TOWARDS ADAPTIVE KNOWLEDGE TRANSFER IN EVOLUTIONARY TRANSFER OPTIMIZATION

WU LIN

PhD

The Hong Kong Polytechnic University

2025

The Hong Kong Polytechnic University
Department of Computing

Towards Adaptive Knowledge Transfer in Evolutionary Transfer Optimization

Wu Lin

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
June 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Wu Lin

Abstract

Evolutionary transfer optimization (ETO) is an emerging search paradigm, which integrates evolutionary algorithms (EAs) with transfer learning techniques. Learning and transferring useful knowledge across related problems can reduce repeated searches, enabling traditional EAs to achieve better optimization efficiency and performance on various complex problems. Generally, the design of ETO approaches grapples with three critical issues concerning knowledge transfer: 1) what to transfer, 2) how to transfer, and 3) when to transfer. Considering what to transfer, it involves identifying the type of knowledge and deciding which one to transfer among all available candidates. Regarding how to transfer, it focuses on the methodology design for implementing knowledge transfer. As for the issue of when to transfer, it aims to identify the optimal timing or the appropriate extent for deciding how much knowledge to transfer. However, in existing ETO studies, most deterministic methods lack the adaptability and flexibility when addressing the above three issues, severely limiting the robustness and effectiveness of knowledge transfer in enhancing the optimization efficiency and performance of EAs. To achieve more effective and robust performance, this thesis focuses on studying and designing adaptive knowledge transfer methods to intelligently address one or more of the three issues.

Firstly, to adaptively decide what to transfer, this thesis proposes a fuzzy classifier-assisted solution transfer method to identify the most useful solution for transfer in evolutionary sequential transfer optimization (ESTO). By constructing the training

data, the fuzzy classifier is built to estimate the solution usefulness of all available source tasks for the target task. Compared to existing solution transfer methods, the proposed method not only estimates whether one source task is useful or useless but also further quantifies the degree of its usefulness when it is estimated to be useful. In this way, the most useful solution is accurately selected from useful source tasks for knowledge transfer. This effectively accelerates the optimization of the target task by adaptively selecting the most useful solution from available source tasks for knowledge transfer.

Secondly, to adaptively decide how to transfer, this thesis proposes an ensemble method to combine multiple domain adaptation methods for evolutionary multitasking (EMT), mitigating the unique biases of each domain adaptation method. It smartly addresses the balance between efficacy and diversity when determining which one domain adaptation method for use, which further enhances knowledge transferability across tasks in EMT. The proposed methodology clearly differentiates from existing ensemble methods by integrating a novel adaptive selection mechanism that considers both past performance and diversity of candidate domain adaptation methods. This could potentially lead to more robust and effective multitasking performance in comparison with existing non-adaptive approaches that adopt one deterministic domain adaptation method to address the issue of how to transfer.

Lastly, to adaptively decide when to transfer and how to transfer, this thesis proposes a fuzzy logic-based method in EMT. The proposed method includes two fuzzy logic-based components. To effectively adapt the transfer extent along the multitasking search process, a fuzzy logic-based parameter adaption component is developed to dynamically adjust the value of the transfer parameter, thereby alleviating the risk of

negative transfer. To adaptively select the most promising method for knowledge transfer, a fuzzy logic-based selection component is developed to select the optimal transfer method from multiple candidates, thereby enhancing knowledge transferability across tasks. The proposed fuzzy logic-based methodology clearly differentiates from existing methods by employing fuzzy logic to effectively process fuzzy and inaccurate information, facilitating effective knowledge transfer.

Publications Arising from the Thesis

1. Wu Lin, Qiuzhen Lin, Liang Feng, and Kay Chen Tan, "Ensemble of Domain Adaptation-Based Knowledge Transfer for Evolutionary Multitasking", Accepted by *IEEE Transactions on Evolutionary Computation (TEVC)* (2024).
2. Wu Lin, Qiuzhen Lin, Liang Feng, and Kay Chen Tan, "Fuzzy Classifier-Assisted Solution Transfer for Evolutionary Sequential Transfer Optimization", manuscript submitted to *IEEE Transactions on Cybernetics (TCYB)*.
3. Wu Lin, Qiuzhen Lin, Liang Feng, and Kay Chen Tan, "Fuzzy Logic-Based Adaptive Knowledge Transfer for Evolutionary Multitasking", manuscript submitted to *IEEE Transactions on Fuzzy Systems (TFS)*.
4. Wu Lin, Qiuzhen Lin, Xiaoming Xue, and Kay Chen Tan, "Sequential Transfer via Clustering-Based Similarity Measurement for Faster Trajectory Optimization", Accepted by *2024 IEEE Conference on Artificial Intelligence (IEEE CAI 2024)*, 2024.

Acknowledgments

During my doctoral journal at The Hong Kong Polytechnic University, I received much support and help from many individuals, enabling me to complete this thesis successfully. I am profoundly grateful for the contributions of each person who has assisted me along this path.

First and foremost, my deepest gratitude goes to my supervisor, Professor Kay Chen Tan. He is an outstanding scholar who has made remarkable achievements in the evolutionary computation community. I am incredibly fortunate to have the opportunity to pursue my doctoral degree under his supervision. The unwavering support and invaluable guidance from Professor Tan have been pivotal during challenging moments in my research. The knowledge and skills I have acquired under his mentorship will immensely benefit me on my future academic path.

I sincerely thank Professor Qiuzhen Lin and Professor Liang Feng for their continuous guidance and help. They have spent much time providing valuable suggestions and comments for my research work, significantly enhancing the quality and depth of my doctoral thesis. I am grateful for their generous support and the time they dedicat-

ed to engaging with my research.

My sincere appreciation also goes to all the members of my research group, including Dr. Jibin Wu, Dr. Zhian Huang, Dr. Songbai Liu, Dr. Xun Zhou, Dr. Rui Liu, Dr. Yinglan Feng, Mr. Xiaoming Xue, Mr. Yao Hu, Mr. Zhenzhong Wang, Ms. Xinyi Chen, Mr. Chenxiang Ma, Mr. Xiang Hao, and all new members who have joined us. The support and encouragement from this research group have been invaluable.

I am also grateful to The Hong Kong Polytechnic University for the resources provided, including teaching materials, laboratory equipment, and the opportunity to learn from many excellent scholars through compulsory subjects and research seminars.

Lastly, I thank my family and friends. Their support and understanding have always motivated me to pursue my doctoral degree.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Optimization problems exist in various scenarios, including data science [1], embodied intelligence [2], unmanned systems planning [3], complex engineering design [4], smart manufacturing [5], and software engineering [6]. In mathematics, solving an optimization problem is to obtain the best solution from its feasible search space. Many practical optimization problems may possess complex characteristics, such as non-convexity and non-differentiability, which significantly complicate the problem-solving process. Thus, traditional mathematical methods, such as linear programming [7], quadratic programming [8], and convex optimization [9], cannot effectively solve these complex optimization problems. In contrast to traditional mathematical methods, evolutionary algorithms (EAs) are a class of population-based metaheuristic search approaches [10], [11], [12], [13]. Generally, given an optimization problem, one clas-

sical EA starts by randomly sampling solutions in the search space to form a population. Subsequently, this population undergoes crossover, mutation, and selection, continuously evolving towards the regions where the optimal solution lies until the stopping criterion is satisfied. The iterative procedure facilitates the exploration of optimal solutions for a diverse range of complex optimization problems. Without strong mathematical theories and related knowledge, EAs can be easily implemented to solve various optimization problems with complex characteristics, such as multiple conflicting objective functions [14], expensive evaluation costs [15], multiple local optima [16], constraints [17], and a large number of decision variables [18].

In the past years, various advanced evolutionary components, such as crossover, mutation, and selection, have been developed to enhance the performance of EAs. While EAs have gained great success, they typically solve one optimization problem by conducting the evolutionary search process from scratch. In fact, problems seldom exist in isolation [19], [20]. In the area of machine learning, transfer learning (TL) has been proven to be effective for improving generalization about the current task by exploiting the knowledge from a previous task [21], [22], [23]. Here, the task can be various classification and regression problems [24], [25]. In the context of evolutionary computation, employing search experience of previously solved optimization problems can avoid repeated searches to find the optimal solutions when solving new optimization problems, thereby improving the optimization efficiency and performance.

Inspired by TL, evolutionary transfer optimization (ETO) is proposed as a new search paradigm, which integrates EAs with TL techniques [20]. The ability of learning and transferring knowledge from related problems enables traditional EAs to possess better efficiency and performance in solving various optimization problems. For optimi-

zation problems, only objective function and limited problem-specific data can be obtained by iteratively conducting the evolutionary search process, presenting significant differences from classification and regression problems in machine learning. Therefore, traditional TL techniques are not suitable for optimization tasks, resulting in new requirements for developing ETO approaches, including new representations of knowledge and learning methods for knowledge transfer across various optimization tasks. In the literature, various advanced ETO approaches have been developed to address problems with diverse characteristics, including dynamic optimization [26], multitask optimization [27], complex optimization [28], multi/many-objective optimization [29], large-scale optimization [30], constrained optimization [31], bi-level optimization [32], combinatorial optimization [33], high-dimensional feature selection [34], and neural architecture search [35].

While there are different conceptual realizations for ETO, such as evolutionary sequential transfer optimization (ESTO), evolutionary multitasking optimization (EMT), and multiform optimization (MFO), the design of ETO approaches grapples with three pivotal issues concerning knowledge transfer: 1) what to transfer, 2) how to transfer, and 3) when to transfer [36], [37]. Determining what to transfer focuses on the types of knowledge, including heuristic algorithms [38], [39], [40], configured parameters [41], [42], [43], and evaluated solutions [44], [45], [46], [47]. Due to the ease of use, transferring knowledge in the form of solutions has attracted increasing attention in existing ETO studies [37]. Regarding the evolutionary search process, there are usually multiple solutions at each generation. Therefore, given the type of knowledge in the form of solutions, determining what to transfer is simplified to determine which one or some solution(s) for transfer. Determining how to transfer focuses on the design of the transfer mechanism, aiming to implement effective

knowledge transfer across various complex problems. There are implicit transfer approaches [27] and explicit transfer approaches [48]. In terms of determining when to transfer, it aims to identify the optimal timing or the appropriate extent for knowledge transfer [49], [50]. The former can be employed to identify different scenarios, such as when to or when not to transfer during the evolutionary search process. The latter indicates how much knowledge to transfer across tasks. Within existing ETO studies, the above three issues are usually addressed by deterministic methods, which may potentially undermine the effectiveness of knowledge transfer due to a lack of adaptability and flexibility [36], [37], [51]. To achieve more effective and robust knowledge transfer in ETO, adaptive knowledge transfer methods should be designed to intelligently decide what to transfer, how to transfer, and when to transfer during the evolutionary search process. Therefore, this thesis is motivated to study and design adaptive knowledge transfer methods in terms of the following three considerations:

- Deciding What to Transfer

In designing ETO approaches, transferring knowledge in the form of solutions is the most straightforward way, which has attracted increasing attention [37]. As a conceptual realization of ETO, ESTO aims to speed up the optimization of a new task (called target task) by utilizing the search experience of previously solved tasks (called source tasks). Hence, in the context of ESTO, as the number of previously solved source tasks increases, their optimal solutions are available, which may be useful for accelerating the optimization of the current target task. In existing ETO studies, several methods have been developed to measure the usefulness of solutions by employing distance metrics or traditional machine learning (ML) models. Using distance metrics cannot determine whether one source task (its optimal solution) is useful for accelerating the optimization of the target task, while these ML models are unable to

quantify the degree of solution usefulness, which may result in ineffective knowledge transfer. Therefore, adaptively deciding the most promising solution from all candidates needs to judge whether there are useful sources for the target task. The most useful solution is selected from all useful candidates if they exist. Otherwise, no solution is selected for knowledge transfer.

- Deciding How to Transfer

EMT is another conceptual realization of ETO, aiming to solve multiple tasks simultaneously. In existing ETO studies, various methods of knowledge transfer have been designed to transfer knowledge across tasks [51]. However, considering domain adaptation methods for conducting knowledge transfer, the differences in their design mechanisms may lead to unique biases of these methods when transferring knowledge between two tasks. One specific transfer method often exhibits the superiority in its preferred transfer scenario, while it may perform poorly in other transfer scenarios. Practical optimization problems usually possess various complex characteristics, making it challenging for any single transfer method to clearly outperform other approaches on existing optimization problems. Therefore, adaptively deciding the optimal method from multiple candidates can take full advantages of the strength of each method for conducting knowledge transfer across tasks, thereby further enhancing the effectiveness and robustness of knowledge transfer.

- Deciding When to Transfer

The transfer timing or the transfer extent can determine the frequency of conducting knowledge transfer across tasks, which may vary as the correlation of tasks changes

[49], [50]. Thus, it is difficult to predefine and fix the optimal timing or the appropriate extent for knowledge transfer in handling various types of problems. For example, frequent knowledge transfer can encourage positive transfer when two tasks are highly related. In such a scenario, the shared knowledge can enhance the optimization efficiency and performance on the target task by reducing repeated evolutionary searches to find its global optimum. However, frequent knowledge transfer may increase the risk of negative transfer in the situation that two tasks possess low correlation due to the waste of unnecessary computational costs. Conversely, in the case that two tasks are unrelated, inactive knowledge transfer can alleviate the risk of negative transfer to some extent. However, inactive knowledge transfer may potentially diminish positive transfer in the situation that two tasks are highly related, as useful knowledge is not fully utilized to accelerate the evolutionary search process. Therefore, adaptively deciding the transfer timing or the transfer extent is critical in further enhancing the effectiveness of robustness of knowledge transfer.

1.2 Contributions

As mentioned above, the three issues of deciding what to transfer, how to transfer, and when to transfer remain much room to be explored in existing ETO studies. To achieve more effective and robust knowledge transfer, this thesis focuses on designing adaptive knowledge transfer methods, aiming to intelligently decide one or more of the three key issues. The main contributions of this thesis are summarized as follows:

1) Adaptively Deciding What to Transfer in ESTO

- This thesis proposes a fuzzy classifier-assisted method (FCM) to select the most useful solution for knowledge transfer in ESTO. By constructing the training data, the fuzzy classifier is built to estimate the solution usefulness of all available source tasks. Compared to existing methods, the proposed method not only estimates whether one source task is useful or useless but also further quantifies the degree of its usefulness if it is useful. In this way, the most useful solution is accurately selected from useful source tasks for knowledge transfer.

- This thesis also presents the implementation of an ESTO algorithm with the proposed method (ESTOA-FCM). The experimental results show the competitive performance of ESTOA-FCM when compared with existing ESTO algorithms.

2) Adaptively Deciding How to Transfer in EMT

- This thesis proposes a domain adaptation-based ensemble (DAE) method to select the promising domain adaptation method for knowledge transfer in EMT. It smartly addresses the balance between efficacy and diversity when determining which one domain adaptation method for use, thereby taking full advantages of the strengths of each domain adaptation method to further enhance knowledge transferability across tasks. This could potentially lead to more robust and effective multitasking performance in comparison with existing non-adaptive approaches that adopt one deterministic domain adaptation method.

- This thesis also presents the implementation of incorporating the proposed ensemble method into an EMT framework. The experimental results validate that incorporating DAE method into three competitive EMT algorithms can significantly improve their performance for solving different multitasking test problems. More-

over, a canonical EMT algorithm enhanced by DAE (called MFEA-DAE) outperforms five recent EMT algorithms on most cases of the multitasking test problems used, and the effectiveness of DAE is also validated on a practical case.

3) Adaptively Deciding When to Transfer and How to Transfer in EMT

- This thesis proposes a fuzzy logic-based method in EMT. The proposed method includes two fuzzy logic-based components. To effectively adapt the transfer extent along the multitasking search process, a fuzzy logic-based parameter adaption component is developed to dynamically adjust the value of the transfer parameter, thereby alleviating the risk of negative transfer. To adaptively select the most promising method for conducting knowledge transfer, a fuzzy logic-based selection component is developed to select the optimal transfer method from multiple candidates, thereby enhancing knowledge transferability across tasks. The proposed fuzzy logic-based methodology clearly differentiates from existing methods by employing fuzzy logic to effectively process fuzzy information collected along the evolutionary search process, thereby enhancing the effectiveness of knowledge transfer.

- This thesis also presents the implementation of a new MFEA (called MFEA-FLM) by incorporating the above fuzzy logic-based method into an EMT framework. The experimental results validate the effectiveness of the proposed method and show the competitive performance of MFEA-FLM when compared with other EMT algorithms.

1.3 Thesis Organization

This thesis consists of six chapters. Chapter 1 provides the basic introduction of ETO, while the rest of the chapters are listed as follows:

Chapter 2 provides some basic concepts of ETO, and then summarizes the related work in ETO.

In Chapter 3, to adaptively decide what to transfer in ESTO, a fuzzy classifier-assisted method is proposed to select the most useful solution by measuring the usefulness of solutions for the target task. The comparison experiments are conducted on a series of test problems, demonstrating that the proposed method can effectively select the most useful source solution to speed up the optimization process of the target task.

In Chapter 4, to adaptively decide how to transfer in EMT, an ensemble method of domain adaption is proposed to effectively select the promising domain adaptation method from multiple candidates for knowledge transfer. The comparison experiments are conducted on a series of test problems, showing that the proposed method can fully utilize the strengths of multiple domain adaption methods to facilitate effective knowledge transfer.

In Chapter 5, to adaptively decide when to transfer and how to transfer in EMT, a fuzzy logic-based method is proposed to adapt the transfer extent and select the promising transfer method along the multitasking search process. The comparison experiments are conducted on a series of test problems, demonstrating that the proposed method can achieve more robust and effective knowledge transfer by dynamically adapting the transfer extent and adaptively selecting the promising transfer method.

Chapter 6 concludes this thesis and provides future research directions for promoting the development of ETO.

Chapter 2

Literature Review

In ETO, there are three different conceptual realizations: 1) evolutionary sequential transfer optimization [47], 2) evolutionary multitasking optimization [51], and 3) multiform optimization [19]. In fact, multiform optimization can be regarded as a particular branch of evolutionary multitasking optimization, where multiple optimization tasks consist of one original problem and its alternative problem formulations. Thus, this thesis focuses on the first two categories. The details of their mathematical definitions are first given in subsection 2.1. Despite the technical distinctions among different conceptual realizations, in designing ETO approaches, there are three critical issues concerning knowledge transfer: 1) what to transfer, 2) how to transfer, and 3) when to transfer. The following subsections 2.2, 2.3, and 2.4 present the related research work in terms of the three issues, respectively.

2.1 Evolutionary Transfer Optimization

In general, the traditional EA solves one optimization problem at one execution by iteratively evolving the population consisting multiple solutions. By contrast, ETO involves some new terminologies, including problem, task, and solution. To be clearer, some related definitions are first given as follows:

- Problem: The problem refers to the ETO problems. Particularly, in the context of ESTO, the problem is the sequential transfer optimization problem (STOP). Similarly, in the context of EMT, the problem is the multitasking optimization problem (MTOP).

- Task: In the context of ETO, the task refers to the optimization task. In terms of ESTO, the STOP contains multiple source tasks that have been previously solved and one target task being optimized. By contrast, in terms of EMT, the MTOP consists of multiple tasks that must be optimized simultaneously.

- Solution: The solution refers to the solution of the task that is obtained during the evolutionary search process. In terms of ESTO, the source solution is the solution of the source task in its search space, while the target solution is the solution of the target task in its search space.

1) Evolutionary Sequential Transfer Optimization (ESTO)

ESTO aims to effectively utilize the search experience of previously solved tasks (called source tasks) to accelerate the optimization of a new task (called target task) [47]. When tackling the new task, i.e., T_K, there are some tasks that have already been solved in the past, i.e., T_1, T_2, $\cdots$, T_{K-1}. Note that T_K is called target task,

while T_1, T_2, $\cdots$, T_{K-1} are called source tasks. Here, each task can be an m-objective optimization problem, which can be formulated by

$$\min_{\mathbf{x}\in\Omega} \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})]^{\mathrm{T}} \tag{2.1}$$

where $\mathbf{x} = \{x_1, \ldots, x_d\}$ is a d-dimensional variable vector in the search space Ω and $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ are m objective functions. Eq. (2.1) can be a single-objective optimization problem (SOP) when m = 1 or a multiobjective optimization problem (MOP) when $m \geq$ 2. Thus, the sequential transfer optimization problem (STOP) can be defined by

$$\min_{\mathbf{x}\in\Omega} \ [\mathbf{F}(\mathbf{x}) \,|\mathcal{M}]\,, \tag{2.2}$$

where $\mathcal{M}$ is the knowledge base with the search experience of source tasks, including heuristic algorithms [38], [39], [40], configured parameters [41], [42], [43], and evaluated solutions [44], [45], [46], [47]. In particular, in solution-based ESTO, $\mathcal{M}$ is formed by collecting the evaluated solutions of each source task during the evolutionary search process, which can be represented by

$$\mathcal{M} = \{T_i \,|i = 1, 2, \ldots, K\}\,, \tag{2.3}$$

where $T_i = \left\{\mathbf{P}_{\mathrm{g}}^i, \mathbf{F}_{\mathrm{g}}^i \mid \mathrm{g} = 1, \ldots, \mathrm{g}_{max}\right\}$ is the set of populations of the i-th source task ($\mathbf{P}_{\mathrm{g}}^i$ and $\mathbf{F}_{\mathrm{g}}^i$ are the solutions and their objective values at the g-th generation, respectively).

2) Evolutionary Multitasking (EMT)

Evolutionary multitasking refers to evolutionary multitasking optimization, which can optimize multiple different tasks concurrently [51]. Regarding each optimization

problem as one task, the multitasking optimization problem (MTOP) with K tasks ($K \geq 2$) can be formulated as follows:

$$\min_{\mathbf{x}^i \in \Omega^i} \mathbf{F}_i(\mathbf{x}^i), i = \{1, 2, \ldots, K\}, \tag{2.4}$$

where $\mathbf{F}_i(\cdot)$ represents the function formulation of the i-th task and $\mathbf{x}^i$ is a solution in its search space (domain) Ω^i. In EMT, useful search experiences can be shared or transferred across different task domains through knowledge transfer, which can help to obtain better optimization performance and efficiency for solving MTOPs.

