ . Note that the upperbound is set to 0.5 to prevent excessive knowledge transfer that may pollute the original evolution of the target task.
- **Transfer Strategy Adaption (TSA)**: We further propose three action sub-groups  $a_{(3.1)}$ ,  $a_{(3.2)}$  and  $a_{(3.3)}$  to provide fine-grained algorithmic control over the low-level evolution algorithm (which is a multi-population DE architecture in this study). Specifically, we control the mutation operator selection, mutation strength  $F$ , and the crossover probability  $Cr$ , respectively, to provide per-step control for the DE. For each  $a_{(3.1),j}$ , it is selected from a predefined operator pool comprising four behavior-varied mutation strategies. As listed in Table III, these four operators provide different extent of exploration-exploitation tradeoff between the target and source tasks, which are dynamically selected within the EMT process to facilitate adaptive optimization. The other two sub-groups control the parameters of the mutation and crossover operators<sup>1</sup>. Specifically, the action  $a_{(3.2),j}$  determines the mutation strength  $F \in [0, 1]$ , and the action  $a_{(3.3),j}$  sets the crossover probability  $Cr \in [0, 1]$ , respectively, for the  $j$ -th sub-task.

In summary, there are three action groups in **MetaMTO**, which cooperate to achieve systematic and fine-grained per-step control for the low-level EMT framework.

<sup>1</sup>In this paper, we do not control the selection of crossover operator, using binomial crossover by default.

**3) State Transition Dynamics:** Given the state and action definitions, the state transition dynamics in our MDP model correspond to one iterative evolution step of the underlying EMT framework (as illustrated in the bottom right of Fig. 2). Specifically, once the meta-level policy provides the action  $a^t$  conditioning on the current state  $s^t$ , the underlying EMT framework performs a mutation-crossover routine to generate an offspring population of  $\{V_j^t\}_{j=1}^K$ . First, a proportion of the offspring, corresponding to  $(1 - a_{(2),j}^t)$  of the population size, is generated through self-evolution in the  $K$  sub-tasks by utilizing the local information within population. Then, the remaining  $a_{(2),j}^t$  proportion of population is generated by cross-task knowledge transfer. Guided by the other components of the action  $a^t$ , the pathway involves: identify the source task by  $a_{(1),j}^t$ , and adaptively transfer knowledge between the selected elites and local population with behaviors controlled by  $a_{(3,1)}^t$ ,  $a_{(3,2)}^t$  and  $a_{(3,3)}^t$ . Finally, to produce the next generation, each sub-task's parent population and offsprings undergo a vanilla DE selection strategy, where the fitter individual from each parent-offspring pair survives to the next generation. The resulting population provides the next state information,  $s^{t+1}$ , in our MDP.

**4) Reward:** Reward design plays an important role in RL systems, which impacts the overall training robustness and effectiveness [68]. It is also reported in MetaBBO literature that designing robust reward signal for learning-assisted optimization is more challenging than typical RL tasks [20]. To strike a balance between each sub-task's self optimization and the exploitation of knowledge transfer, we design a reward function by crediting two terms. This design not only promotes maximal performance gain in each sub-task's evolution, but also encourages positive knowledge transfer. The reward function is defined as:

$$\begin{aligned} \mathcal{R}^t &= \sum_{j=1}^K (\mathcal{R}_{c,j}^t + \mathcal{R}_{k,j}^t), \\ \mathcal{R}_{c,j}^t &= \frac{f_j^t - f_j^{t+1}}{f_j^0 - f_j^*}, \quad \mathcal{R}_{k,j}^t = \frac{N_{\text{success},j}^t}{N_{\text{transfer},j}^t}. \end{aligned} \quad (3)$$

where  $f_j^*$  denotes the optimal of  $j$ -th sub-task,  $f_j^t$  denotes the optimal found until the  $t$ -th optimization step,  $N_{\text{transfer},j}^t$  denotes the number of all transferred solutions from source task and  $N_{\text{success},j}^t$  denotes the survived number. It should be noted that for black-box MTO, the true  $f_j^*$  is agnostic. In such case,  $f_j^*$  can be replaced by the proxy optimal, which is the best solution found using existing EMT approaches through multiple runs. This proxy value is exclusively a training-phase construct; it is not required during the inference phase, when the trained controller is applied to solve problems.

### C. Multi-Role RL System

Previously, we have elaborated the MDP details in our **MetaMTO**, including the state, action and reward designs. To solve this MDP, we introduce the meta-level policy: a multi-role RL system comprising multiple RL agents. As illustrated in the top of Fig. 2, these agents are instantiated as neural

networks. In this subsection, we elaborate the data flow in the system, explaining how these agents operate collectively.

**1) Feature Embedder:** Given the abstracted state feature  $s^t := \{s_j^t\}_{j=1}^K$ , where  $s_j^t \in \mathbb{R}^5$  is the state feature for  $j$ -th sub-task at optimization step  $t$ . We first use a feature embedder  $\mathcal{W}(\cdot | \theta_{\text{FE}})$  to embed them into a hidden feature space through the linear mapping  $\theta_{\text{FE}} : \mathbb{R}^5 \rightarrow \mathbb{R}^{64}$ . Then we obtain embeddings  $e^t = \mathcal{W}(s^t | \theta_{\text{FE}})$  for all sub-tasks, where  $e^t \in \mathbb{R}^{K \times 64}$ . The embeddings are used by multi-role RL agents to determine the corresponding action groups.

**2) TR Agent:** To determine where to transfer the knowledge, the feature embeddings are first fed into the TR agent, which is an attention block [69], represented by  $\mathcal{W}(\cdot | \theta_{\text{TR}})$ . The block contains a single attention head with 64-dimentional hidden state, followed by a batch normalization [70]. Given  $e^t$ , the TR agent outputs two logits:

$$h_{\text{score}}^t, h_{\text{decision}}^t = \mathcal{W}(e^t | \theta_{\text{TR}}) \quad (4)$$

where  $h_{\text{score}}^t \in \mathbb{R}^{K \times K}$  denotes the attention score matrix to be used immediately in current task routing stage, and  $h_{\text{decision}}^t \in \mathbb{R}^{K \times 64}$  is a feature matrix to be passed to subsequent agents.

First, the attention scores are used to pair each target task  $j$  with a source task  $a_{(1),j}^t$  by selecting the task with the highest score (excluding the task itself):

$$a_{(1),j}^t = \underset{k \in \{1, 2, \dots, K\}}{\operatorname{argmax}} h_{\text{score}}^t[j, k], \quad k \neq j. \quad (5)$$

The resulting target-source pairs  $\{j, a_{(1),j}^t\}_{j=1}^K$  are used in the knowledge transfer scheme of the low-level EMT framework.

Next, using these routing pairs, we construct an enriched input for subsequent RL agents. This is achieved by concatenating the decision embedding of each target task  $j$  with that of its assigned source task  $a_{(1),j}^t$ :

$$h_{\text{concat},j}^t = \text{Concat}(h_{\text{decision}}^t[j], h_{\text{decision}}^t[a_{(1),j}^t]) \quad (6)$$

This concatenation yields the final input  $h_{\text{concat}}^t \in \mathbb{R}^{K \times 128}$  to the next stage.

**3) KC Agent:** The KC agent determines what knowledge (how much percentage of elite solutions) should be transferred from the source task  $a_{(1),j}^t$  to the target  $j$ . Its architecture is a two-layer MLP with hidden layer's dimension as 64 (activated by ReLU) and the output layer's dimension as 1 (activated by Tanh). The agent is represented as  $\mathcal{W}(\cdot | \theta_{\text{KC}})$ . The primary design goal is to generate a transfer ratio,  $a_{(2),j}^t$ , bounded within the range  $[0, 0.5]$  to prevent “knowledge pollution”. Specifically, fed with  $h_{\text{concat},j}^t$ , the KC agent outputs a mean value:

$$\mu_{\text{kc},j}^t = 0.25 + 0.25 \times \mathcal{W}(h_{\text{concat},j}^t | \theta_{\text{KC}}) \quad (7)$$

Since the MLP's final Tanh activation ensures  $\mathcal{W}(\cdot | \theta_{\text{KC}}) \in [-1, 1]$ , the  $\mu_{\text{kc},j}^t$  is inherently restricted to the desired  $[0, 0.5]$ . Then the concrete percentage of transferred knowledge is sampled from a Gaussian distribution  $a_{(2),j}^t \sim \mathcal{N}(\mu_{\text{kc},j}^t, 0.1)$  centered at this bounded mean, where we fix the variance term as 0.1 for simplicity. To strictly enforce the boundary constraint, any sampled value falling outside the  $[0, 0.5]$  interval is truncated to fit within this range.