To achieve efficient EMT, multifactorial optimization (MFO) has been proposed to address K tasks at the same time [27]. In MFO, every task has a skill factor, which can affect the evolutionary search process. Here, two key concepts are given as follows:

Definition 1 (Skill Factor): The skill factor τ_i of each solution $\mathbf{x}_i$ indicates the task to which $\mathbf{x}_i$ belongs.

Definition 2 (Scalar Fitness): The scalar fitness of $\mathbf{x}_i$ is computed by $\varphi_i = 1/r_{\tau_i}^i$, where r_i is the index of $\mathbf{x}_i$ in a list sorted in ascending order with respect to the objective values of all solutions evaluated on the task τ_i.

Based on the above concepts, MFEA has been proposed as a realization of the MFO paradigm [27]. Given an MTOP with K tasks, the dimensionality of the search space of each task is given by D_1, D_2, $\cdots$, D_K, respectively. In MFEA [27], the solutions for all tasks are first encoded in a unified search space $Y \in [0, 1]^{D_{max}}$, where $D_{max} = max\{D_1, D_2, \ldots, D_K\}$. Hence, the solutions of all tasks have the same dimension. The solutions with the same skill factor can be regarded as the population of their respective task. Moreover, the random mating probability (rmp) allows to per-

form crossover operator on the solutions of different tasks. By introducing two key components, namely assortative mating and selective imitation, knowledge transfer can take place among different tasks [27].

2.2 What to Transfer

Regarding what to transfer, there are two crucial steps: identifying the form of knowledge and selecting the most promising knowledge for transfer. The form of knowledge can typically be classified into three main types, such as heuristic algorithms [38], [39], [40], configured parameters [41], [42], [43], and evaluated solutions [44], [45], [46], [47]. Due to the ease of use, transferring knowledge in the form of solutions has attracted increasing attention in existing ETO studies [37]. Once the type of knowledge is identified to be in the form of solutions, deciding what to transfer is simplified to the issue of deciding which one solution among all available candidates for transfer. To select the most promising solution for knowledge transfer, various measurement methods have been proposed to estimate the usefulness of solutions from source tasks for the target task, where source and target tasks are a class of continuous optimization problems with the same problem dimensionality. Note that the real-valued encoding is employed to represent the solutions of source and target tasks. The technical details of existing measurement methods are summarized in Tab. 2.1.

Table 2.1: List of existing measurement methods for solution usefulness.

Measurement Method		Formula
Distance Metric	Euclidean Distance (ED)[a]	$\mathrm{ED}(\mathbf{X}_s,\ \mathbf{X}_t) = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_t\|_2$, where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i,\ \mathbf{X} = \{\mathbf{x}_1,\ ...,\ \mathbf{x}_N\}$
	Wasserstein Distance (WD)[a]	$\mathrm{WD}(\mathbf{X}_s,\ \mathbf{X}_t) = \sqrt{\|\boldsymbol{\mu}_s - \boldsymbol{\mu}_t\|_2^2 + \|\boldsymbol{\sigma}_s - \boldsymbol{\sigma}_t\|_2^2}$, where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i,\ \boldsymbol{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^2,\ \mathbf{X} = \{\mathbf{x}_1,\ ...,\ \mathbf{x}_N\}$
	Kullback–Leibler Divergence (KLD)[a,b]	$\mathrm{KLD}(\mathbf{X}_s,\ \mathbf{X}_t) = \frac{1}{2}\left(\mathrm{tr}(\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Sigma}_s) + (\boldsymbol{\mu}_t - \boldsymbol{\mu}_s)^T\Sigma_t^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_s) - D + \ln(\det(\boldsymbol{\Sigma}_t)/\det(\boldsymbol{\Sigma}_s))\right)$, where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i,\ \boldsymbol{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T,\ \mathbf{X} = \{\mathbf{x}_1,\ ...,\ \mathbf{x}_N\}$
	Maximum Mean Discrepancy (MMD)[a,c]	$\mathrm{MMD}(\mathbf{X}_s,\ \mathbf{X}_t) = \frac{1}{n_s(n_s-1)}\sum_{i\neq j}^{n_s}k(\mathbf{x}_i^s,\ \mathbf{x}_j^s) + \frac{1}{n_t(n_t-1)}\sum_{i\neq j}^{n_t}k(\mathbf{x}_i^t,\ \mathbf{x}_j^t) - \frac{2}{n_s n_t}\sum_{i,j=1}^{n_s,n_t}k(\mathbf{x}_i^s,\ \mathbf{x}_j^t)$, where $\mathbf{X}_s = \{\mathbf{x}_1^s,\ ...,\ \mathbf{x}_{n_s}^s\},\ \mathbf{X}_t = \{\mathbf{x}_1^t,\ ...,\ \mathbf{x}_{n_t}^t\},\ k(\mathbf{x},\ \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|_2^2/(2\sigma^2))$
Machine Learning	Anomaly Detection (AD)[d]	$C(\mathbf{x}) = \begin{cases}\text{anomaly}, & \text{if } p(\mathbf{x}\mid\boldsymbol{\mu},\ \boldsymbol{\Sigma}) < \varepsilon\\ \text{normal}, & \text{if } p(\mathbf{x}\mid\boldsymbol{\mu},\ \boldsymbol{\Sigma}) \geq \varepsilon\end{cases},\ p(\mathbf{x}\mid\boldsymbol{\mu},\ \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d\mid\boldsymbol{\Sigma}\mid}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$, where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^t,\ \boldsymbol{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i^t - \boldsymbol{\mu})(\mathbf{x}_i^t - \boldsymbol{\mu})^T,\ \mathbf{X}_t = \{\mathbf{x}_1^t,\ ...,\ \mathbf{x}_N^t\}$
	Incremental Naive Bayes (INB)[d,e]	$C(\mathbf{x}) = \arg\max_{j\in\{1,2\}}\left\{\bar{P}(C_j)\prod_{k=1}^{d}\bar{P}(x_k\mid C_j)\right\},\ \bar{P}(x_k\mid C_j) = \frac{1}{\sqrt{2\pi\bar{\sigma}_j^2}}\exp\left(-\frac{(x_k-\bar{\mu}_j)^2}{2\bar{\sigma}_j^2}\right)$, $\bar{P}(C_j) = \frac{n\cdot P(C_j) + m\cdot P'(C_j)}{n+m}$, where $\bar{\mu}_j = \frac{n\cdot\mu_j + m\cdot\mu'_j}{n+m},\ \bar{\sigma}_j^2 = \frac{n\cdot\sigma_j^2 + m\cdot\sigma'^2_j}{n+m}$

Note that the source and target populations are processed to make their solutions have equal dimensionality.

[a] To explain the notations in distance metrics: $\mathbf{X}_s$, $\mathbf{X}_t$ (the source/target populations); $\boldsymbol{\mu}_s$, $\boldsymbol{\mu}_t$ (the mean vectors of the source/target populations); $\boldsymbol{\sigma}_s^2$, $\boldsymbol{\sigma}_t^2$ (the variance vectors of the source/target populations); $\boldsymbol{\Sigma}_s$, $\boldsymbol{\Sigma}_t$ (the covariance matrices of the source/target populations).

[b] In KLD, $\mathrm{tr}(\cdot)$ and $\det(\cdot)$ are the trace and determinant of a matrix, respectively. In addition, D is the smaller dimension of the source and target populations.

[c] In MMD, σ is the width parameter in the Gaussian kernel function $k(\mathbf{x},\ \mathbf{x}')$.

[d] To explain the notations in ML models: $\mathbf{x} = \{x_1,\ ...,\ x_d\}$ is a d-dimensional test sample (i.e., the source solution) to be classified.

[e] In INB, $P(C_j)$ and $P'(C_j)$ are the priori probability of the class label C_j in the past training data with n samples and the new training data with m samples, respectively. Here, μ_j and σ_j^2 are the mean and variance of the samples belonging to C_j in the past training data, respectively. In addition, μ'_j and σ'^2_j are the mean and variance of the samples belonging to C_j in the new training data, respectively.

2.2.1 Distance Metric-Based Methods

Various measurement methods of solution usefulness have been developed to select useful solutions via distance metrics, aiming to achieve better optimization performance on the target task. Particularly, the source solution is useful when it has better quality than all solutions in the current population of the target task. Otherwise, it is useless for the target task. Generally, when the source task is more similar to the target task in terms of the population distribution, the source solution is more likely to be useful to accelerate the optimization process. Existing distance metrics measure the similarity between the populations of source and target tasks to approximate the solution usefulness of the source task to the target task. For example, in [44] and [45], Euclidean distance (ED) is used to compute the distance between source and target solutions, aiming to appropriate the usefulness of the source solution for the target task. The closest source solution is considered to have the highest usefulness, and it is injected into the target population. Similarly, ED is also employed to measure the distance between solutions in EMT/ET [52]. The source solutions close to the positive-transferred solution are collected as the neighbours, which are selected for transfer as they are supposed to have higher usefulness than others. In addition, in MSSTO [53], Wasserstein distance (WD) is used to compute the distance between source and target populations and then three different selection strategies are used to reasonably select useful source populations for transfer, respectively. In addition, in MaTDE [54] and EMaTO-MKT [55], the distance between source and target populations is computed by Kullback-Leibler divergence (KLD) and the maximum mean discrepancy (MMD), respectively, aiming to estimate the usefulness of source solutions during the evolutionary search process. However, these distance metrics are unable to determine whether the solutions of source tasks are useful for accelerating the optimization of

the target task. Particularly, using distance metrics can compute the numeric values that are no less than zero. Due to the lack of the critical threshold, it is difficult to distinguish useful solutions from useless solutions. Thus, selecting useful source solutions by comparing their distance values may result in ineffective solution transfer, especially when all candidate source tasks are significantly dissimilar to the target task [37].

2.2.2 Machine Learning Model-Based Methods

Moreover, several measurement methods have also been designed by using ML models to measure the solution usefulness of the source tasks. For example, in MTEA-AD [56], an anomaly detection model is built based on the target population, which aims to divide the source solutions into outliers and nonoutliers. Similarly, in EMTIL [57], an incremental Naive Bayes classifier is trained based on the transferred solutions, which divides the source solutions into two different classes. The source solutions in the positive class are considered to have higher usefulness for the target task. However, as the above binary classifiers are unable to accurately quantify the degree of the usefulness of source solutions for the target task, the most useful one could not be accurately selected for transfer, which may degrade the effectiveness of solution transfer [58].

2.3 How to Transfer

Regarding how to transfer, it focuses on the methodology design, which aims to implement effective knowledge transfer across various complex problems. In existing

ETO studies, these transfer methods can be classified into two categories: 1) implicit transfer methods and 2) explicit transfer methods.

2.3.1 Implicit Transfer Methods

To perform implicit knowledge transfer, all solutions are first encoded in a unified search space. After that, knowledge transfer implicitly occurs across tasks by employing genetic operators to achieve the exchange of genetic materials of candidate solutions possessing different skill factors. Over the years, various genetic operators, such as crossover and mutation, have been employed to achieve implicit knowledge transfer across tasks. For example, in MFEA and most of its variants [27], [72], [73], [74], knowledge transfer was achieved by performing the simulated binary crossover (SBX) to exchange the genetic materials of two solutions with different skill factors. In [59], differential evolution (DE) was employed to achieve implicit knowledge transfer. To utilize the search biases of different crossovers, in MFEA-AKT [60], multiple crossovers were adaptively selected for conducting knowledge transfer. Additionally, several popular swarm intelligence algorithms, including particle swarm optimization [61], [62], [63], artificial bee colony [64], and fireworks algorithm [65], have been adapted for achieving implicit knowledge transfer. However, implicit knowledge transfer depends on the explicit similarity of two tasks in terms of their population distributions or fitness landscapes, thereby showing poor performance when solving distinct or even unrelated tasks.

2.3.2 Explicit Transfer Methods

In contrast to implicit knowledge transfer using a population for all tasks, explicit

knowledge transfer configures each task with a separate population, allowing multiple solution encoding schemes for use when solving multiple tasks simultaneously. Thus, knowledge transfer can be conducted across different populations in an explicit manner [48]. In this way, existing evolutionary search mechanisms with unique search biases can also be flexibly configured for each task, effectively enhancing the optimization efficacy and performance on different types of problems [48]. The straightforward way for achieving explicit knowledge transfer across tasks is to directly inject solutions of one population into another population [44], [45]. However, the two tasks may have obvious differences in their dimensionalities, variable ranges, global optima, and fitness landscapes, which bring difficulties for performing explicit knowledge transfer across distinct problem domains [48]. In recent years, several domain adaptation methods have been proposed to enhance knowledge transferability between distinct tasks by building suitable transformation from the source task to the target task. For example, in LDA-MFEA [66], based on the ordinal rank correlation of fitness values, a linear transformation method is used to make the search spaces of source and target tasks highly correlated. In addition, a denoising AE method [48] is used to build the source-target mapping by minimizing the reconstruction loss of the corrupted input on the source task, which can maintain the superiority of transferred solutions on the target task. An affine transformation (AT) method was designed in AT-MFEA [67] to build a superior intertask mapping between source and target tasks, which considers the topological consistency in decision space and the rank correlation in objective space. To capture the nonlinearity between different tasks, a kernelized AE (KAE) [68] was further designed to learn the mapping in a reproduced kernel Hilbert space. Moreover, to learn well-aligned solution representations, two-layer transformations are learnt in continuous spaces via a two-layer feedforward neural network [69], while variable transformation is learnt in the dimension-reduced sub-

spaces of tasks via subspace alignment methods [70], [71].

2.4 When to Transfer

Regarding when to transfer, it aims to identify the optimal time or extent to conduct knowledge transfer. In existing ETO studies, the proposed approaches to address when to transfer can be categorized as follows: 1) fixed parameter-based methods and 2) adaptive parameter-based methods.

2.4.1 Fixed Parameter-Based Methods

The straightforward way to determine when to transfer is to execute knowledge transfer periodically with a fixed generation interval along the evolutionary search process. For example, in [46], [48], [68], the generation interval for triggering knowledge transfer was set to 10. Similarly, in SGDE [75], the generation interval was 20. Besides, in [76], the generation interval for triggering knowledge transfer was set to 50. Additionally, in the basic framework of MFEA [27], the random mating probability, i.e., rmp, is employed to determine the extent of knowledge transfer. Generally, when two parents have different skill factors and a random number is less than rmp, knowledge transfer happens between two tasks by performing a crossover operator on the two parents to exchange their genetic materials. For example, in the original MFEA [27] and most of its variants, rmp is set as a constant of 0.3 to control the extent of knowledge transfer across tasks. However, the above methods with fixed transfer parameters may potentially diminish effective knowledge transfer or even

cause negative transfer when solving some unrelated tasks [49], [50].

2.4.2 Adaptive Parameter-Based Methods

In addition, several adaptive parameter-based methods have been proposed to determine the extent of knowledge transfer by dynamically modifying the values of the transfer parameter. Specifically, in MFEARR [77], the survival rate of offspring generated by triggering knowledge transfer was computed to determine the extent of knowledge transfer. Similarly, in [78], the transfer parameter associated with each task is computed as follows:

$$rmp_i = \frac{S_{\tau_i, NF=0}}{NP_i}, \tag{2.5}$$

where NP_i and $S_{\tau_i, NF=0}$ are the numbers of all solutions and nondominated solutions of the task T_i, respectively. Similarly, in [79], the extent of knowledge transfer was dynamically adjusted by employing two additional parameters, i.e., Δ_{inc} and Δ_{dec}, which is computed as follows:

$$rmp_{i,j} = \begin{cases} \min\left(1.0, \frac{rmp_{i,j}}{\Delta_{inc}}\right), & \text{if } \mathbf{p}' \text{ is better than } \mathbf{p} \\ \max\left(0.1, rmp_{i,j} \times \Delta_{dec}\right), & \text{otherwise} \end{cases}, \tag{2.6}$$

where $\mathbf{p}'$ is the offspring and $\mathbf{p}$ is its immediate parent. Besides, in [62], the learning parameter, i.e., rlp, is used to determine the transfer extent. Particularly, the current rlp remains the same and is collected into the list when there is at least a better solution than the current best solutions of all tasks. Otherwise, rlp is updated by

$$rlp = rlp' + \delta\mathcal{N}(0,1), \tag{2.7}$$

where δ is 0.1, $\mathcal{N}(0,1)$ is a normalized Gaussian distribution, and rlp' is randomly selected from the candidate list. Moreover, in [80], the transfer probability for the task T_i is calculated by

$$rmp_i = \frac{R_i^o}{R_i^o + R_i^s}, \tag{2.8}$$

where R_i^s and R_i^o are the improvement ratios obtained by the offspring of T_i and solutions of other tasks, respectively. In [81], given an MTOP with K tasks, a symmetric $K \times K$ matrix, i.e., W, is employed to record the number of successful transferred solutions across tasks. After that, the transfer probability between any two tasks, is computed as follows:

$$rmp_{i,j} = \frac{W_{i,j}}{\sum_{k=1}^{K} W_{i,k}}. \tag{2.9}$$

In addition, to reduce the threat of negative transfer, in MFEA-II [49] and MO-MFEA-II [50], the probability models were built to estimate the transfer extent by capturing the similarities among tasks. The transfer extent can be adapted along the evolutionary search process by employing the abovementioned methods. However, they do not take into consideration the issue of information inaccuracy due to the uncertainty and randomness of the evolutionary mechanisms. Lacking a mechanism for effectively processing inaccurate and fuzzy information may lead to unreliable or wrong decisions in dynamically adjusting the transfer parameter, which will diminish the effectiveness of knowledge transfer.

Chapter 3

Fuzzy Classifier-Assisted Method for Adaptively Deciding What to Transfer in ESTO

3.1 Introduction

To achieve effective solution transfer, some case-based methods have been proposed by injecting the target population with source solutions stored as cases in the case base [44], [45]. However, injecting useless solutions does not accelerate the optimization process of the target task. Hence, several measurement methods based on distance metrics were employed to estimate the solution usefulness of source tasks [82], [53], [54], [55]. Thus, the useful source solutions can be selected as promising cases,

which are injected into the target population to accelerate its optimization process. Particularly, the Euclidean distance (ED) was used in [82] to calculate the distance between the source and target populations. In addition, the Wasserstein distance (WD) [53] was used to measure the distance between the population distributions of the source and target tasks. Moreover, the Kullback-Leibler divergence (KLD) [54] was employed to measure the distance of the source and target populations based on their Gaussian representations, while the maximum mean discrepancy (MMD) [55] was used to estimate their distance in the reproducing kernel Hilbert space. In fact, the solution usefulness of a source task relies on its explicit similarity to the target task. In situations where all candidate source tasks are distinctly dissimilar to the target task, their optimized solutions would be useless for the target task. However, using distance metrics cannot determine whether the solutions of source tasks are useful for the optimization acceleration of the target task, which may result in ineffective solution transfer [37].

In addition, some studies have been conducted to address the aforementioned issue by leveraging machine learning (ML) techniques [56], [57]. Due to the ability to learn from previous data and make predictions on unseen data [83], [84], ML has shown some advantages in distinguishing the usefulness and uselessness of solutions. For example, in MTEA-AD [56], an anomaly detection model based on the multivariate Gaussian distribution was built on the target population, which divides source solutions into outliers (useful solutions) and nonoutliers (useless solutions) to the target task. Similarly, in EMTIL [57], an incremental Bayes classifier was trained based on the transferred solutions, which divides source solutions into positive and negative classes. The solutions in the positive class are considered to be useful for the optimization of the target task, while those in the negative class are useless. However, these

classifiers are unable to quantify the degree of solution usefulness, which may potentially undermine the effectiveness of solution transfer [58].

To tackle the drawbacks of existing methods, this study focuses on two critical issues in measuring the solution usefulness of source tasks: 1) whether a useful source task exists and 2) the degree of its usefulness. Compared to traditional nonfuzzy classifiers that typically predict a class for each sample of interest, fuzzy classifiers can predict a membership degree of each sample to each class, which have attracted increasing attention [85], [86], [87]. As the evolutionary search process goes on, the boundary of high-quality solutions and low-quality solutions will become unclear and fuzzy. Recently, some efforts have been made to use fuzzy classifiers to predict the quality of solutions in environmental selection, aiming to accurately select high-quality solutions for the next generation [88], [89], [90], [91], [92]. Inspired by the studies above, this chapter proposes a fuzzy classifier-assisted solution transfer method to select useful source solutions for ESTO. First, given the STOP with the target task and a knowledge base including the evaluated solutions of candidate source tasks, the training data are constructed by sampling multiple task pairs, each of which has two different source tasks. Specifically, for each task pair, their solutions are used to compute the differences of their population distributions, which serve as the feature vector of one training sample. Additionally, the training sample is roughly assigned one positive or negative label by evaluating the current best solution of the first task on the second task. If the evaluated objective value is better than the current best objective value of the second task, the training sample is assigned a positive label. Otherwise, it is assigned a negative label. After that, the fuzzy classifier is built using these training data, which can be used to measure the solution usefulness of each source task by the returned class label and its membership degree to that class on its associated test sample. Here, the test sample is generated based on the solu-

tions of its corresponding source and target tasks. In this way, the source task with the maximal membership degree in the positive class is selected to provide its optimized solution for transfer if it exists. Otherwise, solution transfer will not happen. Moreover, the test sample of the selected source task is labelled based on the effectiveness of solution transfer, which is used to update the training data. The main contributions of this study are summarized as follows.

1) This study proposes a fuzzy classifier-assisted solution transfer method (FCM) for ESTO. By constructing the training data, the fuzzy classifier is built to measure the solution usefulness of candidate source tasks. In this way, useful source solutions can be effectively selected to accelerate the evolutionary search of the target task.

2) This study presents the implementation of an ESTO algorithm with the proposed method (ESTOA-FCM). The experimental results on two benchmark suites and one practical case show the competitive performance of ESTOA-FCM when compared with existing ESTO algorithms.

3.2 Background and Motivation

3.2.1 Fuzzy Classifier

Fuzzy classifiers are a type of ML algorithms developed based on fuzzy set theory [85], allowing for more flexible and nuanced classification decisions compared to traditional nonfuzzy classifiers [86], [87]. Instead of assigning a single class label to a sample of interest, a fuzzy classifier assigns a membership degree to each class label, which reflects the degree of the sample belonging to that class.

Algorithm 3.1 Fuzzy K-Nearest Neighbour (FKNN)

Input: $\mathcal{D} = \{< \boldsymbol{s}_i, l_i >\}_{i=1}^{n}$: the training data,
$\boldsymbol{v}$: a new sample to be classified,
K: the number of nearest neighbours.

Output: y, m

```
1  Set Q to an empty set
2  for i = 1 to n
3      dist(v, s_i) ← Compute the Euclidean distance from v to s_i by Eq. (3.1)
4      if i ≤ K
5          Q = Q∪{s_i}
6      else
7          s_max ← Find the farthest sample in Q by Eq. (3.2)
8          if dist(v, s_i) < dist(v, s_max)
9              Q = Q\{s_max}, Q = Q∪{s_i}
10         end
11     end
12 end
13 for each class j in D
14     u_j ← Compute membership degree of v in the j-th class by Eq. (3.3) and Eq. (3.4)
15 end
16 [y, m] ← Get the label of v and membership degree by Eq. (3.5) and Eq. (3.6)
17 return y, m
```

In this study, for simplicity, a fuzzy K-nearest neighbour classifier (FKNN) is employed as the fuzzy classifier. Its pseudocode is given in Algorithm 3.1 with the inputs: $\mathcal{D} = \{< \boldsymbol{s}_i, l_i >\}_{i=1}^{n}$ (the training data consisting of n labelled samples where each sample $\boldsymbol{s}_i$ is assigned a label $l_i \in \{0, 1\}$), $\boldsymbol{v}$ (a new sample with an unknown label), and K (the number of the nearest neighbours of $\boldsymbol{v}$). First, $\mathbf{Q}$ is set to an empty set in Line 1. Then, in Lines 2-12, the K nearest neighbours are collected into $\mathbf{Q}$ through comparing the Euclidean distances between all the samples and $\boldsymbol{v}$. Specifically, the Euclidean distance between the i-th sample $\boldsymbol{s}_i$ and $\boldsymbol{v}$ can be calculated by

$$dist(\boldsymbol{v}, \boldsymbol{s}_i) = ||\boldsymbol{v} - \boldsymbol{s}_i||_2 . \tag{3.1}$$

Next, in Lines 4-11, the first K samples are directly added into $\mathbf{Q}$, while each of the subsequent samples will be checked to determine whether it is closer to $\boldsymbol{v}$ than the farthest sample $\boldsymbol{s}_{max}$. Here, $\boldsymbol{s}_{max}$ is found by

$$\boldsymbol{s}_{max} = \arg\max_{\boldsymbol{s}_i \in \mathbf{Q}} \{||\boldsymbol{v} - \boldsymbol{s}_i||_2\}, \tag{3.2}$$

If $\boldsymbol{s}_i$ is closer to $\boldsymbol{v}$ than $\boldsymbol{s}_{max}$, $\boldsymbol{s}_{max}$ is removed from $\mathbf{Q}$ and then $\boldsymbol{s}_i$ will be added into $\mathbf{Q}$. After performing the above procedures in Lines 2-12, the K nearest neighbours are found and then added into $\mathbf{Q}$. Next, as shown in Lines 13-15, the membership degree of $\boldsymbol{v}$ in the j-th class (i.e., u_j) is computed by

$$u_j = \frac{\sum_{\boldsymbol{s}_i \in \mathbf{Q}} u_{ij}\left(1/||\boldsymbol{v} - \boldsymbol{s}_i||_2^2\right)}{\sum_{\boldsymbol{s}_i \in \mathbf{Q}}\left(1/||\boldsymbol{v} - \boldsymbol{s}_i||_2^2\right)}, \tag{3.3}$$

where u_{ij} is the membership degree of $\boldsymbol{s}_i$ in the j-th class ($j \in \{1,2\}$ in the binary classification). Here, the k-nearest neighbour rule is used in the membership assignment technique [38]. Thus, u_{ij} is computed by

$$u_{ij} = \begin{cases} 0.51 + \left(\frac{n_j}{k}\right) \times 0.49, & \text{if the class label of } \boldsymbol{s}_i \text{ is } j \\ \frac{n_j}{k} \times 0.49, & \text{Otherwise} \end{cases}, \tag{3.4}$$

where k is the number of neighbours and n_j is the number of the neighbours of $\boldsymbol{s}_i$ belonging to the j-th class. As suggested in [38], k is set to 3. Then, in Line 16, the label of $\boldsymbol{v}$ and its membership degree, i.e., y and m, are respectively identified as follows:

$$y = \begin{cases} 1, & \text{if } u_1 \geq u_2 \\ 0, & \text{otherwise} \end{cases}, \tag{3.5}$$

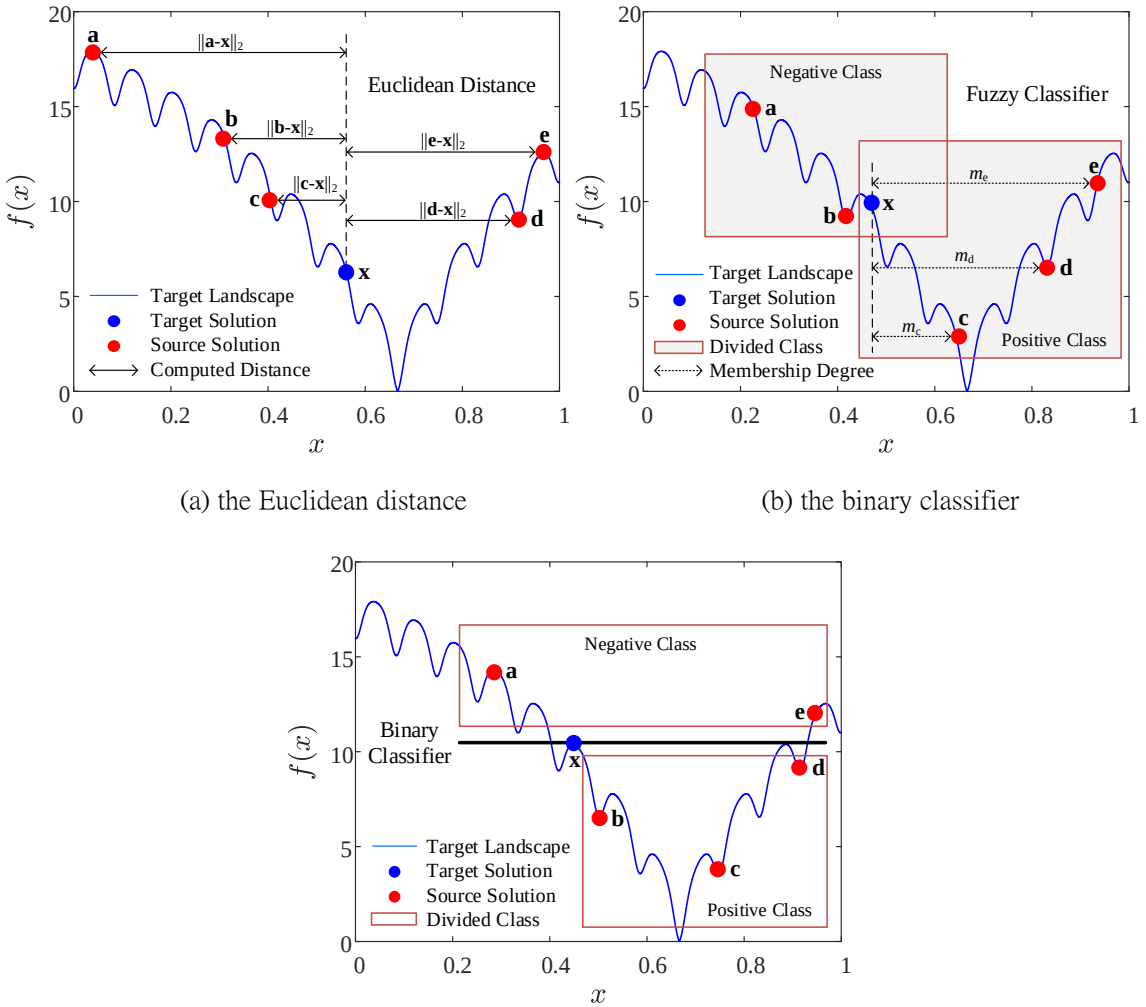
$$m = \max\{u_1, u_2\}, \tag{3.6}$$

where u_1 and u_2 are the membership degrees of $\boldsymbol{v}$ in the 1-th class and 2-th class, respectively. Finally, y and m are returned in Line 17, indicating the predicted class of $\boldsymbol{v}$ and the extent to which $\boldsymbol{v}$ is considered to belong to that class, respectively.

3.2.2 Motivation

Based on the studies of existing measurement methods of solution usefulness, it can be concluded that using the distance metrics or classical ML models is rough and inaccurate in selecting useful source solutions based on the usefulness of solutions. Generally, the usefulness of a source solution is measured by its quality on the target task, i.e., $\mathbb{U}(\mathbf{x}^s) = \max\{f^t(\mathbf{x}) - f^t(\mathbf{x}^s), 0\}$ where $f^t(\cdot)$ is the target function, $\mathbf{x}$ is the current best solution of $f^t(\cdot)$, and $\mathbf{x}^s$ is the source solution. To elaborate further, the comparisons of measuring the usefulness of the source solutions with the Euclidean distance and the binary classifier are given in Fig. 3.1(a) and (b). Here, the target task is a shifted Ackley function with one decision variable. Note that it can be extended to any function with multiple decision variables. The blue circle is the current best solution of $f^t(\cdot)$, which is denoted by $\mathbf{x}$. The red circles represent source solutions of five source tasks, which are denoted by $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$, and $\mathbf{e}$. The usefulness of each source solution is quantitatively measured by computing its Euclidean distance to $\mathbf{x}$. In Fig. 3.1(a), the distances of those source solutions to $\mathbf{x}$ can be ranked in ascending order, i.e., $||\mathbf{c}-\mathbf{x}||_2 < ||\mathbf{b}-\mathbf{x}||_2 < ||\mathbf{d}-\mathbf{x}||_2 < ||\mathbf{e}-\mathbf{x}||_2 < ||\mathbf{a}-\mathbf{x}||_2$. In this way, $\mathbf{c}$ is selected as the most useful solution for transfer because it has the smallest distance value. However, as the objective value of $\mathbf{c}$ is worse than that of $\mathbf{x}$, i.e., $f^t(\mathbf{c}) > f^t(\mathbf{x})$, transferring $\mathbf{c}$ to the target task will not contribute to accelerating the optimization of the target task. In this sense, $\mathbf{c}$ is useless for the

target task as $\mathbb{U}(\mathbf{c}) = 0$. In fact, using distance metric can select a relatively good one among source solutions by comparing their measured distance values, while it fails to determine whether the selected source solution is useful for the target task. Thus, relying on distance metric to measure the usefulness of the solution may result in inefficient solution transfer when the objective values of all the candidate source solutions are worse than the current best solution of the target task.



(a) the Euclidean distance

(b) the binary classifier

(c) the fuzzy classifier

Figure 3.1: Comparisons of different methods for measuring usefulness of solutions.

Additionally, several ML models, i.e., the binary classifiers, can be employed to measure the usefulness of the source solutions by dividing them into two categories. As shown in Fig. 3.1(b), with a binary classifier, the source solutions with better objective values (i.e., **b**, **c**, and **d**) are in the positive class, which will be selected as candidate solutions for transfer. Thus, using the binary classifier can avoid inefficient solution transfer to some extent because the source solutions with worse objective values (i.e., **a** and **e**) in the negative class are not considered. In fact, **c** is the most useful source solution among **b**, **c**, and **d** in the positive class, as their usefulness can be ranked in descending order, i.e., $\mathbb{U}(\mathbf{c}) > \mathbb{U}(\mathbf{b}) > \mathbb{U}(\mathbf{d})$. However, the binary classifier does not quantitatively measure the usefulness of the source solutions in the positive class, which will cause the case that the most useful solution could not be accurately selected for transfer.

In summary, the Euclidean distance and classical binary classifier are flawed in measuring the usefulness of the source solutions for the target task, which may potentially undermine the effectiveness of solution transfer in optimizing the target task. Thus, a more reliable measurement of the solution usefulness should be able to tackle two issues: 1) whether a useful source solution exists and 2) the degree of its usefulness. In the community of classification, the fuzzy classifier not only predicts the class label, but also predict a membership degree of each new pattern to each class. Thus, the fuzzy classifier is more suitable to measure the usefulness of the source solutions than other classical binary classifiers as employing it can easily address the above issues in measuring the usefulness of solutions. As shown in Fig. 3.1(c), with a fuzzy classifier, **a** and **b** are in the negative class, while **c**, **d**, and **e** are in the positive class. Thus, using the fuzzy classifier can identify whether a useful source solution exists by checking whether there is the solution in the positive class. Fur-

thermore, the membership degrees of **c**, **d**, and **e** are computed to indicate their degrees belonging to the positive class, which are ranked in descending order, i.e., $m_{\mathbf{c}} > m_{\mathbf{d}} > m_{\mathbf{e}}$. In this way, **c** with the maximal membership degree is selected as the most useful solution for transfer. Thus, to achieve more effective solution transfer for ESTO, this study is motivated to develop a fuzzy classifier-assisted solution transfer method for selecting useful source solutions.

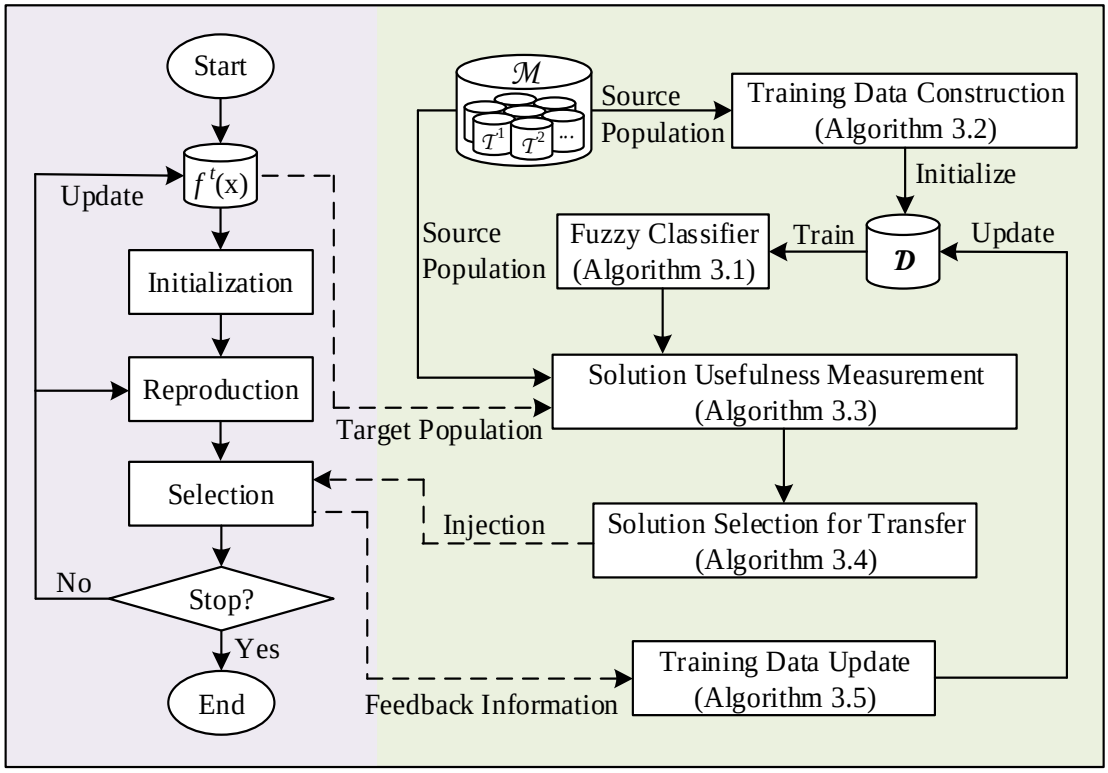


Figure 3.2: Flowchart of the proposed method.

3.3 Methodology

This section presents the details of the proposed method and the implementation of ESTOA-FCM. To be clear, the flowchart of ESTOA-FCM is illustrated in Fig. 3.2. In

particular, Algorithm 3.2 is to construct the training data $\mathcal{D}$ by using the evaluated solutions of source tasks from the knowledge base $\mathcal{M}$. Then, the fuzzy classifier (Algorithm 3.1) is built based on $\mathcal{D}$, which is used in Algorithm 3.3 to measure the solution usefulness of each source task by the returned class label and the membership degree to that class. Hence, the most useful source solution can be selected for knowledge transfer by performing Algorithm 3.4. Besides, Algorithm 3.5 is to update $\mathcal{D}$ by replacing an old training sample with the new one obtained at the current generation. Here, the feature and label spaces of the training data are first introduced in subsection 3.3.1. Then, the following subsections provide the detailed descriptions of Algorithm 3.2 to Algorithm 3.5. Finally, the details of ESTOA-FCM are given in subsection 3.3.6.

3.3.1 Feature Space and Label Space Definition

1) Definition of Feature Space

As the evolutionary search process continues, the evaluated solutions from the initial population to the current population can be sequentially collected to form the evolutionary path, which is defined by

$$E = \{\mathbf{P}_1, \mathbf{P}_1, \dots, \mathbf{P}_G\}, \tag{3.7}$$

where $\mathbf{P_g}$ is the population at generation $\mathbf{g}$ ($\mathbf{g} = \{1, \dots, G\}$, G is the current generation). In view of the high computational efficiency and the retained full information of a progressional representation developed in [67], it is employed to estimate the population distribution of an individual optimization problem, which is defined by

$$p_G(\mathbf{x}) = \begin{cases} \mathfrak{T}(\mathbf{x}; \mathbf{P}_1), & \text{if } G = 1 \\ \alpha p_{G-1}(\mathbf{x}) + (1-\alpha)\mathfrak{T}(\mathbf{x}; \mathbf{P}_{\mathrm{G}}), & \text{if } G > 1 \end{cases}, \tag{3.8}$$

where $\boldsymbol{\alpha}$ is a preference coefficient for determining the contributions of previous populations and newly updated populations and $\mathfrak{T}$ is an operator that estimates the probability distribution of a population. Moreover, the above recursive expression can be rewritten into the following form:

$$p_G(\mathbf{x}) = (1-\alpha)\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\mathfrak{T}(\mathbf{x}; \mathbf{P}_{\mathrm{g}})\right). \tag{3.9}$$

Due to very good mathematical properties of the independent multivariate Gaussian distribution [67], [93], it is employed to estimate the probability distribution of each population, i.e., $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ with the mean vector $\boldsymbol{\mu} = \{\mu_1, \dots, \mu_d\}$ and the variance vector $\boldsymbol{\sigma}^2 = \{\sigma_1^2, \dots, \sigma_d^2\}$ where μ_j and σ_j^2 are the mean and variance of the j-th dimension ($j = 1, \dots, d$), and d is the population dimension. Specifically, for the population $\mathbf{P}_{\mathrm{g}} = \{\mathbf{x}_1^{\mathrm{g}}, \mathbf{x}_2^{\mathrm{g}}, \dots, \mathbf{x}_N^{\mathrm{g}}\}$ at the g-th generation, its estimated distribution, i.e., $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{g}}, \boldsymbol{\sigma}_{\mathrm{g}}^2)$, can be calculated by

$$\begin{cases} \mu_{\mathrm{g}j} = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} x_{ij}^{\mathrm{g}} \\ \sigma_{\mathrm{g}j}^2 = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} (x_{ij}^{\mathrm{g}} - \mu_{\mathrm{g}j})^2 \end{cases}, \tag{3.10}$$

where N is the size of $\mathbf{P}_{\mathrm{g}}$ and x_{ij}^{g} is the value of the j-th dimension of the i-th solution $\mathbf{x}_i^{\mathrm{g}}$. With Eq. (3.9) and Eq. (3.10), we have

$$\begin{aligned} p_G(\mathbf{x}) &\sim (1-\alpha)\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\mathcal{N}\left(\boldsymbol{\mu}_{\mathrm{g}}, \boldsymbol{\sigma}_{\mathrm{g}}^2\right)\right) \\ &= \mathcal{N}\left((1-\alpha)\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\boldsymbol{\mu}_{\mathrm{g}}\right), (1-\alpha)\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\boldsymbol{\sigma}_{\mathrm{g}}^2\right)\right). \end{aligned} \tag{3.11}$$

It can be found that the progressional representation could be easily expressed by

$$p_G(\mathbf{x}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}^2), \tag{3.12}$$

where $\hat{\boldsymbol{\mu}} = \{\hat{\mu}_1, \dots, \hat{\mu}_d\}$ and $\hat{\boldsymbol{\sigma}}^2 = \{\hat{\sigma}_1^2, \dots, \hat{\sigma}_d^2\}$ are respectively calculated by

$$\begin{cases} \hat{\mu}_j = (1-\alpha)\dfrac{1}{N}\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\sum_{i=1}^{N} x_{ij}^{\mathrm{g}}\right) \\ \hat{\sigma}_j^2 = (1-\alpha)\dfrac{1}{N-1}\Sigma_{\mathrm{g}=1}^{G}\left(\alpha^{G-\mathrm{g}}\sum_{i=1}^{N} (x_{ij}^{\mathrm{g}} - \mu_{\mathrm{g}j})^2\right) \end{cases}. \tag{3.13}$$

As the feature vectors are expected to represent the similarity of the evolutionary search processes of two tasks, the distances between their probability distributions are employed to construct the feature space. In particular, given the populations of two tasks T_a and T_b, the solutions in the population with smaller dimension are first padded with zeros to make them have equal dimensionality. Then, their probability distributions can be estimated by building the progressional representations on the populations of T_a and T_b, respectively, which are denoted by $p^a(\mathbf{x}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_a, \hat{\boldsymbol{\sigma}}_a^2)$ and $p^b(\mathbf{x}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_b, \hat{\boldsymbol{\sigma}}_b^2)$. Finally, the feature vector $\boldsymbol{s}_i = \{s_{i1}, s_{i2}\}$ is to represent the differences between the parameters of $p^a(\mathbf{x})$ and $p^b(\mathbf{x})$, which is computed by

$$\begin{cases} s_{i1} = ||\hat{\boldsymbol{\mu}}_a - \hat{\boldsymbol{\mu}}_b||_2 \\ s_{i2} = ||\hat{\boldsymbol{\sigma}}_a^2 - \hat{\boldsymbol{\sigma}}_b^2||_2 \end{cases}, \tag{3.14}$$

where $||\hat{\boldsymbol{\mu}}_a - \hat{\boldsymbol{\mu}}_b||_2$ and $||\hat{\boldsymbol{\sigma}}_a^2 - \hat{\boldsymbol{\sigma}}_b^2||_2$ are the Euclidean distances.