4) *TSA Agent Group*: To further determine how to transfer selected knowledge, the first agent in the TSA group is responsible for selecting a proper mutation operator from the operator pool in Table III for the current step  $t$ . Its architecture is a two-layer MLP parameterized by  $\theta_{\text{OP}}$ , with hidden layer's dimension as 64 (activated by ReLU) and the output layer's dimension as 4 (activated by ReLU). The operator choice  $a_{(3.1),j}^t$  is then sampled by:

$$a_{(3.1),j}^t \sim \text{Softmax}(\mathcal{W}(h_{\text{concat},j}^t | \theta_{\text{OP}})) \quad (8)$$

The second and third agents  $\theta_F$  and  $\theta_{Cr}$  control the mutation strength  $F_j^t$  and  $Cr_j^t$  respectively. Their architectures resemble the KC agent, similarly, they also output 1-dimensional mean terms  $\mu_{F,j}^t$  and  $\mu_{Cr,j}^t$  (activated by Tanh):

$$\mu_{F,j}^t, \mu_{Cr,j}^t = 0.5 + 0.5 \times \mathcal{W}(h_{\text{concat},j}^t | \theta_F, \theta_{Cr}) \quad (9)$$

The concrete values are then sampled as  $a_{(3.2),j}^t \sim \mathcal{N}(\mu_{F,j}^t, 0.1)$  and  $a_{(3.3),j}^t \sim \mathcal{N}(\mu_{Cr,j}^t, 0.1)$ . Likewise, for  $a_{(3.2),j}^t$  and  $a_{(3.3),j}^t$ , we truncate the sampled values from the Gaussian distribution to the bounded range  $[0, 1]$ .

To summarize, the learnable parameters of **MetaMTO** are the union of the feature embedder and the five modules' parameters:  $\theta := \{\theta_{FE}, \theta_{TR}, \theta_{KC}, \theta_{OP}, \theta_F, \theta_{Cr}\}$ . At each optimization step  $t$  of the low-level EMT framework, systematic and fine-grained actions are conditionally sampled by the multi-role RL agents for all  $K$  sub-tasks' evolution processes:  $a^t = \mathcal{W}(s^t | \theta)$ . We denote the sample probability is  $p_\theta(a^t)$ .

#### D. Training Method

1) *Training Distribution*: Given the MDP and the multi-role RL system already defined, the last piece of our **MetaMTO** is a well-defined MTO distribution (problem set), which is crucial to ensure the generalization after the meta-learning [71]. Although existing works such as L2T [34] use representative MTO benchmarks such as CEC2017 [72] or WCCI2020 benchmarks [73], the benchmarks suffer from two limitations: first, the problem instances in these benchmarks are sparsely and non-uniformly distributed, and second, they contain an insufficient number of instances - often fewer than 10. The data may not be sufficient for a meta-learning agent to learn a truly robust and generalizable policy.

To address this issue, we develop a more comprehensive problem set, termed as augmented WCCI (AWCCI), based on the 7 basic functions in WCCI2020 [73]. The AWCCI set is constructed in an incremental, hierarchical and automatic fashion. Specifically, we let the instances in AWCCI as the random combinations of those 7 basic functions, which results in  $\sum_{i=1}^7 C_7^i = 127$  possible non-empty combinations. Then, for each combination, we generate 5 distinct MTO problem instances. Each instance consists of 10 parallel, 50-dimensional sub-tasks. The difference of the 5 instances is derived by the shift intensity level  $l \in \{0.05, 0.1, 0.2, 0.3, 0.4\}$ . Each sub-task in a multitask instance is generated through the following transformation on the basic function defined by its corresponding combination:

$$f' = f(\omega^T \cdot (x - s)), \quad s \sim l \times (\text{lb} + U[0, 1] \times (\text{ub} - \text{lb})) \quad (10)$$

---

#### Algorithm 1 **MetaMTO**'s Training Workflow.

---

**Input:** Initialized policy  $\pi_\theta$ , MTO trainset  $\mathbb{D}_{\text{train}}$ , EMT framework  $\mathcal{A}$ , Training epochs  $L$ , Optimization budget  $G$ .

```

1: set epoch  $l = 1$ ;
2: while  $l \leq L$  do
3:   for each instance  $\mathcal{I} \in \mathbb{D}_{\text{train}}$  do
4:     /* Initialize low-level EMT A */
5:     set optimization step  $t = 1$ ;
6:     initialize populations  $\{X_j^t, f_j^t(X_j^t)\}_{j=1}^K$  for  $\mathcal{I}$ ;
7:     /* Start optimization */
8:     while  $t \leq G$  do
9:       /* RL agent decides on action */
10:      Get state feature  $s^t$  (Sec. III-B1);
11:      Sample action  $a^t \sim \mathcal{W}(s^t | \theta)$  (Eqs. 4 ~ 9);
12:      /* Sub-task self-evolution */
13:      Generate  $(1 - a_{(2),j}^t)$  offspring by  $\mathcal{A}$ , creating  $\{V_{\text{se},j}^t\}$ ;
14:      /* Cross-task knowledge transfer */
15:      Generate  $a_{(2),j}^t$  offspring by  $\mathcal{A}$ , creating  $\{V_{\text{kt},j}^t\}$ ; {source task from  $a_{(1)}^t$ , behaviors from  $a_{(3)}^t$  (Sec. III-B2)}
16:      Combine offspring:  $\{V_j^t\}_{j=1}^K = \{V_{\text{se},j}^t\}_{j=1}^K \cup \{V_{\text{kt},j}^t\}_{j=1}^K$ ;
17:      /* Evaluation and selection */
18:      Evaluate and obtain  $\{V_j^t, f_j^t(V_j^t)\}_{j=1}^K$ ;
19:      Select next generation  $\{X_j^{t+1}\}_{j=1}^K$  from parents  $X_j^t$  and offspring  $V_j^t$  (Sec. III-B3);
20:      /* PPO training */
21:      Compute reward  $\mathcal{R}^t$  (Eq. 3);
22:      Record transition  $< s^t, a^t, \log \pi_\theta(a^t | s^t), \mathcal{R}^t, s^{t+1} >$ ;
23:      if  $t \% T_{\text{ppo}} = 0$  then
24:        Compute policy gradients  $\nabla_\theta(\mathcal{J})$  of  $\mathcal{J}(\theta)$  in Eq. 2;
25:        Update  $\theta$  along  $\nabla_\theta$  using multi-step PPO [74];
26:      end if
27:      set  $t = t + 1$ ;
28:    end while
29:  end for
30:  set  $l = l + 1$ ;
31: end while
32: return trained  $\pi_\theta$ 
```

---

where  $\omega$  is a random rotation matrix, ub and lb denote the searching range of  $f$ . This process yields a total of  $127 \times 5 = 635$  problem instances, which are organized into five datasets according to their shift levels: AWCCI-VS (with very small shifting), AWCCI-S (with small shifting), AWCCI-M (with median shifting), AWCCI-L (with large shifting) and AWCCI-VL (with very large shifting). We provide a more detailed elaboration on AWCCI in Appendix I.B.

2) *Training Procedure*: We now elaborate the meta-training and inference workflows of our **MetaMTO**. The pseudocode of the training is presented in **Algorithm 1**. The input includes i) an initialized meta-level policy parameter  $\theta$ ; ii) a training problem distribution  $\mathbb{D}_{\text{train}}$ , which could be any subset of our AWCCI (i.e., VS ~ VL); iii) a low-level EMT framework, which in this paper we adopt a multi-population DE algorithm  $\mathcal{A}$  with explicit knowledge transfer scheme (similar with the algorithm structure in [12], [14]); iv) the meta-level learning budget  $L$  and the low-level optimization budget  $G$ . In each epoch  $g$ , we traverse each problem instance in  $\mathbb{D}_{\text{train}}$ , letting the meta-level policy  $\theta$  interact with low-level EMT optimization process (lines 4-16). The transitions are obtained through the interaction (lines 17-18) and recorded, which is then used to compute policy gradients (lines 20-21). In **MetaMTO**, we adopt the popular policy gradient method PPO [74]. The learn-

TABLE IV  
COMPARISON IN TERMS OF IN-DISTRIBUTION GENERALIZATION PERFORMANCE.