2) Definition of Label Space

As the labels of feature vectors are expected to reflect the solution usefulness of source tasks for the target task, the label space is defined as {1, 0} where 1 (or 0) de-

notes a positive (or negative) sample. In particular, the label of $\boldsymbol{s_i}$ is identified by comparing the estimated objective value of the optimized solution of T_a to the current best objective value of T_b, which is given by

$$l_i = \begin{cases} 1, & \text{if } \boldsymbol{F}^b(\mathbf{x}) < \boldsymbol{F}^b_{min} \\ 0, & \text{otherwise} \end{cases}, \tag{3.15}$$

where $\mathbf{x}$ is the optimized solution of T_a, $\boldsymbol{F}^b(\mathbf{x})$ is its evaluated objective value on T_b, and $\boldsymbol{F}^b_{min}$ is the current best objective value of T_b. Here, $l_i = 1$ indicates that $\mathbf{x}$ is useful for T_b as transferring $\mathbf{x}$ can contribute to accelerating the optimization process of T_b. Otherwise, $l_i = 0$ indicates $\mathbf{x}$ is useless.

3.3.2 Training Data Construction

Algorithm 3.2 Training Data Construction (TDC)

Input: $\mathcal{M}$, n, α
Output: $\mathcal{D}$

Set $\mathcal{D}_+$ and $\mathcal{D}_-$ to two empty sets
while $|\mathcal{D}_+| < n/2$ or $|\mathcal{D}_-| < n/2$
  $[T_a, T_b]$ ← Randomly select two source tasks from $\mathcal{M}$
  $[E_a, E_b]$ ← Extract evolutionary paths of T_a and T_b from $\mathcal{M}$
  $[p^a(\mathbf{x}), p^b(\mathbf{x})]$ ← Build distributions of E_a and E_b by Eq. (3.12) and Eq. (3.13)
  $< \boldsymbol{s}_i, l_i >$ ← Generate the labelled sample by Eq. (3.14) and Eq. (3.15)
  if l_i is 1
    $\mathcal{D}_+ = \mathcal{D}_+ \cup \{< \boldsymbol{s}_i, l_i >\}$
  else
    $\mathcal{D}_- = \mathcal{D}_- \cup \{< \boldsymbol{s}_i, l_i >\}$
  end
end
$\mathcal{D}$ ← Select $n/2$ samples from $\mathcal{D}_+$ and $\mathcal{D}_-$, respectively
return $\mathcal{D} = \{< \boldsymbol{s}_i, l_i >\}_{i=1}^{n}$

For training data construction (TDC), the pseudocode is given in Algorithm 3.2 with the inputs: $\boldsymbol{\mathcal{M}}$ (the knowledge base), $\boldsymbol{n}$ (the number of required training samples), and $\boldsymbol{\alpha}$ (the preference coefficient for estimating the progressional representation). First, $\boldsymbol{\mathcal{D}_+}$ and $\boldsymbol{\mathcal{D}_-}$ are set to two empty sets in Line 1, which are used to save the generated positive and negative samples in Lines 3-11, respectively. In particular, two source tasks T_a and T_b are randomly selected from $\boldsymbol{\mathcal{M}}$, and then their populations are extracted to form their respective evolutionary paths, i.e., $E_a = \{\mathbf{P}_1^a, \ldots, \mathbf{P}_G^a\}$ and $E_b = \{\mathbf{P}_1^b, \ldots, \mathbf{P}_G^b\}$. Here, G is an integer that is smaller than the maximal number of generations of T_a and T_b in $\boldsymbol{\mathcal{M}}$. In this way, a variety of evolutionary paths can be formed by randomly setting the value of G, thereby improving the diversity of the generated samples. With Eq. (3.12) and Eq. (3.13), the progressional representations of E_a and E_b, i.e., $p^a(\mathbf{x}) \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_a, \widehat{\boldsymbol{\sigma}}_a^2)$ and $p^b(\mathbf{x}) \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_b, \widehat{\boldsymbol{\sigma}}_b^2)$, are built to estimate their population distributions, respectively. Subsequently, the feature vector $\boldsymbol{s}_i$ and its label l_i are calculated by Eq. (3.14) and Eq. (3.15), respectively. The sample $< \boldsymbol{s}_i, l_i >$ is added into $\boldsymbol{\mathcal{D}_+}$ if it has a positive label, i.e., l_i is 1. Otherwise, it is added into $\boldsymbol{\mathcal{D}_-}$ as it has a negative label, i.e., l_i is 0. The above procedures will be iteratively executed to generate the samples until both the sizes of $\boldsymbol{\mathcal{D}_+}$ and $\boldsymbol{\mathcal{D}_-}$ reach $\boldsymbol{n/2}$, respectively. Then, in Line 13, the training data $\boldsymbol{\mathcal{D}}$ is formed by selecting $\boldsymbol{n/2}$ samples from $\boldsymbol{\mathcal{D}_+}$ and $\boldsymbol{\mathcal{D}_-}$, respectively. In this way, the number of positive samples is equal to that of negative samples in $\boldsymbol{\mathcal{D}}$, which can avoid the performance deterioration of the fuzzy classifier brought by imbalanced training data. Finally, $\boldsymbol{\mathcal{D}}$ with $\boldsymbol{n}$ training samples is returned in Line 14.

Algorithm 3.3 Solution Usefulness Measurement (SUM)

Input: FKNN, $\boldsymbol{\mathcal{M}}$, E_t, $\mathbf{g}$
Output: $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{Y}}$
1 Set $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{Y}}$ to two empty sets
2 $p^t(\mathbf{x})$ ← Build distribution of E_t by Eq. (3.12) and Eq. (3.13)
3 **for** each $T_i \in \mathcal{M}$
4 E_i ← Extract evolutionary path of T_i during $\mathbf{g}$ generations
5 $p^i(\mathbf{x})$ ← Build distribution of E_i by Eq. (3.12) and Eq. (3.13)
6 $\boldsymbol{v}_i$ ← Generate test sample by Eq. (3.14)
7 $[y_i, m_i]$ ← FKNN ($\boldsymbol{v}_i$)
8 $\boldsymbol{\mathcal{V}} = \boldsymbol{\mathcal{V}} \cup \{< \boldsymbol{v}_i >\}$, $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{Y}} \cup \{< y_i, m_i >\}$
9 **end**
10 **return** $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{Y}}$

3.3.3 Solution Usefulness Measurement

As the evolutionary search proceeds, the populations of the source and target tasks can be collected to compute the feature vectors as the test samples. In this way, the learned fuzzy classifier (FKNN) based on the training data can predict the class labels and the membership degrees of the test samples, which are used to quantify the solution usefulness of the source tasks for the target task. For solution usefulness measurement (SUM), the pseudocode is given in Algorithm 3.3 with the inputs: FKNN (the fuzzy classifier), $\boldsymbol{\mathcal{M}}$ (the knowledge base), E_t (the current evolutionary path of the target task), and $\mathbf{g}$ (the current generation). In Lines 1-2, $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{Y}}$ are first set to two empty sets, and then the distribution of E_t, i.e., $p^t(\mathbf{x}) \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_t, \widehat{\boldsymbol{\sigma}}_t^2)$, is built by Eq. (3.12) and Eq. (3.13). For each source task T_i in $\mathcal{M}$, its test sample is generated based on the estimated solutions of T_i and the target task in Lines 4-6. Particularly, the evolutionary path of T_i including the populations from the initial generation to the current generation g is extracted from $\mathcal{M}$, i.e., $E_i = \{\mathbf{P}_1^i, \ldots, \mathbf{P}_\mathbf{g}^i\}$. Then, the distribution of E_i, i.e., $p^i(\mathbf{x}) \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_i, \widehat{\boldsymbol{\sigma}}_i^2)$, is built by Eq. (3.12) and Eq. (3.13). Thus, the feature vector $\boldsymbol{v}_i = \{v_{i1}, v_{i2}\}$ is calculated as the test sample by Eq. (3.14).

Next, in Line 7, FKNN is used to predict the class label and the membership degree of $\boldsymbol{v}_i$, i.e., y_i and m_i. The former shows that T_i belongs to the positive or negative class while the latter indicates the degree to which it belongs to that class. In Line 8, $< \boldsymbol{v}_i >$ is added into $\boldsymbol{\mathcal{V}}$ while $< y_i, m_i >$ is added into $\boldsymbol{\mathcal{Y}}$. The above procedures in Lines 4-8 are iteratively performed until all source tasks in $\boldsymbol{\mathcal{M}}$ have been visited. Finally, $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{Y}}$ are returned in Line 10.

3.3.4 Solution Selection for Transfer

Algorithm 3.4 Solution Selection for Transfer (SST)

Input: $\boldsymbol{\mathcal{Y}}$, $\boldsymbol{\mathcal{M}}$, c
Output: **TS**, $index$
1 **Q** ← Collect the indices of useful source tasks by Eq. (3.16)
2 **if** **Q** is not empty
3 $index$ ← Find the index of the most useful source task by Eq. (3.17)
4 **TS** ← Randomly select c optimized solutions of T_{index} from $\boldsymbol{\mathcal{M}}$
5 **else**
6 **TS** = ∅, $index$ = 0
7 **end**
8 **return** **TS**, $index$

Based on the measured solution usefulness of source tasks, the useful solutions can be selected from candidate source tasks and then injected into the population of the target task to accelerate the evolutionary search. For solution selection for transfer (SST), the pseudocode is given in Algorithm 3.4 with the inputs: $\boldsymbol{\mathcal{Y}}$ (the set of class labels and membership degrees of the source tasks), $\boldsymbol{\mathcal{M}}$ (the knowledge base), and c (the number of source solutions for transfer at each generation). First, in Line 1, the indices of useful source tasks are collected into $\mathbf{Q}$ by

$$\mathbf{Q} = \{i \mid y_i == 1, \forall y_i \in \boldsymbol{\mathcal{Y}}\}, \tag{3.16}$$

where y_i is the class label of T_i, which indicates that the i-th source task is useful if y_i is 1. Otherwise, T_i is useless. Next, as shown in Lines 3-4, the source solutions are selected for transfer if $\mathbf{Q}$ is not empty. Particularly, the source task with the maximal membership degree (i.e., T_{index}) is considered the most useful source task, which is identified by

$$index = \arg\max_{i \in \mathbf{Q}} \{m_i \mid m_i \in \boldsymbol{\mathcal{V}}\}, \tag{3.17}$$

where m_i is the membership degree of T_i. Then, c optimized solutions of T_{index} are selected from $\boldsymbol{\mathcal{M}}$, which are added into **TS**. If $\mathbf{Q}$ is empty, **TS** is set to an empty set, and the $index$ is set to 0 in Line 6, indicating that the solution transfer will not happen. Finally, **TS** and $index$ are returned in Line 8.

3.3.5 Training Data Update

Algorithm 3.5 Training Data Update (TDU)

Input: **P**, **TS**, $\boldsymbol{\mathcal{V}}$, $\boldsymbol{\mathcal{D}}$, $index$, $flag$
Output: $\boldsymbol{\mathcal{D}}$, $flag$
1 **if** **TS** is not empty
2 $< \boldsymbol{v}_{index} >$ ← Select the test sample of T_{index} from $\boldsymbol{\mathcal{V}}$
3 $< y_{index} >$ ← Assign the label of $\boldsymbol{v}_{index}$ by Eq. (3.18)
4 $\boldsymbol{\mathcal{D}}$ ← Delete the first training sample $< \boldsymbol{v}_1, y_1 >$ from $\boldsymbol{\mathcal{D}}$
5 $\boldsymbol{\mathcal{D}}$ ← Add the new training sample $< \boldsymbol{v}_{index}, y_{index} >$ into $\boldsymbol{\mathcal{D}}$
6 Renumber all training samples in $\boldsymbol{\mathcal{D}}$
7 $flag$ = true
8 **end**
9 **return** $\boldsymbol{\mathcal{D}}$, $flag$

For training data update (TDU), the pseudocode is given in Algorithm 3.5 with the inputs: **P** (the next-generation population), **TS** (the set of solutions for transfer), $\boldsymbol{\mathcal{V}}$ (the test data), $\boldsymbol{\mathcal{D}}$ (the training data), $index$ (the index of the most useful source

task selected for solution transfer), and $flag$ (a boolean variable). As shown in Lines 1-8, if $\mathbf{TS}$ is not an empty set, the training data $\boldsymbol{\mathcal{D}}$ will be updated as the solution transfer occurs at the current generation. In particular, the test sample $\boldsymbol{v}_{index}$ generated based on the populations of T_{index} and the target task is first selected from $\boldsymbol{\mathcal{V}}$ in Line 2. After that, as shown in Line 3, by checking whether any transferred solution of T_{index} survives in the environmental selection, $\boldsymbol{v}_{index}$ is reassigned a label y_{index}, which is expressed by

$$y_{index} = \begin{cases} 0, & \text{if } \mathbf{TS} \cap \mathbf{P} = \emptyset \\ 1, & \text{otherwise} \end{cases}, \tag{3.18}$$

Next, as shown in Lines 4-5, the first training sample denoted by $< \boldsymbol{v}_1, y_1 >$ is deleted from $\boldsymbol{\mathcal{D}}$, and then the new labelled sample $< \boldsymbol{v}_{index}, y_{index} >$ is added into $\boldsymbol{\mathcal{D}}$. In this way, more reliable training samples will gradually replace the inaccurate ones as the evolutionary search proceeds. Meanwhile, the size of $\boldsymbol{\mathcal{D}}$ remains unchanged, which ensures that the training time of the fuzzy classifier will not increase in the optimization process of the target task. After that, all training samples are renumbered sequentially based on the order in which they are added into $\boldsymbol{\mathcal{D}}$ in Line 6. In addition, $flag$ is set to true in Line 7, which shows that the fuzzy classifier needs to be retrained because $\boldsymbol{\mathcal{D}}$ is updated. Finally, $\boldsymbol{\mathcal{D}}$ and $flag$ are returned in Line 9.

3.3.6 Main Framework

Here, the pseudocode of our main framework is provided in Algorithm 3.6 with the inputs: STOP (a sequential transfer optimization problem with T_t and $\boldsymbol{\mathcal{M}}$, where T_t and $\boldsymbol{\mathcal{M}}$ are the target task and the knowledge base consisting of the evaluated population of the source tasks, respectively), N (the population size), G_{max} (the maximum number of generations for T_t), TG (the transfer generation interval), $\boldsymbol{c}$ (the

number of solutions for transfer at each transferable generation), α (the preference coefficient for calculating the progressional representation), n (the number of samples in the training data), and K (the number of nearest neighbours in FKNN). First, a population $\mathbf{P}$ is initialized by randomly generating N solutions and it is collected into E_t as the initial evolutionary path of T_t in Line 1. Next, the training data $\mathcal{D}$ is initialized by performing Algorithm 3.2 in Line 2. Then, in Line 3, the generation counter g is set to 1, and $flag$ is set to true, which indicates that FKNN needs to be trained on $\mathcal{D}$.

Algorithm 3.6 The Main Framework

Input: An STOP with T_t and $\mathcal{M}$, N, G_{max}, TG, c, α, n, K
Output: $\mathbf{P}$

```
1   Initialize P with N solutions and E_t = {P}
2   D ← TDC (M, n, α)   // Algorithm 3.2
3   Set g = 1, and flag = true
4   while g ≤ G_max
5       if mod (g, TG) == 0
6           if flag is true
7               FKNN ← Retrain a fuzzy classifier on D   // Algorithm 3.1
8               flag = false
9           end
10          [V, Y] ← SUM (FKNN, M, E_t, g)   // Algorithm 3.3
11          [TS, index] ← SST (Y, M, c)   // Algorithm 3.4
12      end
13      O ← Crossover and Mutation on P
14      P ← Environmental Selection on P∪O∪TS
15      [D, flag] ← TDU (P, TS, V, D, index, flag)   // Algorithm 3.5
16      g = g + 1 and E_t = E_t ∪ {P}
17  end
18  return P
```

The main evolutionary search process is shown in Lines 4-17. First, as shown in Lines 5-12, the component of solution transfer is triggered at each of TG generations. In particular, in Lines 6-9, the fuzzy classifier FKNN is trained on $\mathcal{D}$ if $flag$ is

true, and then $flag$ is set to false. After that, with FKNN, Algorithm 3.3 is performed to measure the solution usefulness of all candidate source tasks from $\mathcal{M}$ in Line 10. Then, the solutions for transfer, i.e., $\mathbf{TS}$, are selected by running Algorithm 3.4 in Line 11. Afterward, the simulated binary crossover (SBX) [94] and polynomial-based mutation (PM) [95] are sequentially executed to generate the offspring population $\mathbf{O}$ with the size of $N - |\mathbf{TS}|$ in Line 13. Next, in Line 14, environmental selection is performed on the combined population $\mathbf{P} \cup \mathbf{O} \cup \mathbf{TS}$ to select N solutions based on their objective values, which are employed to form the next-generation population $\mathbf{P}$. Then, in Line 15, Algorithm 3.5 is performed to update $\mathcal{D}$ by replacing the old one with the newly generated training sample with the more accurate label. In Line 16, the generation counter $\mathbf{g}$ is increased by 1, and $\mathbf{P}$ is added into E_t. When $\mathbf{g}$ does not exceed G_{max}, the above process will be iteratively executed. Otherwise, $\mathbf{P}$ is returned as an approximate solution set in Line 18.

3.4 Experimental Study

3.4.1 Experimental Setup

1) Compared Algorithms

The canonical EA: SBX and PM are used to generate the offspring population, while 1/2 truncation selection is used to select elite solutions from parent and offspring populations.

Four ESTO algorithms equipped with distance metrics: ESTOA-ED, ESTOA-WD, ESTOA-KLD, and ESTOA-MMD use ED, WD, KLD, and MMD to select optimized solutions from the most similar source task for transfer, respectively.

Two ESTO algorithms equipped with ML models: ESTOA-AD and ESTOA-INB are developed by replacing the fuzzy classifier in ESTOA-FCM with AD and INB. In ESTOA-AD and ESTOA-INB, one source task in the positive class is randomly selected to provide optimized solutions for transfer if it exists. Otherwise, knowledge transfer will not happen.

2) Parameter Settings

Table 3.1: Parameter settings of all compared algorithms.

Algorithm	Parameter settings
EA	N = 50
ESTOA-ED	N = 50, TG = 1, c = 1
ESTOA-WD	N = 50, TG = 1, c = 1
ESTOA-KLD	N = 50, TG = 1, c = 1
ESTOA-MMD	N = 50, TG = 1, c = 1, σ = 0.5
ESTOA-AD	N = 50, TG = 1, c = 1, ε = 0.1
ESTOA-INB	N = 50, TG = 1, c = 1
ESTOA-FCM	N = 50, TG = 1, c = 1, α = 0.2, n = 100, K = 5

Tab. 3.1 lists the detailed parameter settings of all compared algorithms. Here, SBX with p_c = 1 and η_c = 15, and PM with $p_m = 1/d$ and η_m = 15 are used as the evolutionary operators for generating offspring population. The population size (N), the transfer generation interval (TG), and the number of transferred solutions at each transferable generation (c) are set to 50, 1, and 1, respectively. In addition, in ESTOA-MMD, σ is set to 0.5 in the Gaussian kernel function. Moreover, in ESTOA-AD, the threshold ε is set to 0.1. In ESTOA-FCM, α, n, and K are set to 0.2, 100, and 5, respectively.

The maximum number of generations G_{max} on each test problem is set to 100. The objective values from 50 independent runs of each compared algorithm on the test

problems are collected for performance comparison, where the Wilcoxon rank sum test with a 0.05 significance level is used to show the statistically significant differences in the numerical results.

Table 3.2: Parameter Settings of F1-F18.