problems	Ours		L2T [34]		BLKT [48]		RLMFEA [33]		MFEA [9]	
	Avg obj	+/-	Avg obj	+/-	Avg obj	+/-	Avg obj	+/-	Avg obj	+/-
VS-1	<b>0.1897±0.0504</b>		0.2983±0.0019	<b>68/2</b>	0.2740±0.0108	<b>60/10</b>	0.3613±0.0136	<b>68/2</b>	0.3688±0.0114	<b>68/2</b>
VS-2	<b>0.1083±0.0119</b>		0.2495±0.0021	<b>183/27</b>	0.2736±0.0118	<b>182/28</b>	0.3605±0.0082	<b>203/7</b>	0.3552±0.0063	<b>204/6</b>
VS-3	<b>0.0866±0.0069</b>		0.2225±0.0025	<b>301/49</b>	0.2487±0.0052	<b>316/34</b>	0.3146±0.0048	<b>350/0</b>	0.3140±0.0055	<b>349/1</b>
VS-4	<b>0.0865±0.0031</b>		0.2139±0.0026	<b>255/95</b>	0.2465±0.0043	<b>329/21</b>	0.3094±0.0055	<b>342/8</b>	0.3085±0.0042	<b>342/8</b>
VS-5	<b>0.0929±0.0055</b>		0.1557±0.0027	<b>144/66</b>	0.2079±0.0032	<b>190/20</b>	0.2750±0.0082	<b>210/0</b>	0.2788±0.0056	<b>208/2</b>
VS-6	<b>0.0959±0.0143</b>		0.2035±0.0072	<b>60/10</b>	0.2396±0.0092	<b>66/4</b>	0.3347±0.0149	<b>70/0</b>	0.3325±0.0189	<b>70/0</b>
VS-7	<b>0.1360±0.0345</b>		0.3154±0.0031	<b>10/0</b>	0.3550±0.0095	<b>10/0</b>	0.3120±0.0209	<b>10/0</b>	0.3173±0.0277	<b>10/0</b>
S-1	<b>0.1820±0.0381</b>		0.3074±0.0013	<b>70/0</b>	0.2744±0.0079	<b>60/10</b>	0.3769±0.0067	<b>69/1</b>	0.3716±0.0067	<b>68/2</b>
S-2	<b>0.1317±0.0088</b>		0.2694±0.0026	<b>184/26</b>	0.2787±0.0088	<b>176/34</b>	0.3707±0.0076	<b>203/7</b>	0.3697±0.0066	<b>202/8</b>
S-3	<b>0.1165±0.0053</b>		0.2434±0.0027	<b>304/46</b>	0.2531±0.0037	<b>298/52</b>	0.3287±0.0075	<b>348/2</b>	0.3270±0.0048	<b>348/2</b>
S-4	<b>0.1180±0.0041</b>		0.2371±0.0018	<b>253/97</b>	0.2518±0.0042	<b>312/38</b>	0.3247±0.0060	<b>345/5</b>	0.3250±0.0062	<b>345/5</b>
S-5	<b>0.1285±0.0080</b>		0.1823±0.0036	<b>137/73</b>	0.2108±0.0039	<b>172/38</b>	0.2952±0.0055	<b>210/0</b>	0.2953±0.0075	<b>210/0</b>
S-6	<b>0.1266±0.0149</b>		0.2348±0.0065	<b>60/10</b>	0.2414±0.0045	<b>65/5</b>	0.3495±0.0140	<b>70/0</b>	0.3522±0.0191	<b>70/0</b>
S-7	<b>0.1596±0.0131</b>		0.3205±0.0025	<b>10/0</b>	0.3300±0.0430	<b>10/0</b>	0.3377±0.0256	<b>10/0</b>	0.3339±0.0194	<b>10/0</b>
M-1	<b>0.2605±0.0303</b>		0.3207±0.0015	<b>70/0</b>	0.2829±0.0091	<b>62/8</b>	0.3920±0.0101	<b>67/3</b>	0.3884±0.0049	<b>67/3</b>
M-2	<b>0.1838±0.0080</b>		0.3137±0.0016	<b>186/24</b>	0.2814±0.0093	<b>165/45</b>	0.4050±0.0058	<b>202/8</b>	0.3967±0.0073	<b>201/9</b>
M-3	<b>0.1713±0.0058</b>		0.2862±0.0018	<b>299/51</b>	0.2600±0.0049	<b>256/94</b>	0.3634±0.0041	<b>338/12</b>	0.3605±0.0051	<b>337/13</b>
M-4	<b>0.1749±0.0050</b>		0.2819±0.0019	<b>286/64</b>	0.2611±0.0038	<b>276/74</b>	0.3606±0.0045	<b>348/2</b>	0.3598±0.0051	<b>349/1</b>
M-5	<b>0.1659±0.0062</b>		0.2334±0.0024	<b>169/41</b>	0.2165±0.0037	<b>151/59</b>	0.3338±0.0089	<b>210/0</b>	0.3317±0.0063	<b>210/0</b>
M-6	<b>0.1957±0.0153</b>		0.2934±0.0043	<b>59/11</b>	0.2501±0.0045	<b>53/17</b>	0.3907±0.0125	<b>70/0</b>	0.3819±0.0151	<b>69/1</b>
M-7	<b>0.1813±0.0210</b>		0.3308±0.0042	<b>10/0</b>	0.3556±0.0053	<b>10/0</b>	0.3802±0.0325	<b>10/0</b>	0.3764±0.0322	<b>10/0</b>
L-1	<b>0.2687±0.0215</b>		0.3308±0.0017	<b>70/0</b>	0.2847±0.0131	<b>57/13</b>	0.4239±0.0055	<b>68/2</b>	0.4158±0.0045	<b>67/3</b>
L-2	<b>0.2195±0.0071</b>		0.3487±0.0021	<b>199/11</b>	0.2923±0.0064	<b>141/69</b>	0.4346±0.0033	<b>202/8</b>	0.4335±0.0056	<b>201/9</b>
L-3	<b>0.2121±0.0039</b>		0.3201±0.0021	<b>309/41</b>	0.2669±0.0032	<b>205/145</b>	0.4020±0.0040	<b>339/11</b>	0.3997±0.0029	<b>338/12</b>
L-4	<b>0.2167±0.0017</b>		0.3128±0.0024	<b>298/52</b>	0.2663±0.0026	<b>232/118</b>	0.4006±0.0031	<b>349/1</b>	0.3961±0.0036	<b>349/1</b>
L-5	<b>0.1993±0.0038</b>		0.2701±0.0022	<b>169/41</b>	0.224±0.0056	<b>112/98</b>	0.3652±0.0053	<b>210/0</b>	0.3695±0.0026	<b>210/0</b>
L-6	<b>0.2384±0.0066</b>		0.3302±0.0043	<b>68/2</b>	0.2540±0.0077	<b>46/24</b>	0.4162±0.0108	<b>70/0</b>	0.4236±0.0113	<b>69/1</b>
L-7	<b>0.2040±0.0168</b>		0.3460±0.0073	<b>10/0</b>	0.3628±0.0104	<b>10/0</b>	0.4091±0.0139	<b>10/0</b>	0.4168±0.0416	<b>10/0</b>
VL-1	<b>0.2736±0.0197</b>		0.3411±0.0018	<b>70/0</b>	0.3018±0.0079	<b>56/14</b>	0.4269±0.0050	<b>68/2</b>	0.4316±0.0096	<b>70/0</b>
VL-2	<b>0.2415±0.0065</b>		0.3656±0.0018	<b>201/9</b>	0.3255±0.0062	<b>163/47</b>	0.4498±0.0043	<b>198/12</b>	0.4479±0.0033	<b>198/12</b>
VL-3	<b>0.2413±0.0054</b>		0.3370±0.0007	<b>321/29</b>	0.2943±0.0036	<b>248/102</b>	0.4145±0.0026	<b>338/12</b>	0.4168±0.0030	<b>339/11</b>
VL-4	<b>0.2454±0.0044</b>		0.3335±0.0018	<b>323/27</b>	0.2919±0.0033	<b>267/83</b>	0.4174±0.0028	<b>350/0</b>	0.4168±0.0032	<b>349/1</b>
VL-5	<b>0.2268±0.0047</b>		0.2881±0.0011	<b>189/21</b>	0.2550±0.0043	<b>141/69</b>	0.3878±0.0045	<b>210/0</b>	0.3854±0.0030	<b>210/0</b>
VL-6	<b>0.2676±0.0103</b>		0.3483±0.0023	<b>69/1</b>	0.2956±0.0104	<b>54/16</b>	0.4361±0.0080	<b>69/1</b>	0.4339±0.0090	<b>70/0</b>
VL-7	<b>0.2349±0.0251</b>		0.3923±0.0060	<b>10/0</b>	0.3617±0.0083	<b>10/0</b>	0.4400±0.0184	<b>10/0</b>	0.4388±0.0343	<b>10/0</b>
total	<b>0.1823±0.0128</b>		0.2908±0.0028	<b>5424/926</b>	0.2763±0.0075	<b>4961/1389</b>	0.3743±0.0091	<b>6244/106</b>	0.3735±0.0104	<b>6237/113</b>

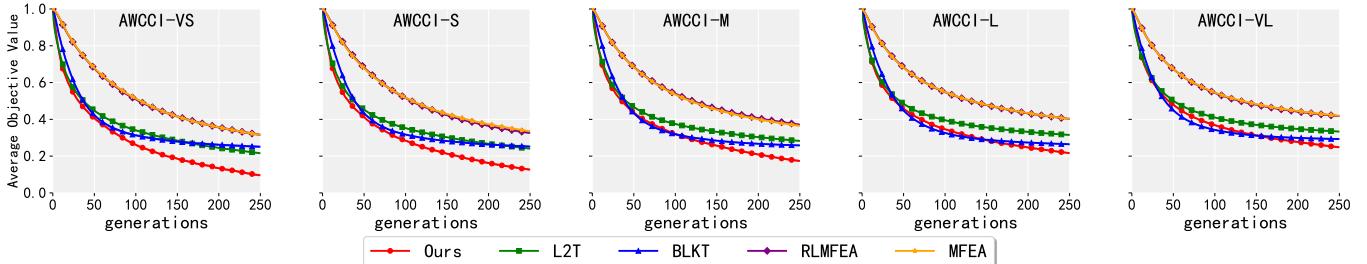


Fig. 3. Convergence curves on the five augmented test sets AWCCI-VS to AWCCI-VL.

ing occurs every  $T_{\text{ppo}}$  optimization steps, and the parameters is updated (line 22) for  $k_{\text{ppo}}$  iterations in each learning step. For the inference, once the training completes, one can directly use the trained model to solve unseen MTO instances, the only difference is that in inference, actions are selected in a deterministic manner (the one with maximal likelihood). We leave a more detailed settings at Sec. IV-A.