ID	Target Task			Source Tasks in Knowledge Base		
	Function	Search Space	Normalized Global Optimum	Function	Search Space	Normalized Global Optimum
F1	CIHS-T1	d = 50, $[-100, 100]^d$	$\mathbf{o}_t^1 = [\underbrace{0.5, \ldots, 0.5}_{d}]$	CIHS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^2 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F2	CIHS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^2 = [\underbrace{0.5, \ldots, 0.5}_{d}]$	CIHS-T1	d = 50, $[-100, 100]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^1 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F3	CIMS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^3 = [\underbrace{0.5, \ldots, 0.5}_{d}]$	CIMS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^4 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F4	CIMS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^4 = [\underbrace{0.5, \ldots, 0.5}_{d}]$	CIMS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^3 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F5	CILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^5 = [\underbrace{0.92, \ldots, 0.92}_{d}]$	CILS-T2	d = 50, $[-500, 500]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^6 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F6	CILS-T2	d = 50, $[-500, 500]^d$	$\mathbf{o}_t^6 = [\underbrace{0.92, \ldots, 0.92}_{d}]$	CILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^5 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F7	PIHS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^7 = [\underbrace{0.5, \ldots, 0.5}_{d}]$	PIHS-T2	d = 50, $[-100, 100]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^8 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F8	PIHS-T2	d = 50, $[-100, 100]^d$	$\mathbf{o}_t^8 = [\underbrace{0.5, \ldots, 0.6}_{d/2} \underbrace{\quad}_{d/2}]$	PIHS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^7 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F9	PIMS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^9 = [\underbrace{0.5, \ldots, 0.51}_{d/2} \underbrace{\quad}_{d/2}]$	PIMS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{10} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F10	PIMS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^{10} = [\underbrace{0.51, \ldots, 0.51}_{d}]$	PIMS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^9 \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F11	PILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^{11} = [\underbrace{0.5, \ldots, 0.5}_{d}]$	PILS-T2	d = 25, $[-0.5, 0.5]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{12} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F12	PILS-T2	d = 25, $[-0.5, 0.5]^d$	$\mathbf{o}_t^{12} = [\underbrace{0.5, \ldots, 0.5}_{d}]$	PILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{11} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F13	NIHS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^{13} = [\underbrace{0.51, \ldots, 0.51}_{d}]$	NIHS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{14} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F14	NIHS-T2	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^{14} = [\underbrace{0.5, \ldots, 0.5}_{d}]$	NIHS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{13} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F15	NIMS-T1	d = 50, $[-100, 100]^d$	$\mathbf{o}_t^{15} = [\underbrace{0.55, \ldots, 0.55}_{d}]$	NIMS-T2	d = 50, $[-0.5, 0.5]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{16} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F16	NIMS-T2	d = 50, $[-0.5, 0.5]^d$	$\mathbf{o}_t^{16} = [\underbrace{0.5, \ldots, 0.5}_{d}]$	NIMS-T1	d = 50, $[-100, 100]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{15} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F17	NILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_t^{17} = [\underbrace{0.5, \ldots, 0.5}_{d}]$	NILS-T2	d = 50, $[-500, 500]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{18} \times (1-\tau_i) + \mathbf{r} \times \tau_i$
F18	NILS-T2	d = 50, $[-500, 500]^d$	$\mathbf{o}_t^{18} = [\underbrace{0.92, \ldots, 0.92}_{d}]$	NILS-T1	d = 50, $[-50, 50]^d$	$\mathbf{o}_{si} = \mathbf{o}_t^{17} \times (1-\tau_i) + \mathbf{r} \times \tau_i$

3) Test Problems

Two benchmark suites are used for performance comparison. The first benchmark suite includes 18 STOPs (called F1-F18), which are constructed based on the existing ingredient functions in the multitasking benchmark suite [96]. The second benchmark suite consists of 12 STOPs (called STOP1-STOP12), which are generated by a specially designed STOP generator [97]. In addition, a practical case is considered by using the planar kinematic arm problem [98] as the ingredient function to construct a series of practical test instances. The source and target tasks are continuous optimization problems and the real-valued encoding is used to represent the solutions of source and target tasks. Note that the canonical EA is used as the optimizer to collect the estimated solutions of source tasks of each test problem to form their respective knowledge bases. For consistency, the maximum numbers of generations for all source tasks are set to the same to that of their target tasks. The detailed descriptions of the two benchmark suites are given as follows:

In the commonly used multitasking benchmark suite [96], nine test problems (i.e., CIHS, CIMS, CILS, PIHS, PIMS, PILS, NIHS, NIMS, and NILS) are designed by considering the task similarity and the degree of global optima intersection, each of which has two different tasks. To construct the sequential transfer optimization problem (STOP), the target task is set to one existing task from one multitasking test problem while its associated knowledge base is formed by configuring another task with different global optima to generate the source tasks. As shown in Tab. 3.2, a weight parameter τ_i in [0, 1] is introduced as the perturbation factor to generate the normalized global optima of the source tasks in the normalized search space $[0, 1]^d$. Setting τ_i to 0 means that the normalized global optimum of the generated source task is the same as that of the original task. When τ_i is set to 1, its normalized global optimum

is generated by randomly sampling in the normalized search space. Therefore, using the Gaussian distribution to sample τ_i can diversify the relationship between the normalized global optima of the source tasks and the original task. Note that $\mathbf{r}$ is a d-dimensional vector randomly sampled in $[0, 1]^d$. Here, the mean (μ), the standard deviation (σ), and the number of the source tasks (k) are 0.5, 0.1, and 100, respectively. The detailed parameter settings of F1-F18 are given in Tab. 3.2.

As introduced in [97], STOPs with diverse properties can be generated by using a problem generator with six necessary parameters, including task family ($\mathcal{TF}$), transfer scenario ($\mathcal{TS}$), optimum coverage of the image (ξ), similarity distribution ($\mathcal{SD}$), problem dimension (d), and the number of source tasks (k). By setting different parameters, the similarity distribution of source tasks to the target task can be flexibly adjusted according to specific requirements. Therefore, the problem generator generates a specific STOP by setting the parameters, i.e., $\mathcal{TF}$-$\mathcal{TS}$-ξ-$\mathcal{SD}$-k-d. Here, eight widely used single-objective optimization functions are employed as the candidate families for formulating the source and target tasks, i.e., $\mathcal{TF}$ = {Sphere, Ellipsoid, Schwefel 2.2, Quartic, Ackley, Rastrigin, Griewank, Levy}. In addition, there are two different transfer scenarios, i.e., $\mathcal{TS} = \{\mathcal{T}_a, \mathcal{T}_e\}$. The former shows that the source and target tasks belong to the same family while the latter indicates that they have different families. The parameter $\xi \in [0,1]$ determines the relative size of the image over the decision space. To create a series of STOPs with diverse similarity distributions, $\tau_i \in [0,1]$ is used to adjust the relationship between the optimal solutions of source and target tasks, which is given as follows:

$$\begin{cases} \mathbf{o}_t = \widehat{\mathbf{x}}_{lb} + \mathbf{r} \times (\widehat{\mathbf{x}}_{ub} - \widehat{\mathbf{x}}_{lb}) \\ \mathbf{o}_{si}^b = \widehat{\mathbf{x}}_{lb} + \mathbf{r} \times (\widehat{\mathbf{x}}_{ub} - \widehat{\mathbf{x}}_{lb}), i = 1, 2, \ldots, k \\ \mathbf{o}_{si} = \mathbf{o}_t \times \tau_i + \mathbf{o}_{si}^b \times (1 - \tau_i), i = 1, 2, \ldots, k \end{cases} \tag{3.19}$$

where $\mathbf{o}_t$ and $\mathbf{o}_{si}$ are the optima of the target task and the i-th source task, and τ_i is the weight parameter for the i-th source task. Note that $\mathbf{r}$ is a d-dimensional vector randomly sampled in $[0, 1]^d$. Here, five different probability distributions of τ_i are built based on one or multiple Gaussian distributions with the pre-set mean and standard deviation, which are respectively presented, as follows:

$$p_1(\tau) = \begin{cases} \mathcal{N}(0.15, 0.1^2), i = 1, \ldots, \lfloor 2k/3 \rfloor \\ \mathcal{N}(0.45, 0.2^2), i = \lfloor 2k/3 \rfloor + 1, \ldots, k \end{cases} \tag{3.20}$$

$$p_2(\tau) = \begin{cases} \mathcal{N}(0.45, 0.2^2), i = 1, \ldots, \lfloor k/3 \rfloor \\ \mathcal{N}(0.70, 0.1^2), i = \lfloor k/3 \rfloor + 1, \ldots, k \end{cases} \tag{3.21}$$

$$p_3(\tau) = \mathcal{N}(0.45, 0.2^2), i = 1, \ldots, k \tag{3.22}$$

$$p_4(\tau) = \begin{cases} \mathcal{N}(0.15, 0.1^2), i = 1, \ldots, \lfloor k/3 \rfloor \\ \mathcal{N}(0.45, 0.1^2), i = \lfloor k/3 \rfloor + 1, \ldots, \lfloor 2k/3 \rfloor \\ \mathcal{N}(0.70, 0.1^2), i = \lfloor 2k/3 \rfloor + 1, \ldots, k \end{cases} \tag{3.23}$$

$$p_5(\tau) = \begin{cases} \mathcal{N}(0.15, 0.1^2), i = 1, \ldots, \lfloor k/2 \rfloor \\ \mathcal{N}(0.70, 0.1^2), i = \lfloor k/2 \rfloor + 1, \ldots, k \end{cases} \tag{3.24}$$

where k is the number of source tasks and $\lfloor \cdot \rfloor$ is the operator of rounding down. The similarity between the global optimum of the i-th source task and the target task is measured by

$$\mathcal{S}_i = 1 - \max_j \left(|\mathbf{o}_t^j - \mathbf{o}_{si}^j| \right), \tag{3.25}$$

where $\mathbf{o}_t^j$ and $\mathbf{o}_{si}^j$ denote the j-th variables of $\mathbf{o}_t$ and $\mathbf{o}_{si}$, and $|\cdot|$ denotes the absolute value. The computed similarity degrees in the ranges, i.e., [0, 0.3], (0.3, 0.7), and [0.7, 1], are considered to be high, medium, and low, respectively. Therefore, $\mathcal{SD} = \{p_1(\tau), p_2(\tau), p_3(\tau), p_4(\tau), p_5(\tau)\}$ is employed to mimic the diversity of

similarity relationships of the source task to the target task in various STOPs. Moreover, d and k are the dimensionality of problem and the number of source tasks, respectively.

Note that different STOPs can possess different dimensions, but the source and target tasks have the same dimension in the same STOP. In the problem generator [97], a conventional evolutionary algorithm is employed as the basic solver to optimize all the source tasks and their evaluated solutions are collected to form the knowledge base. According to the above parameter settings, a benchmark suite consisting of STOP1-STOP12 is designed as test problems for examining the performance of ES-TO algorithms, where their knowledge bases are configured with different proportions of the source tasks with low, medium, and high similarity for their target tasks. The detailed parameter settings are given in Tab. 3.3.

Table 3.3: Parameter Settings of STOP1-STOP12.

ID	Problem Specification ($\mathcal{TF}$-$\mathcal{TS}$-ξ-$\mathcal{SD}$-k-d)	The Proportions of Different Types of Source Tasks		
		Low ($0 \leq \mathcal{S} \leq 0.3$)	Medium ($0.3 < \mathcal{S} < 0.7$)	High ($0.7 \leq \mathcal{S} \leq 1$)
STOP1	Sphere-$\mathcal{T}_a$-1-$\mathcal{N}_1$-35-100	38.00%	57.00%	5.00%
STOP2	Ellipsoid-$\mathcal{T}_e$-1-$\mathcal{N}_1$-50-100	47.00%	49.00%	4.00%
STOP3	Schwefel-$\mathcal{T}_a$-1-$\mathcal{N}_1$-60-100	56.00%	40.00%	4.00%
STOP4	Quartic-$\mathcal{T}_e$-1-$\mathcal{N}_2$-35-100	3.00%	44.00%	53.00%
STOP5	Ackley-$\mathcal{T}_a$-1-$\mathcal{N}_2$-50-100	6.00%	48.00%	46.00%
STOP6	Rastrigin-$\mathcal{T}_e$-1-$\mathcal{N}_2$-60-100	2.00%	52.00%	46.00%
STOP7	Griewank-$\mathcal{T}_a$-1-$\mathcal{N}_3$-35-100	13.00%	75.00%	12.00%
STOP8	Levy-Te-$\mathcal{T}_e$-1-$\mathcal{N}_3$-35-200	9.00%	72.00%	19.00%
STOP9	Ellipsoid-$\mathcal{T}_a$-1-$\mathcal{N}_4$-50-100	25.00%	49.00%	26.00%
STOP10	Quartic-$\mathcal{T}_e$-1-$\mathcal{N}_4$-50-200	24.50%	51.50%	24.00%
STOP11	Ackley-$\mathcal{T}_a$-1-$\mathcal{N}_5$-60-100	36.00%	31.00%	33.00%
STOP12	Griewank-$\mathcal{T}_e$-1-$\mathcal{N}_5$-60-200	35.00%	32.50%	32.50%

Table 3.4: Mean objective values and standard deviations obtained by ESTOA-FCM and compared algorithms on F1-F18.

Problem		EA	ESTOA-ED	ESTOA-WD	ESTOA-KLD	ESTOA-MMD	ESTOA-AD	ESTOA-INB	ESTOA-FCM
F1	mean	1.05e+00(-)	1.02e+00(~)	1.02e+00(~)	1.02e+00(~)	1.03e+00(-)	1.03e+00(-)	1.03e+00(-)	**1.01e+00**
	std	1.79e-02	2.92e-02	3.08e-02	2.98e-02	3.80e-02	1.67e-02	1.90e-02	**2.86e-02**
F2	mean	5.52e+02(~)	5.57e+02(-)	5.52e+02(~)	5.39e+02(~)	5.62e+02(-)	**5.36e+02(~)**	5.39e+02(~)	5.36e+02
	std	4.18e+01	3.98e+01	4.55e+01	3.62e+01	4.43e+01	**3.26e+01**	3.32e+01	3.78e+01
F3	mean	9.44e+00(-)	5.50e+00(-)	5.57e+00(-)	5.36e+00(-)	6.02e+00(-)	5.41e+00(-)	5.26e+00(~)	**5.09e+00**
	std	1.83e+00	7.20e-01	7.67e-01	4.35e-01	8.42e-01	4.31e-01	4.87e-01	**5.01e-01**
F4	mean	5.53e+02(-)	5.41e+02(~)	5.37e+02(~)	5.31e+02(~)	5.50e+02(~)	**5.28e+02(~)**	5.58e+02(-)	5.39e+02
	std	4.07e+01	3.19e+01	3.23e+01	2.82e+01	4.59e+01	**3.81e+01**	3.94e+01	3.20e+01
F5	mean	2.13e+01(~)	2.13e+01(~)	2.13e+01(~)	2.13e+01(~)	**2.13e+01(~)**	2.13e+01(-)	2.13e+01(~)	2.13e+01
	std	4.12e-02	8.53e-02	9.47e-02	3.92e-02	**3.36e-02**	4.36e-02	3.94e-02	4.55e-02
F6	mean	**1.99e+03(~)**	2.11e+03(~)	2.18e+03(~)	2.19e+03(~)	2.14e+03(~)	2.15e+03(~)	2.10e+03(~)	2.09e+03
	std	**5.21e+02**	5.17e+02	4.93e+02	4.64e+02	5.48e+02	4.44e+02	4.61e+02	4.42e+02
F7	mean	5.58e+02(-)	5.58e+02(-)	5.63e+02(-)	5.39e+02(~)	5.61e+02(-)	5.42e+02(~)	5.59e+02(-)	**5.34e+02**
	std	4.63e+01	5.05e+01	4.22e+01	3.75e+01	3.18e+01	2.81e+01	3.68e+01	**3.52e+01**
F8	mean	2.33e+02(-)	2.01e+02(~)	1.90e+02(~)	**1.65e+02(~)**	2.38e+02(~)	1.68e+02(~)	2.50e+02(-)	1.82e+02
	std	6.24e+01	7.43e+01	7.80e+01	**4.31e+01**	1.35e+02	3.77e+01	6.51e+01	4.94e+01
F9	mean	1.02e+01(-)	5.31e+00(-)	5.12e+00(-)	5.32e+00(-)	6.15e+00(-)	5.36e+00(-)	9.83e+00(-)	**4.79e+00(~)**
	std	2.28e+00	5.40e-01	5.80e-01	3.24e-01	1.36e+00	4.32e-01	2.26e+00	**3.19e-01**
F10	mean	3.03e+05(-)	1.23e+05(-)	1.08e+05(-)	6.99e+04(-)	7.80e+04(-)	7.11e+04(-)	3.01e+05(-)	**3.89e+04(~)**
	std	1.65e+05	7.56e+04	6.50e+04	3.26e+04	4.16e+04	3.88e+04	1.38e+05	**2.16e+04**
F11	mean	9.67e+00(-)	7.37e+00(~)	7.71e+00(~)	7.97e+00(-)	**7.33e+00(~)**	7.80e+00(~)	8.75e+00(-)	7.50e+00(~)
	std	1.68e+00	9.47e-01	1.09e+00	1.09e+00	**8.96e-01**	1.08e+00	1.76e+00	1.01e+00
F12	mean	9.64e+00(-)	7.05e+00(~)	6.87e+00(~)	**6.44e+00(~)**	7.16e+00(~)	6.84e+00(~)	7.64e+00(-)	6.72e+00(~)
	std	1.91e+00	1.64e+00	1.47e+00	**1.37e+00**	2.20e+00	1.30e+00	2.11e+00	1.34e+00
F13	mean	3.43e+05(-)	9.03e+04(-)	9.80e+04(-)	6.95e+04(-)	1.18e+05(-)	7.37e+04(-)	7.46e+04(-)	**2.37e+04(~)**
	std	1.67e+05	6.68e+04	6.24e+04	2.78e+04	7.87e+04	2.85e+04	3.94e+04	**1.16e+04**
F14	mean	5.51e+02(-)	5.53e+02(-)	5.51e+02(~)	**5.31e+02(~)**	5.49e+02(~)	5.35e+02(~)	5.51e+02(~)	5.36e+02(~)
	std	3.61e+01	3.09e+01	3.69e+01	**3.34e+01**	4.02e+01	3.38e+01	4.33e+01	3.53e+01
F15	mean	1.05e+00(-)	1.04e+00(~)	1.05e+00(~)	1.04e+00(~)	1.06e+00(-)	1.05e+00(~)	**1.03e+00(+)**	1.05e+00(~)
	std	1.97e-02	1.90e-02	1.57e-02	1.98e-02	2.21e-02	1.63e-02	**2.83e-02**	1.71e-02
F16	mean	3.41e+01(-)	2.14e+01(~)	2.04e+01(~)	2.03e+01(~)	2.32e+01(-)	**2.00e+01(~)**	3.23e+01(-)	2.11e+01(~)
	std	4.86e+00	2.98e+00	3.53e+00	2.12e+00	5.26e+00	**2.93e+00**	4.05e+00	3.45e+00
F17	mean	5.62e+02(~)	5.70e+02(~)	5.72e+02(~)	5.69e+02(~)	5.62e+02(~)	**5.59e+02(~)**	5.67e+02(~)	5.62e+02(~)
	std	3.63e+01	4.03e+01	4.21e+01	3.67e+01	3.55e+01	**4.15e+01**	4.49e+01	3.34e+01
F18	mean	2.08e+03(~)	2.14e+03(~)	2.05e+03(~)	2.25e+03(~)	2.12e+03(~)	2.14e+03(~)	**1.97e+03(~)**	2.04e+03(~)
	std	5.58e+02	4.51e+02	4.28e+02	4.83e+02	5.21e+02	4.23e+02	**4.96e+02**	4.83e+02
Best/All		1/18	0/18	0/18	3/18	2/18	4/18	2/18	6/18
+/-/~		0/13/5	0/7/11	0/5/13	0/5/13	0/9/9	0/6/12	1/10/7	\

"+", "-", and "~" indicate that the results of the corresponding algorithm are better than, worse than, and similar to that of ESTOA-FCM, respectively. The best result on each test problem is highlighted in bold.

Table 3.5: Mean objective values and standard deviations obtained by ESTOA-FCM and compared algorithms on STOP1-STOP12.

Problem		EA	ESTOA-ED	ESTOA-WD	ESTOA-KLD	ESTOA-MMD	ESTOA-AD	ESTOA-INB	ESTOA-FCM
STOP1	mean	1.65e+02(~)	2.76e+02(-)	2.59e+02(-)	1.69e+02(~)	2.74e+02(-)	**1.57e+02**(~)	2.16e+02(~)	1.67e+02
	std	1.09e+02	1.46e+02	1.13e+02	1.20e+02	1.25e+02	**1.07e+02**	1.55e+02	9.52e+01
STOP2	mean	4.72e+03(-)	3.96e+03(-)	3.40e+03(-)	3.63e+03(-)	3.55e+03(-)	3.20e+03(~)	4.35e+03(-)	**2.71e+03**
	std	2.09e+03	1.58e+03	1.05e+03	1.56e+03	1.23e+03	1.40e+03	2.23e+03	**1.01e+03**
STOP3	mean	2.23e+01(~)	3.23e+01(-)	2.91e+01(-)	2.44e+01(~)	2.75e+01(~)	**2.05e+01**(+)	2.24e+01(~)	2.57e+01
	std	8.16e+00	1.39e+01	9.83e+00	9.67e+00	9.10e+00	**7.17e+00**	8.26e+00	8.81e+00
STOP4	mean	4.05e+00(-)	5.20e-01(-)	6.11e-01(-)	5.34e-01(-)	5.58e-01(-)	3.96e-01(~)	6.36e-01(-)	**3.79e-01**
	std	5.08e+00	2.18e-01	2.52e-01	2.38e-01	1.94e-01	2.09e-01	3.78e-01	**1.51e-01**
STOP5	mean	7.43e+00(-)	4.24e+00(-)	4.33e+00(-)	**3.68e+00**(~)	4.37e+00(-)	4.09e+00(-)	7.57e+00(-)	3.80e+00
	std	1.49e+00	6.32e-01	6.52e-01	**5.03e-01**	6.33e-01	5.59e-01	1.16e+00	6.08e-01
STOP6	mean	1.46e+02(-)	6.93e+01(~)	6.99e+01(-)	9.02e+01(-)	7.27e+01(~)	8.19e+01(-)	9.37e+01(-)	**6.73e+01**
	std	1.79e+01	2.58e+01	3.91e+01	1.60e+01	2.46e+01	1.98e+01	2.53e+01	**1.28e+01**
STOP7	mean	1.17e+00(-)	1.06e+00(~)	1.06e+00(~)	1.07e+00(-)	1.07e+00(-)	1.11e+00(-)	1.17e+00(-)	**1.05e+00**
	std	8.32e-02	3.85e-02	2.97e-02	3.88e-02	3.41e-02	9.11e-02	1.06e-01	**3.56e-02**
STOP8	mean	6.64e+00(-)	1.33e+00(~)	1.68e+00(-)	3.15e+00(-)	1.73e+00(~)	3.14e+00(-)	7.16e+00(-)	**1.27e+00**
	std	4.21e+00	7.09e-01	8.54e-01	2.35e+00	1.45e+00	2.39e+00	4.28e+00	**6.42e-01**
STOP9	mean	4.17e+03(-)	1.32e+03(~)	1.85e+03(-)	**1.30e+03(~)**	1.96e+03(-)	1.83e+03(-)	4.67e+03(-)	1.37e+03
	std	2.21e+03	4.95e+02	7.72e+02	**6.30e+02**	6.89e+02	1.00e+03	2.72e+03	4.68e+02
STOP10	mean	1.05e+02(-)	6.85e+00(~)	7.38e+00(~)	1.29e+01(-)	9.26e+00(~)	1.17e+01(-)	7.12e+01(-)	**6.04e+00**
	std	8.15e+01	7.21e+00	4.90e+00	1.04e+01	8.08e+00	8.50e+00	8.46e+01	**2.89e+00**
STOP11	mean	9.67e+00(-)	5.49e+00(-)	5.40e+00(-)	5.71e+00(-)	5.68e+00(-)	5.17e+00(~)	9.40e+00(-)	**5.04e+00**
	std	1.29e+00	1.01e+00	9.34e-01	9.40e-01	1.18e+00	7.78e-01	1.53e+00	**8.41e-01**
STOP12	mean	2.88e+00(-)	1.47e+00(-)	1.45e+00(~)	1.98e+00(-)	1.47e+00(-)	1.70e+00(-)	2.82e+00(-)	**1.38e+00**
	std	8.30e-01	2.52e-01	2.00e-01	6.05e-01	3.36e-01	2.90e-01	7.60e-01	**1.30e-01**
Best/All		0/12	0/12	0/12	2/12	0/12	2/12	0/12	8/12
+/-/~		0/10/2	0/7/5	0/9/3	0/8/4	0/8/4	1/7/4	0/10/2	\

"+", "-", and "~" indicate that the results of the corresponding algorithm are better than, worse than, and similar to that of ESTOA-FCM, respectively. The best result on each test problem is highlighted in bold.