#### IV. RESULTS AND DISCUSSIONS

In this section, we conduct meticulous experimental validation to demonstrate the effectiveness and performance advantages of our approach. Specifically, we delve into following key research questions: **RQ #1:** have **MetaMTO** learned systematic and generalizable knowledge transfer policy for diverse MTO distributions? **RQ #2:** can the learned policy perform robustly under extreme distribution shifts? **RQ #3:** what

factors contribute to the superior performance of **MetaMTO** and can these factors be interpreted intuitively? **RQ #4:** are our designs for learning where, what and how to transfer knowledge really play important roles? Below we first introduce the experimental settings and then discuss these RQs respectively. To ensure the reproducibility, we provide sourcecodes of our project at <https://anonymous.4open.science/r/MetaMTO-B634>.

##### A. Experimental Setup

**MetaMTO:** The low-level EMT framework is a multi-population DE that is instantiated with population size as 50 for each sub-task and with the default optimization budget  $G = 250$  generations. For each sub-task's self-evolution routine, the setting adopts DE/rand/1 and binomial crossover, with  $F = 0.5$  and  $Cr = 0.7$ . The other parts in this EMT

framework are all controlled by our proposed multi-role RL system. For the PPO we used to train our RL system, its  $T_{ppo}$  is set as 10 and  $k_{ppo}$  is set as 3, other hyper-parameters such as the epsilon-greedy rate  $\epsilon$ , learning rate  $\eta$  and discount factor  $\gamma$  are set to 0.2, 0.0003 and 0.99 respectively. The training lasts  $L = 10$  epochs.

**Baselines:** We select four representative EMT approaches. MFEA [9] and BLKT [48] represent strong EMT baselines for implicit and explicit knowledge transfer paradigms, respectively. We also include two MetaBBO baselines for direct comparison to our approach: RLMFEA [33] leverages q-learning to train a parameter control policy for MFEA; L2T [34] is proposed recently to control knowledge transfer in the explicit EMT framework. We adopt the settings in their original papers. Note that L2T is originally proposed to address 2-task optimization problem, while the testbed in our experiment is AWCCI, of which the instances are 10-task problems. We hence revise the MLP architecture in L2T to make it compatible with our settings.

**Train & Test Settings:** We have three MetaBBO approaches (i.e., RLMFEA, L2T and our **MetaMTO**) in the experiments. They are trained using our proposed AWCCI problem set. More specifically, we first set a training random seed and then generate five problem sets (i.e., AWCCI-VS ~ AWCCI-VL) according to Eq. 10. Then for each approach, we train its policy model exclusively on each of the five problem sets, which results in five trained models. During the testing phase, we first set a testing random seed and then generate five new problem sets correspondingly. The five trained models of a MetaBBO approach are then tested on corresponding problem set. For example, a model trained on AWCCI-VS training set is tested on the corresponding AWCCI-VS testing set. This testing protocol ensure a fine-grained comparison among the approaches across different MTO distributions. It is important to note that the protocol described above is designed for our in-distribution tests. In addition to this, we conduct a series of out-of-distribution experiments to assess model robustness, including evaluations across different difficulty distributions (e.g., training on one level and testing on another) and on problems with a varying number of tasks, as will be detailed in subsequent sections. Besides, MFEA and BLKT are directly tested on all testing sets. Each approach is tested for  $N = 10$  independent runs to reduce experimental variance.

**Performance Metrics:** To objectively reflect the performance difference between EMT approaches, two issues have to be addressed: i) a MTO problem include multiple sub-tasks, which anticipates proper aggregation of performance statistics; ii) the multiple sub-tasks may hold various objective value scales, which require careful normalization tricks. To this end, we propose a normalized performance metric (smaller is better) to measure the performance of an approach  $\mathcal{A}$  on an MTO instance  $\mathcal{I}$  with  $K$  sub-tasks:

$$\begin{aligned} \text{perf}(\mathcal{A}, \mathcal{I}) &= \frac{1}{K} \sum_{j=1}^K \text{perf}_j, \\ \text{perf}_j &= \frac{1}{N} \sum_{i=1}^N \frac{f_{j,i}^G - f_j^*}{f_{j,i}^0 - f_j^*}. \end{aligned} \quad (11)$$

where  $\text{perf}_j$  is the normalized performance of  $\mathcal{A}$  on  $j$ -th sub-task across the  $N = 10$  independent test runs,  $f_{j,i}^G$  denotes the optimal objective value found in  $G$  generations,  $f_{j,i}^0$  denotes the optimal in the initial population,  $f_j^*$  denotes the true (proxy<sup>2</sup>) optimal of this sub-task.

## B. Performance Comparison

1) *In-distribution Generalization (RQ #1):* In this part, we focus on the in-distribution generalization ability of different methods. For each of the augmented test sets AWCCI-VS to AWCCI-VL, we test all approaches on the 127 MTO instance for 10 independent runs. In Table IV, we report the averaged  $\text{perf}(\mathcal{A}, \mathcal{I})$  across MTO instances in each combination-based sub-set. For example, VS-4 denotes the MTO instances that include 4 of 7 different basic functions in its 10 sub-tasks. Hence, VS-4 include  $C_7^4 = 35$  instances. We also attached significance statistics beside the normalized performance of four baselines to tell how many test runs our **MetaMTO** performs significantly better or worse than the others. The last row of the table is the summary across all 6350 ( $127 \times 5 \times 10$ ) test runs.

As shown in Table IV, our approach significantly outperforms all four baselines across all test subsets in terms of the average normalized fitness and the number of winning problem instances. The superior performance validates the effectiveness of our proposed learning-based method, which automatically and comprehensively resolves the three key challenges of where to transfer, what to transfer and how to transfer. To be more specific, among the two explicit EMT algorithms, L2T outperforms BLKT on the VS and S subsets. However, its performance declines gradually on the M, L, and VL subsets, where it is surpassed by BLKT. In terms of the overall average performance, BLKT achieves superior results compared to L2T. The possible reason is that the block-based knowledge transfer mechanism in BLKT exhibits stronger optimization capability on more complex problems. Whereas, L2T suffers from limited generalization ability, which leads to a rapid degradation in solution quality. Further, in the comparison between the two implicit EMT algorithms, RLMFEA and MFEA demonstrate comparable optimization ability and RLMFEA is marginally inferior to MFEA. A possible explanation is that the learning-based knowledge transfer probability mechanism in RLMFEA lacks sufficient generalization capability, which limits its ability to effectively enhance problem solving on the AWCCI dataset. When considering both implicit and explicit EMT algorithms, RLMFEA and MFEA demonstrate notably inferior performance compared to BLKT and L2T. A likely reason is that the implicit methods follow the conventional MFEA mechanism and framework and lag behind the novel transfer strategies employed by explicit methods BLKT and L2T. Apparently, The latter ones demonstrates superior problem-solving capability. Additionally, to further present the convergence behavior of our approach and the four compared algorithms, we illustrate the convergence curves for all subsets of the test set in the Fig. 3. It can be observed that the

<sup>2</sup>A proxy optimal can be easily obtained by traversing the optimal values found by all baselines and across all test runs.

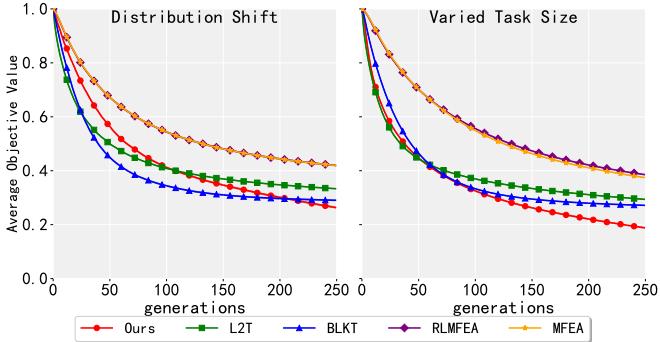


Fig. 4. Left is the comparison by shifting from training on AWCCI-VS to testing on AWCCI-VL, and right is the comparison by shifting from training on 10 sub-tasks to testing on 30 sub-tasks.

convergence curves cross-validate our observation in Table IV: our approach consistently converges to better objective values than the baselines.