3.4.2 Comparison with Peer Methods

The numerical results of all compared algorithms on two test suites are provided in Tab. 3.4 and Tab. 3.5, respectively.

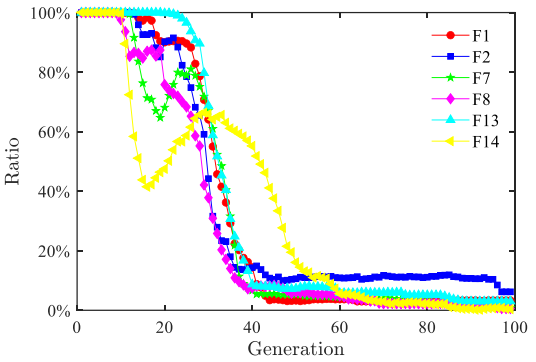
1) Comparison on F1-F18

As shown in Tab. 3.4, for F1-F18, ESTOA-FCM performs better than other compared algorithms on 13, 7, 5, 5, 9, 6, and 10 out of 18 cases, respectively. In addition, ES-

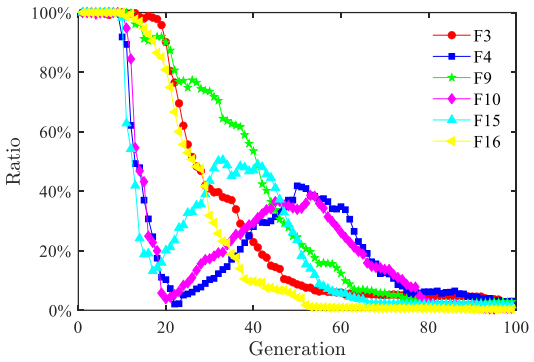
TOA-FCM and the compared algorithms obtain similar performance on most test cases (i.e., 5, 13, 13, 13, 9, 12, and 7 out of 18 cases, respectively). The above comparison results on F1-F18 show some advantages of ESTOA-FCM on some test problems when compared to its competitors.

Moreover, to further analyse the effectiveness of ESTOA-FCM in measuring solution usefulness of source tasks, the ratios of the source tasks with a positive label on F1-F18 are recorded at each generation in Fig. 3.3. In particular, for F1, F2, F7, F8, F13, and F14, as the similarity degree between the ingredient functions for constructing source and target tasks is high, there are a lot of useful source tasks for their corresponding target tasks. These source tasks can provide useful optimized solutions to accelerate the optimization of the target task at the early stage. However, as the evolutionary search proceeds, these optimized solutions will no longer be useful for the target task as they have been used. As observed in Fig. 3.3(a), on most test problems, the number of source tasks with a positive label gradually decreases as the evolutionary search proceeds. In contrast, the degree of similarity between the ingredient functions of source and target tasks is median for F3, F4, F9, F10, F15, and F16. As shown in Fig. 3.3(b), the similar results are observed on F3, F9, and F16. However, on F4, F10, and F15, the number of source tasks with a positive label initially decreases, then increases, and finally decreases again. This could possibly be that the source tasks have a low similarity degree to the target task at the early stage while they have a higher similarity degree at the middle stage. In addition, for F5, F6, F11, F12, F17, and F18, due to the low similarity degree between the ingredient functions of source and target tasks, the number of useful source tasks is very low. In Fig. 3.3(c), except at the very early stage, the number of source tasks with a positive label on most test problems is small and even close to 0. It shows that most source tasks

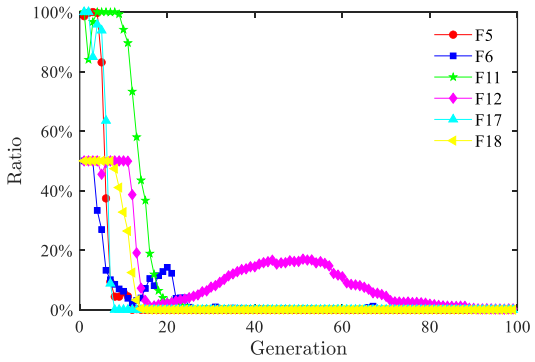
are accurately identified to be useless as they cannot provide useful source solutions for the target task. The above observations in Fig. 3.3 show that ESTOA-FCM can effectively select useful source tasks along the evolutionary search process on most test problems.



(a) F1, F2, F7, F8, F13, and F14



(b) F3, F4, F9, F10, F15, and F16



(c) F5, F6, F11, F12, F17, and F18

Figure 3.3: Ratios of the source tasks with a positive label during the evolutionary search processes on F1-F18.

2) Comparison on STOP1-STOP12

In addition, the superiority of ESTOA-FCM can be observed on STOP1-STOP12. As shown in Tab. 3.5, compared with EA, ESTOA-FCM achieves significantly better

performance on most test problems (i.e., 10 out of 12 cases), which shows that ESTOA-FCM is able to further improve optimization performance on the target task by selecting the source solutions for transfer. However, as the proportion of highly similar source tasks for the target task is very low in the knowledge bases of STOP1, STOP2, and STOP3, the optimized solutions of most source tasks are useless for accelerating the optimization process of the target task. Thus, it is reasonable that ESTOA-FCM and EA obtain similar results on STOP1 and STOP3. Similarly, in comparison with four ESTO algorithms equipped with distance metrics, ESTOA-FCM performs significantly better on most test problems (i.e., 7, 9, 8, and 8 out of 12 cases), while it achieves similar results on the remaining cases. The comparison results show that FCM has better performance than these existing distance metrics including ED, WD, KLD, and MMD. In addition, the significant superiority of ESTOA-FCM is observed when compared with ESTOA-AD and ESTOA-INB. For example, the total numbers of better and similar results obtained by ESTOA-FCM are 11 and 12 out of 12 cases, respectively. The comparisons demonstrate the effectiveness of FCM in measuring the solution usefulness of source tasks when compared with existing measurement methods.

Moreover, to further analyse the effectiveness of ESTOA-FCM, Fig. 3.4 records the ratios of the source tasks with a positive label at each generation on some representative test problems, i.e., STOP3, STOP7, STOP9, and STOP6. In their respective knowledge bases, the true ratios of highly similar source tasks for the target task are 4%, 12%, 26%, and 46%, respectively. In particular, due to the very low proportion of source tasks with high similarity (4%), the number of useful source tasks for the target task is very small on STOP3. As shown in Fig. 3.4, the majority of source tasks have a negative label, which demonstrates that ESTOA-FCM is capable of predicting

that most source tasks are useless. In addition, for STOP6, the true proportion of highly similar source tasks is very high (46%), which means that a lot of source tasks can provide the optimized solutions to accelerate the optimization process of the target task. In Fig. 3.4, for STOP6, the majority of source tasks have a positive label at the early stage. However, as the population of the target task gradually converges, these source tasks will no longer be able to provide significant performance improvement for accelerating the optimization on the target task. Thus, as the optimization process continues, the ratio of source tasks with a positive label gradually declines. Similar observations can be found on STOP7 and STOP9 with a relatively moderate proportion of highly similar source tasks (i.e., 12% and 26%, respectively). Additionally, Fig. 3.4 reveals that the ratio of the source tasks with a positive label increases as the proportion of source tasks with high similarity increases on STOP3, STOP7, STOP9, and STOP6. The above comparisons further validate the effectiveness and superiority of ESTOA-FCM in measuring the usefulness of the source tasks for the target task.

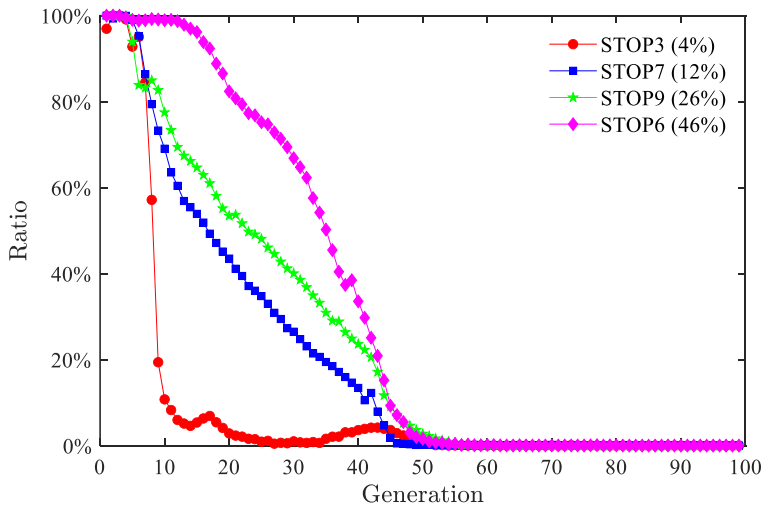


Figure 3.4: Ratios of the source tasks with a positive label during the evolutionary search processes of STOP3, STOP6, STOP7, and STOP9.

Table 3.6: Mean objective values and standard deviations obtained by ESTOA-FCM and its variants.

Problem	Variant-I	Variant-II	Variant-III	ESTOA-FCM
STOP1	1.64e+02(1.01e+02)~	1.56e+02(1.19e+02)~	1.07e+02(6.32e+01)+	1.67e+02(9.52e+01)
STOP2	3.20e+03(1.27e+03)-	4.43e+03(2.39e+03)-	2.79e+03(1.58e+03)~	2.71e+03(1.01e+03)
STOP3	2.03e+01(7.15e+00)+	2.02e+01(7.35e+00)+	1.95e+01(8.02e+00)+	2.57e+01(8.81e+00)
STOP4	2.05e+00(2.10e+00)-	2.61e+00(2.28e+00)-	4.83e-01(1.67e-01)-	3.79e-01(1.51e-01)
STOP5	4.84e+00(9.19e-01)-	7.88e+00(1.22e+00)-	4.92e+00(7.91e-01)-	3.80e+00(6.08e-01)
STOP6	1.82e+02(3.80e+01)-	1.47e+02(2.08e+01)-	1.14e+02(2.64e+01)-	6.73e+01(1.28e+01)
STOP7	1.27e+00(2.14e-01)-	1.17e+00(1.35e-01)-	1.07e+00(2.78e-02)-	1.05e+00(3.56e-02)
STOP8	2.42e+00(2.19e+00)-	6.74e+00(3.75e+00)-	2.90e+00(2.11e+00)-	1.27e+00(6.42e-01)
STOP9	1.67e+03(7.13e+02)-	4.11e+03(2.01e+03)-	1.72e+03(7.05e+02)-	1.37e+03(4.68e+02)
STOP10	1.15e+01(7.44e+00)-	1.08e+02(9.12e+01)-	1.43e+01(1.22e+01)-	6.04e+00(2.89e+00)
STOP11	5.66e+00(1.04e+00)-	9.91e+00(1.64e+00)-	5.51e+00(9.49e-01)-	5.04e+00(8.41e-01)
STOP12	1.90e+00(4.25e-01)-	2.73e+00(7.82e-01)-	1.50e+00(2.11e-01)-	1.38e+00(1.30e-01)
+/-/~	1/10/1	1/10/1	2/9/1	\

“+” (or “-”) indicates that the results of the corresponding variant are better (or worse) than that of ESTOA-FCM, and “~” indicates that they obtain similar performance.

Table 3.7: Mean objective values and standard deviations obtained by ESTOA-FCM and its variants.

Problem	Variant-IV	Variant-V	Variant-VI	ESTOA-FCM
STOP1	2.04e+02(1.19e+02)~	1.87e+02(1.09e+02)~	2.17e+02(1.49e+02)~	1.67e+02(9.52e+01)
STOP2	4.22e+03(2.07e+03)-	3.54e+03(1.83e+03)-	3.19e+03(1.45e+03)~	2.71e+03(1.01e+03)
STOP3	2.20e+01(9.43e+00)+	2.17e+01(7.20e+00)+	2.03e+01(6.74e+00)+	2.57e+01(8.81e+00)
STOP4	1.49e+00(1.05e+00)-	4.39e-01(1.73e-01)~	5.78e-01(2.79e-01)-	3.79e-01(1.51e-01)
STOP5	6.97e+00(1.20e+00)-	3.98e+00(6.28e-01)~	5.27e+00(7.48e-01)-	3.80e+00(6.08e-01)
STOP6	1.48e+02(2.19e+01)-	8.06e+01(1.66e+01)-	6.86e+01(1.49e+01)~	6.73e+01(1.28e+01)
STOP7	1.20e+00(1.25e-01)-	1.13e+00(9.53e-02)-	1.19e+00(1.06e-01)-	1.05e+00(3.56e-02)
STOP8	6.26e+00(4.04e+00)-	3.11e+00(2.55e+00)-	2.91e+00(1.83e+00)-	1.27e+00(6.42e-01)
STOP9	4.44e+03(1.67e+03)-	1.68e+03(7.48e+02)-	2.68e+03(1.26e+03)-	1.37e+03(4.68e+02)
STOP10	1.13e+02(8.75e+01)-	9.72e+00(8.94e+00)~	4.27e+01(3.72e+01)-	6.04e+00(2.89e+00)
STOP11	7.79e+00(1.32e+00)-	5.06e+00(7.26e-01)~	5.66e+00(9.76e-01)-	5.04e+00(8.41e-01)
STOP12	2.83e+00(7.22e-01)-	1.61e+00(3.29e-01)-	1.43e+00(2.02e-01)~	1.38e+00(1.30e-01)
+/-/~	1/10/1	1/6/5	1/7/4	\

“+” (or “-”) indicates that the results of the corresponding variant are better (or worse) than that of ESTOA-FCM, and “~” indicates that they obtain similar performance.

3.4.3 Ablation Experiments

The ablation experiments are conducted on STOP1-STOP12 by comparing ESTOA-FCM to its various variants. The detailed numerical results are provided in Tab. 3.6 and Tab. 3.7.

1) Effectiveness of Training Data Construction

To validate its effectiveness in generating labelled training data, two different variants (i.e., Variant-I and Variant-II) are designed. Variant-I uses positive samples to construct the initial training data, while Variant-II considers negative samples to form the initial training data. It can be observed from Tab. 3.6 that ESTOA-FCM significantly outperforms Variant-I and Variant-II on most test problems (i.e., 10 and 10 out of 12 test problems). The comparison results show the effectiveness of training data construction in ESTOA-FCM.

2) Effectiveness of Solution Usefulness Measurement

One variant (Variant-III) without solution usefulness measurement is used for performance comparison. Here, one source task to provide one optimized solution for transfer is randomly selected from all available source tasks. As shown in Tab. 3.6, ESTOA-FCM outperforms Variant-III on most test problems (i.e., 9 out of 12 test problems), which demonstrates that the use of the measurement method can effectively measure the solution usefulness of source tasks.

3) Effectiveness of Solution Selection for Transfer

Here, two different variants (i.e., Variant-IV and Variant-V) are designed for perfor-

mance comparison. For Variant-IV, the source task to provide one optimized solution for transfer is randomly selected from the negative class. As shown in Tab. 3.7, ESTOA-FCM achieves better results than Variant-IV on 10 out of 12 test problems, validating that source tasks with a positive label have higher solution usefulness than those in the negative class. Besides, Variant-IV selects one optimized solution of the source task with the minimal membership degree in the positive class. It is observed that ESTOA-FCM outperforms Variant-IV on 6 out of 12 test problems while it is only beat by Variant-IV on 1 case. On other test problems, they show similar results. The comparison results further validate the effectiveness of selecting the source task with maximal membership degree in the positive class.

4) Effectiveness of Training Data Update

To validate its effectiveness, one variant (Variant-VI) without training data update is used as a compared algorithm. As shown in Tab. 3.7, ESTOA-FCM achieves significantly better results than Variant-VI on most test cases (i.e., 7 out of 12 test problems), while it is outperformed by Variant-VI on 1 case. The performance improvements on most test problems show the effectiveness of training data update in ESTOA-FCM.

3.4.4 Parameter Sensitive Analysis

1) The Effect of α

To study the impact of α in the progressional representation, the comparisons of ESTOA-FCM using the progressional representation with different values of α from {0, 0.2, 0.4, 0.6, 0.8, 1} are done on STOP1-STOP12. The mean objective values and standard deviations are listed in Tab. 3.8. Compared to ESTOA-FCM with other val-

ues of α that are smaller than 1, i.e., α = 0, 0.4, 0.6, and 0.8, ESTOA-FCM with α = 0.2 shows similar performance on most test problems (i.e., 10, 9, 11, and 9 cases out of 12 test problems). However, it can be observed that the significant performance deterioration of ESTOA-FCM with α = 1. In particular, ESTOA-FCM with α = 0.2 achieves significantly better performance on 10 test problems when compared to that with α = 1. In addition, there is only one case on which ESTOA-FCM with α = 1 is better than ESTOA-FCM with α = 0.2. The above comparison results show that the population at the current generation plays a critical role in using the progressional representation to estimate the population distribution. In summary, setting α to a value less than 1 is suggested. Thus, α is set to 0.2.

2) The Effect of K

To study the impact of the number of nearest neighbours in FKNN, the comparisons of ESTOA-FCM using FKNN with different values of K from {1, 3, 5, 7, 9} are done on STOP1-STOP12. The mean objective values and standard deviations are listed in Tab. 3.9. It can be observed that ESTOA-FCM with K = 5 performs better than that with K = 1 on most test problems (i.e., 8 out of 12 test problems), and achieves similar results on 3 cases. In addition, compared to ESTOA-FCM with K = 3, 7, and 9, ESTOA-FCM with K = 5 achieves similar results on most test cases, i.e., 8, 12, and 10 cases out of 12 test cases, respectively. Meanwhile, there is no performance degradation on other test problems. The above comparison results show that ESTOA-FCM with a smaller value of K, i.e., K = 1, will cause the performance degradation of ESTOA-FCM on some test problems. In addition, the performance of ESTOA-FCM on most test problems is not very sensitive to FKNN with a larger value of K. Thus, setting K to 5 is suggested in this study.

Table 3.8: Mean objective values and standard deviations obtained by ESTOA-FCM with different values of α.

Problem	α = 0.0	α = 0.4	α = 0.6	α = 0.8	α = 1.0	α = 0.2
STOP1	1.62e+02(1.11e+02)~	1.96e+02(1.30e+02)~	1.80e+02(9.64e+01)~	1.63e+02(1.17e+02)~	1.74e+02(1.17e+02)~	1.67e+02(9.52e+01)
STOP2	2.81e+03(1.13e+03)~	3.19e+03(1.16e+03)-	3.00e+03(1.09e+03)~	2.87e+03(9.64e+02)~	3.94e+03(2.27e+03)-	2.71e+03(1.01e+03)
STOP3	2.36e+01(9.15e+00)~	2.45e+01(9.45e+00)~	2.39e+01(8.56e+00)~	2.36e+01(7.91e+00)~	2.08e+01(8.34e+00)+	2.57e+01(8.81e+00)
STOP4	3.49e-01(1.55e-01)~	4.58e-01(1.97e-01)-	6.15e-01(2.94e-01)-	4.95e-01(2.49e-01)-	2.93e+00(2.72e+00)-	3.79e-01(1.51e-01)
STOP5	3.66e+00(5.20e-01)~	3.85e+00(6.21e-01)~	3.88e+00(5.11e-01)~	3.67e+00(5.43e-01)~	8.40e+00(9.62e-01)-	3.80e+00(6.08e-01)
STOP6	8.21e+01(1.89e+01)-	8.22e+01(2.12e+01)-	7.14e+01(1.97e+01)~	8.39e+01(2.59e+01)-	1.78e+02(1.98e+01)-	6.73e+01(1.28e+01)
STOP7	1.08e+00(3.96e-02)-	1.06e+00(6.55e-02)~	1.06e+00(3.19e-02)~	1.07e+00(2.26e-02)-	1.57e+00(2.24e-01)-	1.05e+00(3.56e-02)
STOP8	1.15e+00(5.28e-01)~	1.39e+00(7.10e-01)~	1.46e+00(7.38e-01)~	1.51e+00(8.02e-01)~	1.34e+01(8.54e+00)-	1.27e+00(6.42e-01)
STOP9	1.27e+03(4.88e+02)~	1.43e+03(6.54e+02)~	1.40e+03(5.06e+02)~	1.26e+03(5.90e+02)~	4.32e+03(2.19e+03)-	1.37e+03(4.68e+02)
STOP10	7.01e+00(5.30e+00)~	5.64e+00(2.84e+00)~	5.25e+00(2.64e+00)~	5.64e+00(3.12e+00)~	8.98e+01(6.79e+01)-	6.04e+00(2.89e+00)
STOP11	5.00e+00(8.22e-01)~	5.20e+00(9.10e-01)~	5.26e+00(8.41e-01)~	4.94e+00(9.80e-01)~	9.38e+00(1.33e+00)-	5.04e+00(8.41e-01)
STOP12	1.35e+00(1.28e-01)~	1.41e+00(1.58e-01)~	1.41e+00(1.29e-01)~	1.40e+00(1.91e-01)~	2.85e+00(7.06e-01)-	1.38e+00(1.30e-01)
+/-/~	0/2/10	0/3/9	0/1/11	0/3/9	1/10/1	\

"+" (or "-") indicates that the results of ESTOA-FCM with the corresponding parameter are better (or worse) than that of ESTOA-FCM with the suggested parameter, and "~" indicates that they obtain similar performance.