2) *Out-of-distribution Generalization (RQ #2):* To validate the true generalization potential of our **MetaMTO**, we conduct two out-of-distribution generalization tests on all approaches. Specifically, we focus on scenarios which feature extreme distribution shifts and where the number of sub-tasks varies. The former case occurs when an online computing server encounter high throughput optimization tasks from different user distributions. The latter case occurs when the server encounters unstable optimization request stream. We provide the following discussion on these two aspects.

**Distribution Shift:** We pay attention to an extreme distribution shift case, where the approaches trained on AWCCI-VS are directly used to optimize instances in AWCCI-VL. We plot the convergence curves of all approaches in the left of Fig. 4 (averaged normalized objective across all tested instances and 10 independent runs). We can observe that our approach shows robust optimization performance under such severe distribution shift. In contrast, though the recent L2T framework performs with certain performance advantage in AWCCI-VS test set (in-distribution) against human-crafted baseline such as BLKT, its generalization towards unseen distributional characteristics is limited and hence underperforms BLKT in the figure. The overall results demonstrate that our **MetaMTO** is more capable of dealing with multitask scenario in the wild.

**Varied Task Size:** We next explore whether **MetaMTO** could provide stable optimization when the number of sub-tasks in a MTO instance varies. Specifically, since our proposed neural network architecture uses attention-based blocks, **MetaMTO** is capable of processing MTO problems with different task sizes. In this test, we use approaches trained on AWCCI-VS to optimize a newly constructed AWCCI-VS-30 test set, instances of which comprises 30 sub-tasks (larger than AWCCI-VS which comprises 10 sub-tasks per instance). We plot the convergence curves of all approaches in the right of Fig. 4 (averaged normalized objective across all tested instances and 10 independent runs). The results present a clear conclusion that our approach, once trained, could be used at unstable multitask environment.

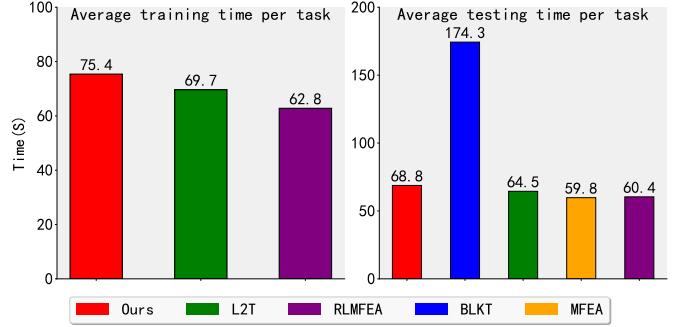


Fig. 5. Left is the average training time and right is the testing time per task.

TABLE V  
THE PROPORTION OF TIME BETWEEN NETWORK INFERENCE AND ALGORITHMIC PROBLEM SOLVING

NETWORK INFERENCE	ALGORITHM SOLVING
2.56%	97.44%

3) *Runtime Complexity:* As our approach is learning-based, its computational efficiency is therefore a critical concern. To this end, we compare the average training time of our approach with RLMFEA and L2T and compare the average testing time against all the four compared algorithms in Fig. 5. Additionally, the overall time proportion between network inference and algorithmic problem solving is presented in Table V. The results show that our approach remains the same order of average training time with the other two learning-based baselines. For inference, BLKT requires significantly more computational resources to achieve its state-of-the-art performance. This can be attributed to several computationally expensive operations in its optimization process including dividing individuals into blocks, performing block-based k-means clustering and reconstructing individuals from blocks. Our approach consumes the second in test time, but the gap with the other baselines is still acceptable. As further evidenced by Table V, the network inference time constitutes only a small fraction of the total runtime. This reveals the fact that our proposed approach improves performance with negligible computational complexity. In short, our approach achieves better solution quality without substantial time increase, making it both efficient and effective.

### C. Attribution Analysis (RQ #3)

1) *Successful Knowledge Transfer:* An intuitive factor that impacts EMT approaches is the effectiveness of knowledge transfer. Given the sub-tasks of a MTO problem at hand, positive knowledge transfer could share useful genetic information between sub-tasks hence accelerates convergence, while negative knowledge transfer may harms the co-evolution among them. To this end, we summarize knowledge transfer success rates of different methods, which is defined as the proportion of survived transferred solutions, averaged across the 250 generations of all test runs. A higher knowledge transfer success rate indicates that an approach can effectively

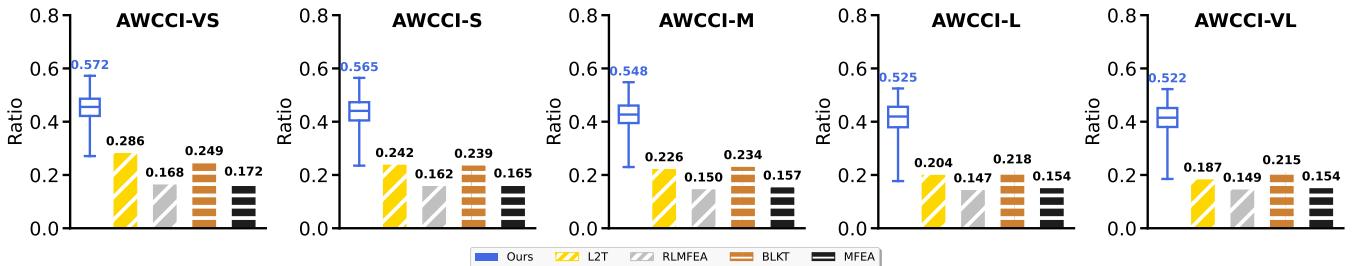


Fig. 6. Average KT success ratio over the five subsets

recognize similarity between sub-tasks and hence capable of promoting positive knowledge transfer. This metric indirectly reflects the effectiveness of the knowledge transfer mechanism, which helps interpret superiority of an approach. The results on the five test sets AWCCI-VS to AWCCI-VL are presented in Fig. 6 from left to right respectively. These results provide us a clear observation that, by using our proposed multi-role RL system to learn systematic and fine-grained knowledge transfer, a basic multi-population DE backbone can be boosted with more accurate positive knowledge transfer and hence surpasses advanced human-crafted or learning-assisted baselines.

2) *Successful Task Routing*: Given the successful knowledge transfer observed above, we further provide an in-depth analysis on the rationale behind such positive knowledge transfer of our **MetaMTO**. This analysis benefits from our attention-based neural network design, where the TR agent in the proposed multi-role RL system compute similarity scores via its attention layers. The objective is to discover whether this mechanism demonstrates the capability to identify the relevant and complementary task while filter out irrelevant ones. To create controlled test cases, we leverage two task pairs from the CEC2017 multitask optimization dataset [36], whose inter-task relationships are known: 1) the CI\_H (Complete Intersection and High Similarity) instance, which consists of two sub-tasks CI\_H\_Griewank and CI\_H\_Rastrigin; 2) the NI\_L (No Intersection and Low Similarity) instance, which consists of two sub-tasks NI\_L\_Rastrigin and NI\_L\_Schwefel. The former case has been validated as similar sub-tasks that knowledge transfer between them is preferable, while the latter case has been validated as irrelevant sub-tasks that should not have knowledge transfer. We then constructed two special, 9-task MTO problems: the first 7 tasks are selected from the WCCI2020 [73], each uses a different basic function; and the remaining two tasks are from either the CI\_H or NI\_L pairs. We then use the model trained on AWCCI-S to optimize these two 9-task instances. We show the average attention scores output by the TR agent in the trained model across all test runs and all 250 optimization generations in Fig. 7. It can be observed that the attention scores between the two CI\_H tasks are mutually the highest, indicating that the model consistently identifies them as each other's most valuable source for knowledge transfer. Conversely, the attention scores between the two NI\_L tasks are the lowest, suggesting that neither of them ever selects the other as source knowledge. These results provide direct evidence that the successful knowledge transfer of our approach is rooted in its ability to perform intelligent

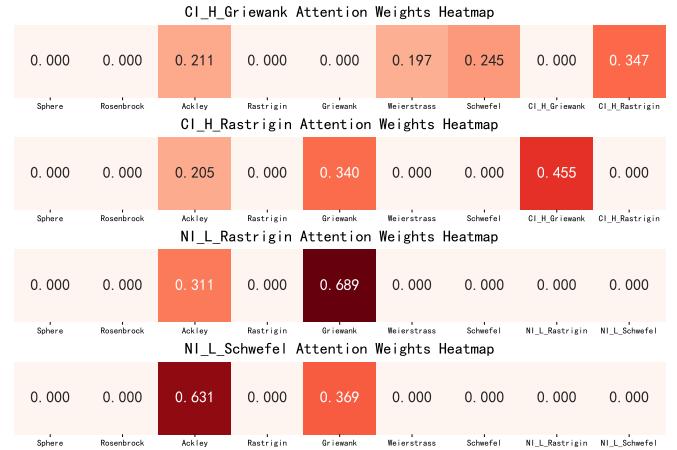


Fig. 7. Inter-task similarity recognition capability of our approach.

task routing, driven by an accurate, learned recognition of inter-task similarity.