Table 3.9: Mean objective values and standard deviations obtained by ESTOA-FCM with different values of K.

Problem	K = 1	K = 3	K = 7	K = 9	K = 5
STOP1	1.38e+02(7.46e+01)~	1.58e+02(9.55e+01)~	1.75e+02(1.30e+02)~	1.40e+02(8.76e+01)~	1.67e+02(9.52e+01)
STOP2	3.36e+03(1.35e+03)-	3.10e+03(1.31e+03)~	2.94e+03(1.19e+03)~	2.95e+03(1.65e+03)~	2.71e+03(1.01e+03)
STOP3	2.03e+01(7.35e+00)+	2.32e+01(8.05e+00)~	2.62e+01(9.60e+00)~	2.70e+01(1.03e+01)~	2.57e+01(8.81e+00)
STOP4	5.20e-01(2.93e-01)-	4.55e-01(2.10e-01)-	4.33e-01(1.87e-01)~	4.58e-01(2.03e-01)-	3.79e-01(1.51e-01)
STOP5	4.19e+00(6.49e-01)-	4.15e+00(6.47e-01)-	3.81e+00(5.85e-01)~	3.85e+00(6.29e-01)~	3.80e+00(6.08e-01)
STOP6	1.08e+02(2.10e+01)-	8.03e+01(1.73e+01)-	7.33e+01(1.67e+01)~	6.50e+01(1.52e+01)~	6.73e+01(1.28e+01)
STOP7	1.09e+00(3.07e-02)-	1.07e+00(3.42e-02)-	1.06e+00(3.19e-02)~	1.07e+00(2.76e-02)-	1.05e+00(3.56e-02)
STOP8	1.67e+00(1.05e+00)-	1.23e+00(6.03e-01)~	1.27e+00(7.26e-01)~	1.40e+00(8.07e-01)~	1.27e+00(6.42e-01)
STOP9	1.36e+03(4.69e+02)~	1.48e+03(6.68e+02)~	1.31e+03(5.01e+02)~	1.34e+03(5.45e+02)~	1.37e+03(4.68e+02)
STOP10	8.62e+00(5.24e+00)-	6.82e+00(4.78e+00)~	7.23e+00(4.76e+00)~	6.31e+00(4.39e+00)~	6.04e+00(2.89e+00)
STOP11	6.13e+00(8.93e-01)-	5.06e+00(7.87e-01)~	5.11e+00(7.76e-01)~	4.99e+00(8.63e-01)~	5.04e+00(8.41e-01)
STOP12	1.41e+00(1.33e-01)~	1.42e+00(1.70e-01)~	1.40e+00(1.65e-01)~	1.38e+00(1.46e-01)~	1.38e+00(1.30e-01)
+/-/~	1/8/3	0/4/8	0/0/12	0/2/10	\

"+" (or "-") indicates that the results of ESTOA-FCM with the corresponding parameter are better (or worse) than that of ESTOA-FCM with the suggested parameter, and "~" indicates that they obtain similar performance.

3) The Effect of c

To study the impact of the number of transferred solutions at each transferable generation, the comparisons of ESTOA-FCM with different values of c from {1, 5, 10, 20}

are done on STOP1-STOP12. The mean objective values and standard deviations are listed in Tab. 3.10. It is observed that ESTOA-FCM with $c = 1$ achieves significantly better performance on most test problems (i.e., 11, 11, and 12 out of 12 test problems) when compared to that with other values of c. The comparison results show that selecting more transferred solutions from one source task will result in performance degradation. Thus, setting c to 1 is suggested in this study.

Table 3.10: Mean objective values and standard deviations obtained by ESTOA-FCM with different values of c.

Problem	$c = 5$	$c = 10$	$c = 20$	$c = 1$
STOP1	5.28e+02(3.50e+02)-	1.09e+03(9.72e+02)-	2.98e+03(2.48e+03)-	1.67e+02(9.52e+01)
STOP2	7.90e+03(3.91e+03)-	1.36e+04(1.05e+04)-	3.19e+04(3.60e+04)-	2.71e+03(1.01e+03)
STOP3	4.76e+01(2.44e+01)-	8.93e+01(3.97e+01)-	1.59e+02(8.42e+01)-	2.57e+01(8.81e+00)
STOP4	4.63e-01(2.03e-01)-	5.85e-01(2.75e-01)-	6.87e-01(3.19e-01)-	3.79e-01(1.51e-01)
STOP5	4.20e+00(7.59e-01)-	4.42e+00(7.39e-01)-	4.67e+00(7.13e-01)-	3.80e+00(6.08e-01)
STOP6	7.22e+01(2.43e+01)~	7.93e+01(3.16e+01)~	9.05e+01(3.52e+01)-	6.73e+01(1.28e+01)
STOP7	1.26e+00(1.59e-01)-	1.35e+00(3.30e-01)-	1.58e+00(4.62e-01)-	1.05e+00(3.56e-02)
STOP8	5.34e+00(3.36e+00)-	8.84e+00(7.03e+00)-	1.47e+01(1.46e+01)-	1.27e+00(6.42e-01)
STOP9	8.93e+03(5.14e+03)-	1.36e+04(8.65e+03)-	3.50e+04(2.12e+04)-	1.37e+03(4.68e+02)
STOP10	9.19e+01(1.13e+02)-	1.04e+02(1.54e+02)-	3.37e+02(4.14e+02)-	6.04e+00(2.89e+00)
STOP11	7.26e+00(1.89e+00)-	7.83e+00(1.74e+00)-	9.60e+00(2.59e+00)-	5.04e+00(8.41e-01)
STOP12	3.79e+00(2.31e+00)-	3.81e+00(1.56e+00)-	8.21e+00(6.69e+00)-	1.38e+00(1.30e-01)
+/-/~	0/11/1	0/11/1	0/12/0	\

"+" (or "-") indicates that the results of ESTOA-FCM with the corresponding parameter are better (or worse) than that of ESTOA-FCM with the suggested parameter, and "~" indicates that they obtain similar performance.

3.4.5 Computational Complexity Analysis

In terms of four distance metrics (i.e., ED, WD, KLD, and MMD), their computational complexities are $O(Nd)$, $O(Nd)$, $O(N^3d^3)$, and $O(N^2d)$, respectively. Here, N and d are the population size and the problem's dimensionality, respectively. In the proposed method, FKNN does not introduce additional time complexity during the model training phase, as it operates directly on the training data without the

need for an explicit parameter estimation process. Here, the computational complexity of FKNN for predicting the result of each test sample is $O(Ndn)$, where n is the number of the training samples. Fig. 3.5 presents the total running time of each algorithm for solving STOP1 to STOP12. All compared algorithms are implemented in MATLAB R2020b and run on a computer with AMD Ryzen 9 5900X 12-Core Processor. It is observed that the running speed of ESTOA-FCM is slightly slower than that of ESTOA-ED and ESTOA-WD, while it can significantly outperform the running speed of ESTOA-KLD. The comparative analysis thus indicates that introducing FKNN into ESTOA-FCM does not lead to a substantial increase in computational overhead.

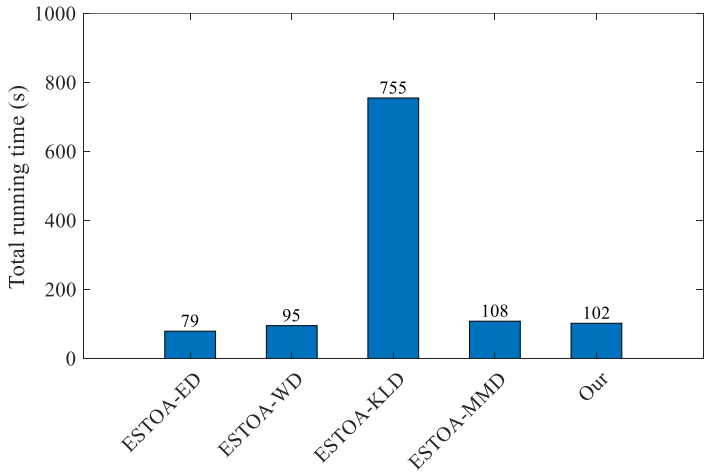


Figure 3.5: Comparison of the total running times of all compared algorithms on STOP1-STOP12.

3.4.6 Practical Case Study

1) Problem Definition

To further analyse the effectiveness of ESTOA-FCM in solving practical optimization

case, the planar kinematic arm [98] is used in the comparative experiments. The arm consists of d links (all the links have the same length L), which are connected by d joints (all joints have the same angle limits α_{max}). The objective of the arm is to optimize the angle of each joint $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ to make the tip position of the arm $\mathbf{p}^d$ as close as possible to a predefined target position γ in the plane, which can be formulated as follows:

$$\min_{\alpha} f(\boldsymbol{\alpha}) = ||\mathbf{p}^d - \gamma||_2 , \tag{3.26}$$

where $\alpha_i \in [-\alpha_{max}, \alpha_{max}]$ for $i = 1, \ldots, d$. More details of the kinematics used to calculate $\mathbf{p}^d$ can be found in [98]. In practice, the target position in the plane usually varies, which leads to a change in the optimal angles of the arm. It is expected to utilize the search experience from previously-solved tasks to speed up the search for the optimal solution of a new task. Thus, different optimization tasks can be constructed by setting their target positions to different coordinate points in the normalized plane $[0, 1]^2$ for the planar kinematic arm. Here, L and α_{max} are set to $1/d$ and $0.8/d$, respectively, which make the arm has the same total length (1 meter) and reaching abilities regardless of the dimensionality. First, the test instances with the same dimensionality are constructed. Specifically, 50 optimization tasks are generated by randomly sampling their target positions in $[0, 1]^2$. One task is used as the target task, while the rest will serve as source tasks to form the knowledge base. Moreover, 50 test instances with d = 150 are generated in the same way. For each of these test instances, its target task and all source tasks in the corresponding knowledge base have the same dimensionality. Additionally, the test instances with different dimensionalities are constructed. Here, 50 optimization tasks are generated by randomly sampling their target positions in $[0, 1]^2$ and setting their dimensionalities to the integers in the range of [50, 100]. Note that the solutions of source tasks would be truncated or pad-

ded with zeros to make them have the same dimension as the target task. Each of the 50 tasks is regarded as the target task and its knowledge base consists of the remaining tasks.

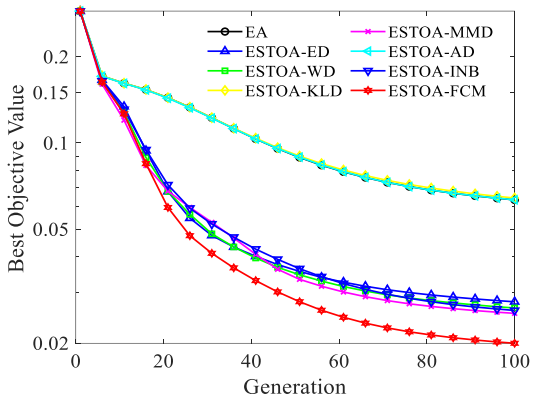
Table 3.11: Summarized results of ESTOA-FCM and all competitors.

Algorithm Comparison	Test instances (+/-/~)		
	d = 100	d = 150	$d \in$ [50, 100]
EA vs ESTOA-FCM	2/46/2	0/45/5	0/46/4
ESTOA-ED vs ESTOA-FCM	0/38/12	0/39/11	6/31/13
ESTOA-WD vs ESTOA-FCM	0/36/14	1/36/13	4/31/15
ESTOA-KLD vs ESTOA-FCM	3/42/5	0/45/5	3/39/8
ESTOA-MMD vs ESTOA-FCM	0/37/13	1/36/13	6/29/15
ESTOA-AD vs ESTOA-FCM	3/44/3	1/45/4	0/46/4
ESTOA-INB vs ESTOA-FCM	2/35/13	0/31/19	5/23/22

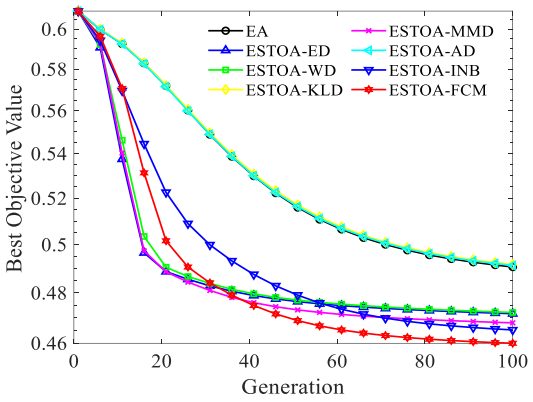
"+" (or "-") indicates the number of test problems on which the competitor is better (or worse) than ESTOA-FCM, and "~" indicates the number of test problems on which they obtain similar performance.

2) Comparison Results

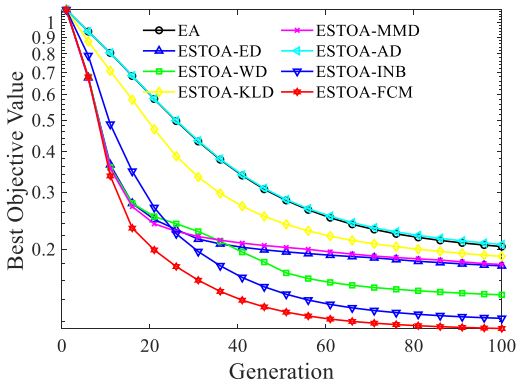
The summarized compared results are given in Tab. 3.11. It can be observed that ESTOA-FCM can significantly outperform its competitors including EA, ESTOA-ED, ESTOA-WD, ESTOA-KLD, ESTOA-MMD, ESTOA-AD, and ESTOA-INB on most test instances. Furthermore, the convergence curves of all compared algorithms on three representative test instances with d = 100, d = 150, and $d \in$ [50, 100] are provided in Fig. 3.6. It can be clearly observed that ESTOA-FCM achieves the best convergence performance among all compared algorithms on the three practical test instances. The above comparison results further demonstrate the effectiveness and superiority of ESTOA-FCM in solving practical planar kinematic arm problems when compared with existing ESTO algorithms.



(a) $d = 100$



(b) $d = 150$



(c) $d \in [50, 100]$

Figure 3.6: Convergence curves on three representative test problems with different dimensions.

3.5 Conclusion

This chapter has proposed a fuzzy classifier-assisted solution transfer method for ES-TO. Different from existing solution transfer methods, the proposed method learns a fuzzy classifier to measure the solution usefulness of source tasks, which aims to select useful solutions to accelerate the optimization of the target task. The training data is first constructed based on the evaluated solutions of candidate source tasks in the

knowledge base. As the optimization process proceeds, the solutions of the source and target tasks are collected to generate the test samples. After that, the solution usefulness of source tasks can be estimated based on the predicted results of the fuzzy classifier on the test samples. In this way, useful source solutions can be selected from source tasks to accelerate the optimization process of the target task. Additionally, to further improve the accuracy of the fuzzy classifier, the training data is dynamically updated with the test samples obtained during the evolutionary search process. Extensive experiments have been done on two benchmark suites and one practical case to validate the effectiveness and superiority of the proposed method when compared to existing ESTO algorithms.

Chapter 4

Ensemble Method of Domain Adaptation for Adaptively Deciding How to Transfer in EMT

4.1 Introduction

As domain adaptation (DA) methods have shown effectiveness for enhancing knowledge transferability from the source domain to the target domain in traditional transfer learning methods [99], some recent studies of EMT also suggest using DA methods to further improve knowledge transferability via a suitable transformation between distinct tasks, where the knowledge from one task called the source task is

transferred to another task called the target task. Thus, each task in EMT can be treated as either a source task or a target task, and the search space of the source/target task is called the source/target domain. In MFEA-DV [100], the elite solution of the target task is translated along the difference vector provided by the source task, while the solutions of all tasks in G-MFEA [74] are mapped to a common central location for knowledge transfer. To facilitate high ordinal correlation among different search spaces, a linear transformation is learnt in LDA-MFEA [66] to transfer solutions across tasks, while the denoising autoencoder (AE) method was used in [48] to obtain the solution mapping relationship among different tasks. In addition, an affine transformation was proposed in AT-MFEA [67] to avoid chaotic matching by considering the fitness rank correlation and the topological consistency for learning the mapping relationship, while a kernelized AE [68] was used to capture the nonlinearity between different tasks. In addition, a two-layer feedforward neural network [69] was used to learn the aligned solution representations through two-layer transformations, while subspace alignment methods were suggested in [70], [71] to run the transformation in the reduced variable subspaces of source and target tasks.

However, the above DA methods often show certain specific biases when learning the mappings or transformations from the source task to the target task, which will be experimentally studied in the following subsection 4.2.2. Thus, it is a natural idea to use an ensemble method to combine multiple complementary DA methods for knowledge transfer, as inspired by the effectiveness of ensemble methods to combine multiple algorithms [101], [102], [103], selection criteria [104], [105], evolutionary operators [106], [107], resource allocation strategies [108], constraint handling techniques [109], and parameter settings [110] for solving a diverse set of optimization problems. However, to the best of our knowledge, few studies have paid attention to the complementarity of

DA methods in knowledge transfer or considered combining their advantages in EMT. To fill this research gap, this chapter proposes a domain adaptation ensemble (DAE) method for knowledge transfer, in which the efficacy and diversity of DA methods are considered in different situations in EMT. First, a hierarchical clustering method (HCM) [111], [112] is used to divide the solutions of each task into multiple clusters. Then, once two parent solutions are selected for knowledge transfer across tasks, the efficacy and diversity of DA methods are considered in two cases by checking the solutions within the same cluster: if none of these solutions have been transferred before, the efficacy of domain adaptation methods is considered by using roulette wheel selection (RWS) based on the obtained performance improvements of these methods in the evolutionary optimization process; otherwise, the diversity of DA methods is emphasized by randomly selecting one of them for knowledge transfer. Different from existing ensemble methods, it is a first attempt to consider both the efficacy and diversity of DA methods for knowledge transfer in EMT, which could improve knowledge transferability across tasks. To summarize, the main contributions of this study are summarized as follows.

1) This study proposes a DAE method to combine the advantages of multiple complementary DA methods for knowledge transfer in EMT. To select one favourable DA method for use, the efficacy and diversity of DA methods are considered in DAE.

2) This study presents the implementation of incorporating the proposed ensemble method into an EMT framework. The experimental results validate that incorporating into DAE method into three competitive EMT algorithms can significantly improve their performance for solving different multitasking test problems. Moreover, a canonical EMT algorithm enhanced by DAE(called MFEA-DAE) outperforms five recent EMT algorithms on most cases of the multitasking test problems used, and the effectiveness of DAE is also validated on a practical case.

4.2 Background and Motivation

4.2.1 Domain Adaption for Evolutionary Multitasking

The mapping parameters of AE [48], KAE [68] and AT [67] are expressed in closed-form solution, which can be computed by a finite number of operators on the populations of source and target tasks without training. Thus, they can be seamlessly used in any EMT framework as an independent module to transfer solutions across tasks. Here, their learning mechanisms to construct solution mapping from the source task to the target task are briefly introduced as follows:

1) Denoising AE [48]

Considering source task T_s and target task T_t, their solution sets are represented by $\mathbf{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_N\}$ and $\mathbf{Q} = \{\mathbf{q}_1, \dots, \mathbf{q}_N\}$, respectively, where N is the number of solutions and the solutions in $\mathbf{P}$ or $\mathbf{Q}$ with smaller dimension are padded with zeros to make them have equal dimensionality. To have a high ordinal correlation in the mapping construction, the solutions in $\mathbf{P}$ and $\mathbf{Q}$ are sorted in the same way. As suggested in [48], sorting can be performed based on the objective values of solutions for SOPs or the nondominated rankings and crowding distances of solutions for MOPs. The mapping $\mathbf{M}$ for transferring solutions from T_s to T_t can be learnt by minimizing the squared reconstruction loss of the corrupted input, which can be formulated by

$$L_{sq}(\mathbf{M}) = \frac{1}{2N}\sum_{i=1}^{N} ||\mathbf{q}_i - \mathbf{M}\mathbf{p}_i||^2 , \tag{4.1}$$

To obtain the matrix form of Eq. (4.1), a constant feature is added to the input, i.e., $\mathbf{p}_i = [\mathbf{p}_i; 1]$, $\mathbf{q}_i = [\mathbf{q}_i; 1]$, and a bias $\mathbf{c}$ is incorporated within $\mathbf{M}$, i.e., $\mathbf{M} = [\mathbf{M}, \mathbf{c}]$.