#### D. Ablation Studies (RQ #4)

We consider five ablation variants of **MetaMTO** corresponding to functional components in our proposed multi-role RL system: 1) Ours\_w/o\_TR: we use a random (uniformly) source task selector to replace the trained TR agent; 2) Ours\_w/o\_KC: the portion of elite information to be transferred from the source sub-task to the target one no longer depends on the KC agent, instead, replaced by a random portion value sampled uniformly from [0, 1]; 3) Ours\_w/o\_OP, Ours\_w/o\_F and Ours\_w/o\_CR: these three variants represents ablation on the TSA agent group, where Ours\_w/o\_OP replaces the operator selection agent by a random selection mechanism, Ours\_w/o\_F and Ours\_w/o\_CR replace the corresponding configuration agents by fixed mutation strength 0.5 and crossover rate 0.5 respectively. To ensure an objective and comprehensive validation, the five models are trained on the five different AWCCI sets and then tested on the corresponding test sets. We leverage Wilcoxon-Holm post-hoc analysis [75] for summarizing rank-based significance of all ablated variants and original **MetaMTO** (denoted as Ours). The average rank is computed across testing results collected from 10 independent test runs. The corresponding CD diagrams are illustrated in Fig. 8, from which we can observe that:

- Consistent evidence can be attained in all five tests to support that all specific designs in our multi-role RL

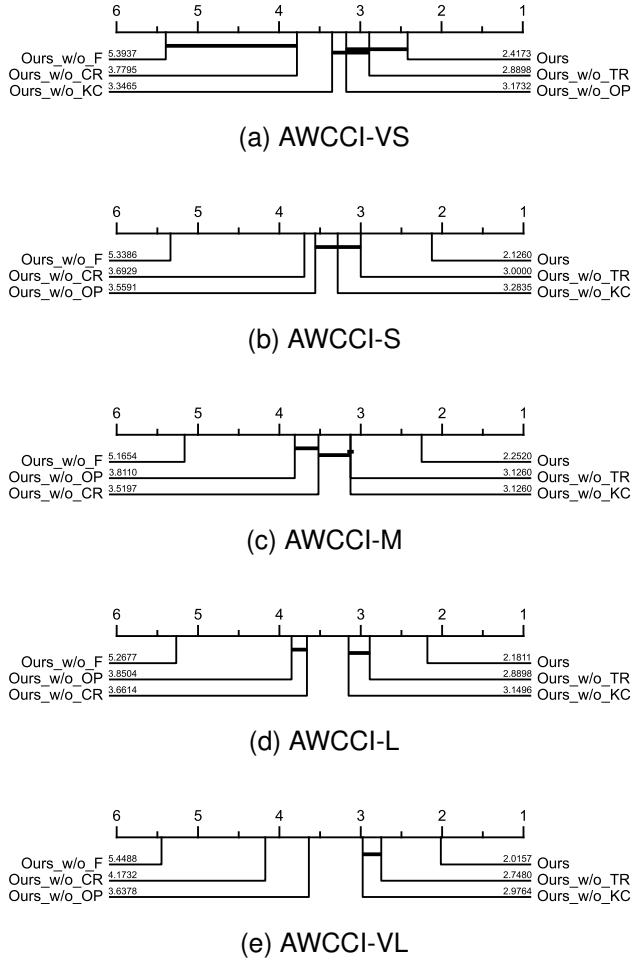


Fig. 8. Ablation studies on the proposed designs.

system make positive effects on the final performance of **MetaMTO** as a synergy, underscoring the possibility and importance of meta-learning an EMT framework in a systematic and fine-grained fashion to control where, what and how to transfer knowledge;

- **Ours v.s. Ours\_w/o\_TR:** Despite the weak significance in AWCCI-VS, **MetaMTO** significantly outperforms the variant without task routing. This demonstrates that on more challenging MTO scenarios, selecting correct knowledge source for each sub-tasks is the first key step to ensure optimization effectiveness. The meta-trained TR agent in our **MetaMTO** could accurately locate similar task pairs, which facilitates subsequent EMT operations;
- **Ours v.s. Ours\_w/o\_F:** We found that a well-trained knowledge transfer strategy adaption policy might plays the most important role in boosting the underlying EMT framework. This also aligns with the existing research focus of MetaBBO: use RL for dynamic algorithm configuration.

## V. CONCLUSION

In this paper, we propose a novel MetaBBO framework termed as **MetaMTO** to address MTO problems. Our work presents several technical innovations to complement existing

works for learning-assisted EMT. First, we develop a systematic and comprehensive MDP model to control the entire EMT process via explicit knowledge transfer. By formulating “where, what and how” to transfer knowledge as actionable algorithmic configuration, we propose a versatile multi-role RL system to achieve fine-grained and adaptive knowledge transfer. The multi-role system is equipped with attention-based architecture to facilitate both generic MTO modeling and inter-task similarity recognition. We train **MetaMTO** through PPO method and test the trained models in both in-distribution and out-of-distribution scenarios. Reproducible comparison results demonstrate that our approach consistently outperforms representative EMT approaches and advanced learning-assisted variants, while sharing similar computational overhead with these baselines. Further, the interpretability analysis reveals insightful patterns, including promising knowledge transfer rate, attention score heatmaps, and RL control preferences, which offer a reasonable explanation for the superiority of **MetaMTO**. Ablation studies on key designs in our approach not only validate their isolated effectiveness but also reveal that learning “where, what and how” to transfer knowledge in a holistic way is more promising than learning them separately.

## REFERENCES

- [1] P. Godfrey-Smith, *Darwinian populations and natural selection*. Oxford University Press, 2009.
- [2] J. H. Holland, “Genetic algorithms,” *Scientific American*, 1992.
- [3] R. Storn and K. Price, “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces,” *J. Glob. Optim.*, 1997.
- [4] P. A. Vinkar, “Evolutionary algorithms: A critical review and its future prospects,” in *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication*, 2016.
- [5] J. Zhang, Z.-h. Zhan, Y. Lin, N. Chen, Y.-j. Gong, J.-h. Zhong, H. S. Chung, Y. Li, and Y.-h. Shi, “Evolutionary computation meets machine learning: A survey,” *IEEE CIM*, 2011.
- [6] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, “A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications,” *IEEE TEVC*, 2012.
- [7] M. H. Yar, V. Rahmati, and H. R. D. Oskouei, “A survey on evolutionary computation: Methods and their applications in engineering,” *Mod. Appl. Sci.*, 2016.
- [8] A. M. Gopakumar, P. V. Balachandran, D. Xue, J. E. Gubernatis, and T. Lookman, “Multi-objective optimization for materials discovery via adaptive design,” *Sci. Rep.*, 2018.
- [9] A. Gupta, Y.-S. Ong, and L. Feng, “Multifactorial evolution: Toward evolutionary multitasking,” *IEEE TEVC*, 2015.
- [10] K. C. Tan, L. Feng, and M. Jiang, “Evolutionary transfer optimization—a new frontier in evolutionary computation research,” *IEEE CIM*, 2021.
- [11] M. Gong, Z. Tang, H. Li, and J. Zhang, “Evolutionary multitasking with dynamic resource allocating strategy,” *IEEE TEVC*, 2019.
- [12] R.-T. Liaw and C.-K. Ting, “Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems,” in *IEEE Congress on Evolutionary Computation*, 2017.
- [13] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y.-S. Ong, K.-C. Tan, and A. K. Qin, “Evolutionary multitasking via explicit autoencoding,” *IEEE TC*, 2018.
- [14] Y. Chen, J. Zhong, L. Feng, and J. Zhang, “An adaptive archive-based evolutionary framework for many-task optimization,” *IEEE TETCI*, 2019.
- [15] W. Lin, Q. Lin, L. Feng, and K. C. Tan, “Ensemble of domain adaptation-based knowledge transfer for evolutionary multitasking,” *IEEE Transactions on Evolutionary Computation*, 2024.
- [16] S.-H. Wu, Z.-H. Zhan, K. C. Tan, and J. Zhang, “Transferable adaptive differential evolution for many-task optimization,” *IEEE TC*, 2023.
- [17] H. Xu, A. K. Qin, and S. Xia, “Evolutionary multitask optimization with adaptive knowledge transfer,” *IEEE TEVC*, 2022.

- [18] Z. Liang, X. Xu, L. Liu, Y. Tu, and Z. Zhu, "Evolutionary many-task optimization based on multisource knowledge transfer," *IEEE TEVC*, 2022.
- [19] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE TEVC*, 1997.
- [20] Z. Ma, H. Guo, Y.-J. Gong, J. Zhang, and K. C. Tan, "Toward automated algorithm design: A survey and practical guide to meta-black-box optimization," *IEEE TEVC*, 2025.
- [21] X. Yang, R. Wang, K. Li, and H. Ishibuchi, "Meta-black-box optimization for evolutionary algorithms: Review and perspective," *Swarm and Evolutionary Computation*, 2025.
- [22] Z. Ma, H. Guo, J. Chen, Z. Li, G. Peng, Y.-J. Gong, Y. Ma, and Z. Cao, "Metabox: A benchmark platform for meta-black-box optimization with reinforcement learning," *NeurIPS*, 2023.
- [23] Z. Ma, Y.-J. Gong, H. Guo, W. Qiu, S. Ma, H. Lian, J. Zhan, K. Chen, C. Wang, Z. Huang, Z. Huang, G. Peng, R. Cheng, and Y. Ma, "Metabox-v2: A unified benchmark platform for meta-black-box optimization," in *NeurIPS*, 2025.
- [24] S. Thrun and L. Pratt, "Learning to learn: Introduction and overview," in *Learning to learn*, 1998.
- [25] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.
- [26] H. Guo, Z. Ma, J. Chen, Y. Ma, Z. Cao, X. Zhang, and Y.-J. Gong, "Configx: Modular configuration for evolutionary algorithms via multi-task reinforcement learning," in *AAAI*, 2025.
- [27] X. Li, K. Wu, Y. B. Li, X. Zhang, H. Wang, and J. Liu, "Pretrained optimization model for zero-shot black box optimization," in *NeurIPS*, 2024.
- [28] M. Chen, C. Feng, and R. Cheng, "Metade: Evolving differential evolution by differential evolution," *IEEE TEVC*, 2025.
- [29] K. Xue, J. Xu, L. Yuan, M. Li, C. Qian, Z. Zhang, and Y. Yu, "Multi-agent dynamic algorithm configuration," *NeurIPS*, 2022.
- [30] S. Shao, Y. Tian, and Y. Zhang, "Deep reinforcement learning assisted surrogate model management for expensive constrained multi-objective optimization," *Swarm and Evolutionary Computation*, 2025.
- [31] H. Lian, Z. Ma, H. Guo, T. Huang, and Y.-J. Gong, "Rlemmo: Evolutionary multimodal optimization assisted by deep reinforcement learning," in *ACM GECCO*, 2024.
- [32] G. Li, Y. Xin, J. Niu, Z. Wang, J. Chen, and F. Wu, "Reinforcement learning assisted automatic niche selection for constrained multimodal multi-objective optimization," *Expert Systems with Applications*, 2026.
- [33] S. Li, W. Gong, L. Wang, and Q. Gu, "Evolutionary multitasking via reinforcement learning," *IEEE TETCI*, 2023.
- [34] S.-H. Wu, Y. Huang, X. Wu, L. Feng, Z.-H. Zhan, and K. C. Tan, "Learning to transfer for evolutionary multitasking," *IEEE TC*, 2025.
- [35] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. The MIT Press, 1998.
- [36] 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.
- [37] Z. Tan, L. Luo, and J. Zhong, "Knowledge transfer in evolutionary multi-task optimization: A survey," *Applied Soft Computing*, 2023.
- [38] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: Mfea-ii," *IEEE TEVC*, 2019.
- [39] X. Zheng, A. K. Qin, M. Gong, and D. Zhou, "Self-regulated evolutionary multitask optimization," *IEEE TEVC*, 2019.
- [40] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm," in *IJCAI*, 2018.
- [41] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, "Linearized domain adaptation in evolutionary multitasking," in *IEEE Congress on Evolutionary Computation*, 2017.
- [42] 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 TC*, 2020.
- [43] W. Li, X. Gao, L. Wang, and Q. Xu, "Evolutionary multitasking based on team learning strategy," in *2022 6th Asian Conference on Artificial Intelligence Technology (ACAIT)*, 2022.
- [44] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE TC*, 2016.
- [45] Y. Feng, L. Feng, Y. Hou, and K. C. Tan, "Large-scale optimization via evolutionary multitasking assisted random embedding," in *IEEE Congress on Evolutionary Computation*, 2020.
- [46] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, 1951.
- [47] C. Wang, J. Liu, K. Wu, and Z. Wu, "Solving multitask optimization problems with adaptive knowledge transfer via anomaly detection," *IEEE TEVC*, 2021.
- [48] Y. Jiang, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Block-level knowledge transfer for evolutionary multitask optimization," *IEEE TC*, 2024.
- [49] A. Gupta, L. Zhou, Y.-S. Ong, Z. Chen, and Y. Hou, "Half a dozen real-world applications of evolutionary multitasking, and more," *IEEE CIM*, 2022.
- [50] 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 TC*, 2020.
- [51] L. Feng, L. Zhou, A. Gupta, J. Zhong, Z. Zhu, K.-C. Tan, and K. Qin, "Solving generalized vehicle routing problem with occasional drivers via evolutionary multitasking," *IEEE TC*, 2019.
- [52] Z. Ma, H. Guo, Y. Gui, and Y.-J. Gong, "An efficient computational approach for automatic itinerary planning on web servers," in *ACM GECCO*, 2021.
- [53] D. Wu and X. Tan, "Multitasking genetic algorithm (mtga) for fuzzy system optimization," *IEEE TFS*, 2020.
- [54] Z. Ma, J. Chen, H. Guo, and Y.-J. Gong, "Neural exploratory landscape analysis for meta-black-box-optimization," in *ICLR*, 2025.
- [55] H. Guo, Z. Ma, Y. Ma, X. Zhang, W.-N. Chen, and Y.-J. Gong, "Designx: Human-competitive algorithm designer for black-box optimization," *NeurIPS*, 2025.
- [56] Z. Ma, Z. Huang, J. Chen, Z. Cao, and Y.-J. Gong, "Surrogate learning in meta-black-box optimization: A preliminary study," in *ACM GECCO*, 2025.
- [57] Z. Ma, Z. Cao, Z. Jiang, H. Guo, and Y.-J. Gong, "Meta-black-box-optimization through offline q-function learning," in *ICML*, 2025.
- [58] H. Guo, S. Ma, Z. Huang, Y. Hu, Z. Ma, X. Zhang, and Y.-J. Gong, "Reinforcement learning-based self-adaptive differential evolution through automated landscape feature learning," in *ACM GECCO*, 2025.
- [59] C. Wang, Z. Ma, Z. Cao, and Y.-J. Gong, "Instance generation for meta-black-box optimization through latent space reverse engineering," *arXiv preprint arXiv:2509.15810*, 2025.
- [60] X. Chen, Y. Huang, W. Zhou, and L. Feng, "Evolutionary multitasking via artificial neural networks," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2021.
- [61] Z.-F. Xue, Z.-J. Wang, Z.-H. Zhan, S. Kwong, and J. Zhang, "Neural network-based knowledge transfer for multitask optimization," *IEEE TC*, 2024.
- [62] K. Huang, X. Wang, and Y. Cai, "Surrogate-assisted task selection for evolutionary multitasking optimization," in *IEEE 2nd International Conference on Software Engineering and Artificial Intelligence*, 2022.
- [63] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
- [64] M. Han, X. Li, K. Wu, X. Zhang, and H. Wang, "Enhancing zero-shot black-box optimization via pretrained models with efficient population modeling, interaction, and stable gradient approximation," *NeurIPS*, 2025.
- [65] Q. Renau, J. Dreo, C. Doerr, and B. Doerr, "Expressiveness and robustness of landscape features," in *ACM GECCO*, 2019.
- [66] Q. Renau, C. Doerr, J. Dreo, and B. Doerr, "Exploratory landscape analysis is strongly sensitive to the sampling strategy," in *PPSN*, 2020.
- [67] M. V. Seiler, P. Kerschke, and H. Trautmann, "Deep-ela: Deep exploratory landscape analysis with self-supervised pretrained transformers for single-and multi-objective continuous optimization problems," *Evolutionary Computation*, 2025.
- [68] D. Hadfield-Menell, S. Mili, P. Abbeel, S. J. Russell, and A. Dragan, "Inverse reward design," *NeurIPS*, 2017.
- [69] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, 2017.
- [70] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.
- [71] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales, "Learning to generalize: Meta-learning for domain generalization," in *AAAI*, 2018.
- [72] 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.
- [73] L. Feng, K. Qin, A. Gupta, Y. Yuan, Y.-S. Ong, and X. Chi, "Ieee wcci 2020 competition on evolutionary multi-task optimization," in *WCCI*, 2020. [Online]. Available: [http://www.bdsc.site/websites/MTO\\_competition\\_2020/#benchmark](http://www.bdsc.site/websites/MTO_competition_2020/#benchmark)
- [74] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

- [75] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Deep learning for time series classification: a review,” *Data Mining and Knowledge Discovery*, 2019.
- [76] G. W. Stewart, “The efficient generation of random orthogonal matrices with an application to condition estimators,” *SIAM Journal on Numerical Analysis*, vol. 17, no. 3, pp. 403–409, 1980.

## APPENDIX

### I. TECHNICAL DETAILS

#### A. Computational details of the state features

In this section, we provide a description of the computational details in constructing the 5-dimensional feature vector that characterizes the optimization state information of each sub-task at every step of the evolutionary multitasking (EMT) process. To elaborate, the feature  $s_1$  characterizes the diversity of a sub-task population in the decision space. Consider a population  $X = \{x_1, x_2, \dots, x_N\}$  of  $N$  individuals for a given sub-task, where each individual is encoded in the unified space  $[0, 1]^D$ . The computation of  $s_1$  proceeds in two steps: first, the standard deviation is calculated across the population for each dimension in the decision space, resulting in a  $D$ -dimensional vector of standard deviations; second,  $s_1$  is obtained by taking the mean of this vector. This process can be mathematically formulated by the following equation:

$$\sigma^j = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_{i,j} - \mu_j)^2} \quad (12)$$

$$s_1 = \frac{1}{D} \sum_{j=1}^D \sigma^j \quad (13)$$

where  $\sigma_j$  represents the standard deviation of the  $j$ -th dimension across the population and  $\mu_j$  is the mean value of the population along the  $j$ -th dimension. The second feature  $s_2$  characterizes the convergence of the sub-task population in the objective space. Let  $f$  denote the objective function of the sub-task, which is defined as a minimization problem. The feature  $s_2$  is computed as follows. First, leveraging the known (proxy) optimal value  $f^*$  and the maximum objective value  $f_{max}^0$  from the initial population as the baseline, the objective value of each individual is normalized to  $[0, 1]$ . Then,  $s_2$  is defined as the standard deviation of these normalized objective values. This process can be formally described by the following equations:

$$\hat{f}(x_i) = \frac{f(x_i) - f^*}{f_{max}^0 - f^*} \quad (14)$$

$$s_2 = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{f}(x_i) - \mu_{\hat{f}})^2} \quad (15)$$

where  $\hat{f}(x_i)$  is the normalized objective value of individual  $x_i$  and  $\mu_{\hat{f}}$  is the mean value of the normalized objective values. The third feature  $s_3$  quantifies the degree of stagnation encountered during the optimization process. The stagnation situation is defined as the absence of improvement in the best-found objective value compared to the previous generation.  $s_3 = \frac{G_{stagnation}}{G}$  is then defined as the ratio of the cumulative

stagnation count  $G_{stagnation}$  to the total number of generations  $G$ . The fourth feature  $s_4$  is a bool value that signals whether the best-so-far solution is updated. Let  $f(X^t)$  and  $f(X^{t+1})$  represent the best-so-far objective values at steps  $t$  and  $t + 1$  respectively. Then, the mathematic formulation of the feature is  $s_4 = \mathbb{I}(f(X^{t+1}) < f(X^t))$ , where  $\mathbb{I}(\cdot)$  is the indicator function. Consequently,  $s_4 = 1$  indicates a successful update of the best-so-far objective value. The fifth feature  $s_5$  quantifies the effectiveness of knowledge transfer for a sub-task. Let  $N_{tra}$  be the total number of transferred individuals of the sub-task and  $N_{sur}$  be the number of these individuals that successfully survive the selection process. The feature  $s_5 = \frac{N_{sur}}{N_{tra}}$  is defined as the survival rate of the transferred solutions. Finally, the complete state features of a sub-task is constructed as a vector containing the five features from  $s_1$  to  $s_5$ , which holistically and succinctly encapsulates optimization information from many key aspects during the EMT process.

#### B. Details of the problem set construction

The representative Multitask Optimization (MTO) problem sets such as CEC2017 [72] and WCCI2020 [73] are primarily limited by two issues: i) sparse and non-uniform distribution and ii) insufficient problem instances with various sub-task combinations. To address these issues, we propose a more comprehensive MTO problem set termed as AWCCI. In AWCCI, the sub-tasks within each MTO problem instance are generated by applying rotation and shifting operations to the following seven benchmark functions:

##### 1) Sphere:

$$f(\mathbf{x}) = \sum_{i=1}^D z_i^2 \quad (16)$$

$$\mathbf{x} \in [-100, 100]^D$$

##### 2) Rosenbrock:

$$f_2(\mathbf{x}) = \sum_{i=1}^{D-1} \left( 100 (z_i^2 - z_{i+1})^2 + (z_i - 1)^2 \right) \quad (17)$$

$$\mathbf{x} \in [-50, 50]^D$$

##### 3) Ackley:

$$f_3(\mathbf{x}) = -20 \exp \left( -0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D z_i^2} \right)$$

$$- \exp \left( \frac{1}{D} \sum_{i=1}^D \cos(2\pi z_i) \right) + 20 + e \quad (18)$$

$$\mathbf{x} \in [-50, 50]^D$$

##### 4) Rastrigin:

$$f_4(\mathbf{x}) = \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i) + 10) \quad (19)$$

$$\mathbf{x} \in [-50, 50]^D$$

TABLE VI  
CONFIGURATION OF THE AWCCI SUBSET

Problem subset	Shifting vector configuration( $\mathbf{s}$ )
AWCCI-VS	$\mathbf{s} \sim 0.05 \times (lb + U[0, 1]^D \times (ub - lb))$
AWCCI-S	$\mathbf{s} \sim 0.1 \times (lb + U[0, 1]^D \times (ub - lb))$
AWCCI-M	$\mathbf{s} \sim 0.2 \times (lb + U[0, 1]^D \times (ub - lb))$
AWCCI-L	$\mathbf{s} \sim 0.3 \times (lb + U[0, 1]^D \times (ub - lb))$
AWCCI-VL	$\mathbf{s} \sim 0.4 \times (lb + U[0, 1]^D \times (ub - lb))$

seven benchmark functions in  $\mathbf{f}$ . More specifically, each combination specifies the possible benchmark functions that each sub-task may adopt. Then for each combination, we generate a problem instance comprising 10 sub-tasks, with each sub-task randomly designated as one of the constitutive benchmark functions (with replacement) and instantiated by a randomly generated rotation matrix and shifting vector. Since the distribution of the shifting vector is associated with each subset, the generation process is repeated for all the five subsets respectively. Consequently, we generate 127 problem instances for each subset, resulting in a total of  $127 \times 5 = 635$  problem instances for the entire AWCCI problem set.

5) Griewank:

$$f_5(\mathbf{x}) = 1 + \frac{1}{4000} \sum_{i=1}^D z_i^2 - \prod_{i=1}^D \cos\left(\frac{z_i}{\sqrt{i}}\right) \quad (20)$$

$$\mathbf{x} \in [-100, 100]^D$$

6) Weierstrass:

$$f_6(\mathbf{x}) = \sum_{i=1}^D \left( \sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k (z_i + 0.5))] \right) - D \sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k \cdot 0.5)] \quad (21)$$

$$a = 0.5, b = 3, k_{max} = 20$$

$$\mathbf{x} \in [-0.5, 0.5]^D$$

7) Schwefel:

$$f_7(\mathbf{x}) = 418.9829 \times D - \sum_{i=1}^D z_i \sin(|z_i|^{\frac{1}{2}}) \quad (22)$$

$$\mathbf{x} \in [-500, 500]^D$$

where for each benchmark function the dimension  $D$  is 50 and  $\mathbf{z} = \mathbf{w}^T(\mathbf{x} - \mathbf{s})$ . Note that  $\mathbf{w}$  denotes the random rotation matrix by applying Householder transformation [76] and  $\mathbf{s}$  denotes the shifting vector. The  $lb$  and  $ub$  denote the lower and upper bound of the shifting vector, which is within the same searching range as the decision space.

For the first issue, we apply five various distribution shifting levels  $l \in \{0.05, 0.1, 0.2, 0.3, 0.4\}$  with incremental difficulties. As the configuration details presented in Table VI, according to different shifting levels, the shifting vector of the sub-task is uniformly distributed within different bounded range. As a result, the AWCCI is further partitioned into five subsets, denoted respectively as AWCCI-VS (with very small shifting), AWCCI-S (with small shifting), AWCCI-M (with median shifting), AWCCI-L (with large shifting) and AWCCI-VL (with very large shifting) corresponding to the specific shifting levels.

For the second issue, let  $\mathbf{f} := \{f_1, f_2, f_3, f_4, f_5, f_6, f_7\}$  be the benchmark function set, we begin by constructing  $\sum_{i=1}^7 C_7^i = 127$  possible non-empty combinations of the