Then, Eq. (4.1) can be written

$$L_{sq}(\mathbf{M}) = \frac{1}{2N} tr[(\mathbf{Q} - \mathbf{MP})^T(\mathbf{Q} - \mathbf{MP})] , \tag{4.2}$$

where $tr[\cdot]$ is the trace of a matrix. As the closed-form solution for ordinary least squares [113], $\mathbf{M}$ is expressed by

$$\mathbf{M} = (\mathbf{QP}^T)(\mathbf{PP}^T)^{-1} . \tag{4.3}$$

Given a solution $\mathbf{p}_s$ on T_s and the learnt mapping $\mathbf{M}$, its transferred solution $\mathbf{p}_t$ on T_t can be generated as follows:

$$\mathbf{p}_t = \mathbf{M}\mathbf{p}_s . \tag{4.4}$$

2) Kernelized AE (KAE) [68]

In KAE, the learning of $\mathbf{M}$ based on the kernel function is to capture the nonlinearity between $\mathbf{P}$ and $\mathbf{Q}$. Mapping $\mathbf{P}$ to a reproduced kernel Hilbert space $\mathcal{H}$ by a non-linear mapping function φ, the squared reconstruction loss in Eq. (4.2) is revised as follows:

$$L_{sq}(\mathbf{M}) = \frac{1}{2N} tr[(\mathbf{Q} - \mathbf{M}\varphi(\mathbf{P}))^T(\mathbf{Q} - \mathbf{M}\varphi(\mathbf{P}))] . \tag{4.5}$$

According to [114], the linear mapping $\mathbf{M}$ in $\mathcal{H}$ can be represented as a linear combination of the data points $\varphi(\mathbf{P})$ in $\mathcal{H}$, i.e., $\mathbf{M} = \mathbf{M}_k\varphi(\mathbf{P})^T$. Thus, we have $\mathbf{M}\varphi(\mathbf{P}) = \mathbf{M}_k\varphi(\mathbf{P})^T\varphi(\mathbf{P})$. The kernel matrix is denoted as $\mathbf{K}(\mathbf{P},\mathbf{P}) = \varphi(\mathbf{P})^T\varphi(\mathbf{P})$, where the (i,j)-th element of $\mathbf{K}(\mathbf{P},\mathbf{P})$ is represented by the kernel function $\kappa(\mathbf{p}_i,\mathbf{p}_j)$. Then, Eq. (4.5) is reformulated by

$$L_{sq}(\mathbf{M}_k) = \frac{1}{2N} tr\left[\left(\mathbf{Q} - \mathbf{M}_k\mathbf{K}(\mathbf{P},\mathbf{P})\right)^T\left(\mathbf{Q} - \mathbf{M}_k\mathbf{K}(\mathbf{P},\mathbf{P})\right)\right]. \quad (4.6)$$

Thus, a closed-form solution can be deduced by

$$\mathbf{M}_k = \mathbf{Q}\mathbf{K}(\mathbf{P},\mathbf{P})^T(\mathbf{K}(\mathbf{P},\mathbf{P})\mathbf{K}(\mathbf{P},\mathbf{P})^T)^{-1}. \quad (4.7)$$

In terms of $\mathbf{p}_s$ on T_s and the learnt mapping $\mathbf{M}_k$, its transferred solution $\mathbf{p}_t$ on T_t is generated as follows:

$$\mathbf{p}_t = \mathbf{M}\varphi(\mathbf{p}_s) = \mathbf{M}_k\varphi(\mathbf{P})^T\varphi(\mathbf{p}_s) = \mathbf{M}_k\mathbf{K}(\mathbf{P},\mathbf{p}_s), \quad (4.8)$$

where $\kappa(\mathbf{p}_i,\mathbf{p}_s)$ is the i-th element of $\mathbf{K}(\mathbf{P},\mathbf{p}_s)$.

3) Affine Transformation (AT) [67]

Given the source task T_s and the target task T_t, the optimal mapping ϕ^* that transfers solutions from T_s to T_t can be obtained by minimizing a rank loss function, which is formulated as follows:

$$\phi^* = \min_{\phi_a\in\Phi} \int_{\mathbf{x}} \|\Re[F_s(\mathbf{x})] - \Re[F_t(\tilde{\mathbf{x}})]\|^2, \quad (4.9)$$

where $\mathbf{x}$ is a solution encoded in a unified space and its transferred solution $\tilde{\mathbf{x}}$ is obtained by applying the linear transformation ϕ_a on $\mathbf{x}$, which is expressed as follows:

$$\tilde{\mathbf{x}} = \phi_a(\mathbf{x};\boldsymbol{\theta}) = \mathbf{x}\mathbf{A} + \mathbf{b}, \quad (4.10)$$

where $\boldsymbol{\theta} = [\mathbf{A},\mathbf{b}]$ denotes the parameters of affine transformation. Here, $\Re$ is a rank operation for converting the function values of solutions into ranks, in which $F_s(\mathbf{x})$ and $F_t(\tilde{\mathbf{x}})$ denote the function values of $\mathbf{x}$ on T_s and $\tilde{\mathbf{x}}$ on T_t, respective-

ly. As a surrogate of the objective values of the solutions in the evolutionary optimization process, the progressive representation model $\hat{p}(\mathbf{x})$ for a total of g generations is formulated by

$$\hat{p}(\mathbf{x}) = \sum_{k=1}^{g} \alpha^{g-k}(1-\alpha)\hat{p}_k(\mathbf{x}), \tag{4.11}$$

where $0 \leq \alpha \leq 1$ is the coefficient for adjusting the weights of local representation models $\hat{p}_1(\mathbf{x}), \ldots, \hat{p}_g(\mathbf{x})$. Here, $\hat{p}_k(\mathbf{x})$ is built based on a multivariate Gaussian distribution by calculating the mean $\boldsymbol{\mu}_k$ and covariance $\sum_k$ as follows:

$$\begin{cases} \boldsymbol{\mu}_k = \dfrac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^k \\ \sum_k = \dfrac{1}{N-1}\sum_{i=1}^{N} (\mathbf{x}_i^k - \boldsymbol{\mu}_k)(\mathbf{x}_i^k - \boldsymbol{\mu}_k)^T \end{cases} \tag{4.12}$$

where N is the number of solutions at the k-th generation and $\mathbf{x}_i^k$ denotes the k-th solution in the current population. Therefore, the parameters $\hat{\boldsymbol{\mu}}$ and $\hat{\sum}$ in $\hat{p}(\mathbf{x})$ can be calculated by

$$\begin{cases} \hat{\boldsymbol{\mu}} = (1-\alpha)\sum_{k=1}^{g} \alpha^{g-k}\boldsymbol{\mu}_k \\ \hat{\sum} = (1-\alpha)\sum_{k=1}^{g} \alpha^{g-k}\sum_k \end{cases}. \tag{4.13}$$

By introducing the affine transformation and representation model into the loss function, Eq. (4.9) can be reformulated by

$$\boldsymbol{\theta}^* = \min_{\boldsymbol{\theta}} \int_{\mathbf{x}} \left\| \Re[\hat{p}^s(\mathbf{x})] - \Re\left[\hat{p}^t\left(\phi_a(\mathbf{x};\boldsymbol{\theta})\right)\right] \right\|^2, \tag{4.14}$$

where $\hat{p}^s(\cdot)$, $\hat{p}^t(\cdot)$ are the Gaussian progressive representation models of T_s and T_t, respectively, which are expressed by

$$\begin{cases} \hat{p}^s(\mathbf{x}) = \dfrac{1}{(2\pi)^{n/2}|\sum_s|^{1/2}} \times \mathbb{E}\left(\mathbf{x}, \hat{\boldsymbol{\mu}}_s, \hat{\sum}_s\right) \\ \hat{p}^t(\tilde{\mathbf{x}}) = \dfrac{1}{(2\pi)^{n/2}|\sum_t|^{1/2}} \times \mathbb{E}(\mathbf{xA} + \mathbf{b}, \hat{\boldsymbol{\mu}}_t, \hat{\sum}_t) \end{cases} \tag{4.15}$$

where $\mathbb{E}\left(\mathbf{x}, \hat{\boldsymbol{\mu}}, \hat{\sum}\right) = \exp\left(-1/2(\mathbf{x} - \hat{\boldsymbol{\mu}})\hat{\sum}^{-1}(\mathbf{x} - \hat{\boldsymbol{\mu}})^T\right)$, the parameters $\hat{\boldsymbol{\mu}}_s$, $\hat{\sum}_s$ from the source representation model and $\hat{\boldsymbol{\mu}}_t$, $\hat{\sum}_t$ from the target representation model are calculated by Eq. (4.12) and Eq. (4.13), respectively.

To solve Eq. (4.14), the equivalent algebraic formulation is given as follows:

$$\hat{p}^s(\mathbf{x}) = \delta \times \hat{p}^t\left(\phi_a(\mathbf{x}; \boldsymbol{\theta})\right), \tag{4.16}$$

where δ is a multiplier. By completing the square, an analytical solution can be obtained as follows:

$$\begin{cases} \mathbf{A} = L_s {L_t}^{-1} \\ \mathbf{b} = \hat{\boldsymbol{\mu}}_t - \hat{\boldsymbol{\mu}}_s \mathbf{A} \end{cases} \tag{4.17}$$

where L_s and L_t can be deduced from $\hat{\sum}_s^{-1} = L_s {L_s}^{-1}$ and $\hat{\sum}_t^{-1} = L_t {L_t}^{-1}$ through the Cholesky decomposition. According to Eq. (4.10), the transferred solution $\mathbf{p}_t$ on T_t can be derived from $\mathbf{p}_s$ on T_s by

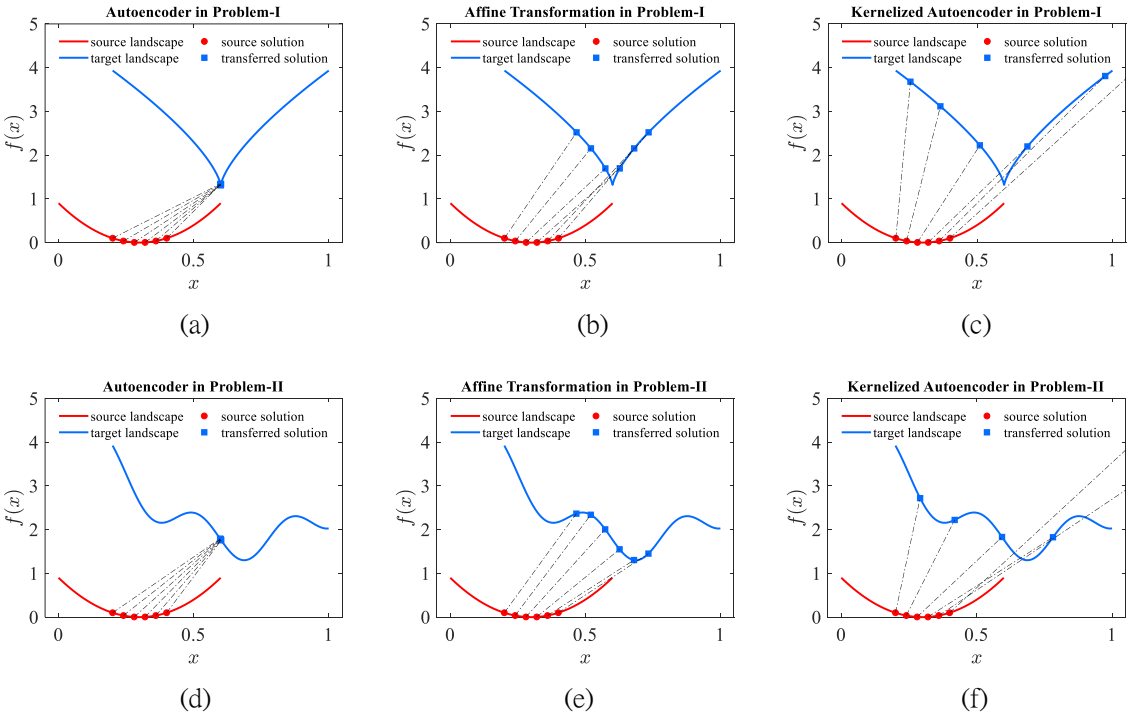
$$\mathbf{p}_t = \phi_a(\mathbf{p}_s; \boldsymbol{\theta}) = \mathbf{p}_s \mathbf{A} + \mathbf{b}. \tag{4.18}$$

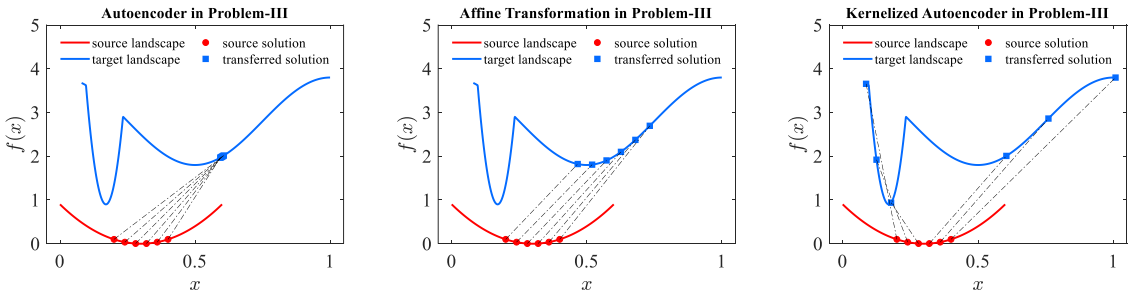
4.2.2 Study on the Effectiveness of DA Methods

1) Mapping Bias Comparison

DA methods can enhance the solution transferability from the source task to the target task by learning the mapping relationship between the solutions of two distinct tasks

that have different function landscapes and global optima. However, different DA methods often have a specific bias in their mapping behaviours, which show different mapping characteristics in transferring solutions from the source task to the target task in EMT. To study the solution mapping behaviours of AE, AT and KAE, three adapted multitasking optimization problems (Problem-I, Problem-II and Problem-III) from [96] are considered herein, where one 1-D minimization optimization function is set as the source task and three different 1-D minimization optimization functions are set as their target tasks. In the experiment, their mapping parameters are learnt from the same training data, including 400 sampled solutions, in which half of these solutions are randomly sampled in the search space of the source task with a boundary (0, 0.6) and the other half are uniformly sampled in the search space of the target task with a boundary (0.2, 1). The test data consist of 6 superior solutions of the source task, which are mapped from the source task to the target task to observe the mapping behaviours of the AE, AT, and KAE.





(g) (h) (i)

Figure 4.1: Solution mapping behaviours of autoencoder, affine transformation, and kernelized autoencoder in three test problems.

As observed in Figs. 4.1(a), (d), and (g), AE tends to map the solutions of the source task into the central region of the search space of the target task, which shows the strong exploitation bias in the process of knowledge transfer. This phenomenon is attributed to the chaotic mapping of the training samples from the source task to the target tasks. Specifically, the training samples are first sorted based on their fitness values, and then they are pairwise matched to learn the mapping relationship from the source task to the target task. As experimentally elaborated in [67], a great number of intersections will exist in these pairwise matchings, which will impede the detection of linear correlation from source solutions to target solutions. Thus, considering Problem-I whose global optimum of the target task is near the central region, the transferred solutions via AE have very high quality in Fig. 4.1(a), while its effectiveness cannot be guaranteed once the global optimum of the target task is far away from its central region, as shown in Figs. 4.1(d) and (g). Compared to AE, AT has a stronger exploration capability in the solution mapping behaviour, which can enable the transferred solutions to cover a relatively larger region in the search space of the target task, as shown in Figs. 4.1(b), (e) and (h). It is observed that the transferred solutions can effectively explore the global optimum of the target task of Problem-II in Fig. 4.1(e), although they fail to maintain superiority on Problem-I in Fig. 4.1(b) and Prob-

lem-III in Fig. 4.1(h). In addition, compared with AE and AT, KAE has the strongest exploration bias in mapping solutions from the source task to the target task, which leads the transferred solutions to cover a much wider region of the search space of the target task, as shown in Figs. 4.1(c), (f) and (i). This bias enables the transferred solutions to escape from the local optimum and provides the opportunity to search the global optimum of the target task of Problem-III in Fig. 4.1(i) but neglects the exploitation of the promising region to search for the global optima of the target tasks of Problem-I in Fig. 4.1(c) and Problem-II in Fig. 4.1(f). The significant distinctions among the mapping behaviours reveal their complementarity in transferring solutions from the source task to the target task on different multitasking test problems.

2) Multitasking Performance Comparison

To experimentally confirm the advantages of various DA methods in knowledge transfer, a common implicit EMT framework (MFEA [27]) is used as the basic solver for a fair comparison. To seamlessly incorporate these DA methods into MFEA, a generalized DA-based intertask crossover [67] is introduced to incorporate a general DA method into the crossover across different tasks. The pseudocode of DA-based intertask crossover is provided in Algorithm 4.1 with the inputs: two parent solutions $\mathbf{p}_a$ and $\mathbf{p}_b$ with their skill factors τ_a and τ_b, and the DA method (DA). In Line 1, the mapping parameters from task τ_a to τ_b and from task τ_b to τ_a are obtained based on the used DA. Then, $\mathbf{p}_a$ and $\mathbf{p}_b$ are transferred to their respective target tasks τ_b and τ_a by the corresponding mapping parameters in Lines 2-3, which then undergo crossover with the transferred solutions $\mathbf{p}_b'$ and $\mathbf{p}_a'$, respectively, in Lines 4-5. In this way, negative transfer can be effectively alleviated. Note that Algorithm 4.1 is triggered only when $\mathbf{p}_a$ and $\mathbf{p}_b$ have different skill factors (i.e., $\tau_a \neq \tau_b$) and a random number $rand$ is smaller than a pre-set probability rmp.

Algorithm 4.1 DA-Based Intertask Crossover

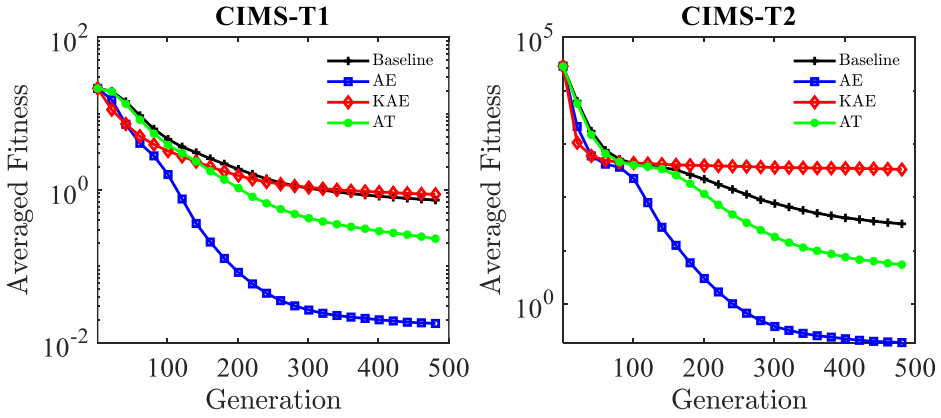
Input: $\mathbf{p}_a$ and $\mathbf{p}_b$: two parents with skill factors τ_a and τ_b
DA: one domain adaptation method
Output: Two offspring $\mathbf{c}_a$ and $\mathbf{c}_b$

1 Get mapping parameters between tasks τ_a and τ_b based on DA
2 $\mathbf{p}_a{}'$ ← Transfer $\mathbf{p}_a$ to task τ_b by the mapping from τ_a to τ_b
3 $\mathbf{p}_b{}'$ ← Transfer $\mathbf{p}_b$ to task τ_a by the mapping from τ_b to τ_a
4 $\mathbf{c}_a$ ← Perform crossover between $\mathbf{p}_a$ and $\mathbf{p}_b{}'$
5 $\mathbf{c}_b$ ← Perform crossover between $\mathbf{p}_b$ and $\mathbf{p}_a{}'$

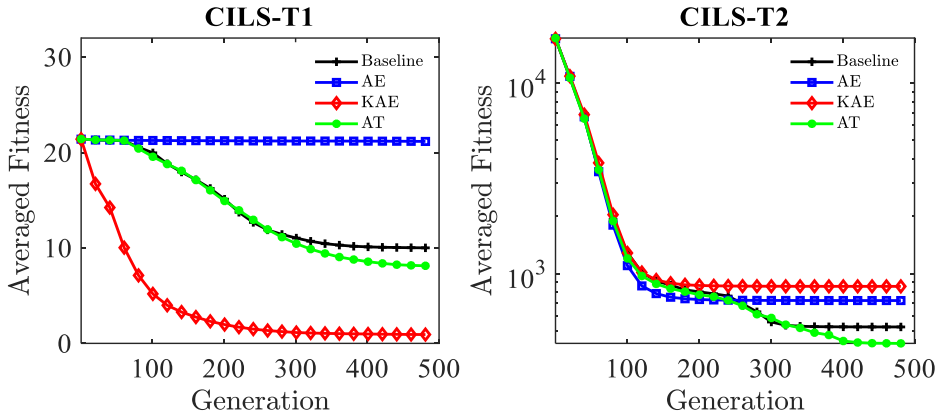
Replacing the original intertask crossover with Algorithm 1 in the MFEA, four different DA methods (AE, KAE, AT, and the baseline) can be seamlessly embedded into MFEA, which are realized by learning the mapping parameters using their particular mapping construction schemes in Line 1 of Algorithm 4.1. As suggested in [28]-[30], 100 solutions are sampled on source and target tasks to learn the mappings offline in AE and KAE, while the mapping parameters in AT are learnt based on the established progressive representation. The baseline can be seen as a special DA method to transfer solutions from the source task to the target task without any adaptation. The comparison experiments are conducted on two single-objective multitasking benchmark suites [50], [11]. More details of the benchmarks and parameter settings will be given later in subsection 4.4.1. The average convergence curves of the MFEA with the baseline, AE, KAE, and AT over 20 independent runs on four representative problems (CIMS, CILS, F1, and F2) are shown in Fig. 4.2, in which the x-axis and y-axis denote the generation number and the average objective value, respectively.

As observed in Fig. 4.2, different DA methods show their respective advantages on different test problems. In Fig. 4.2(a), AE achieves the best performance on the CIMS

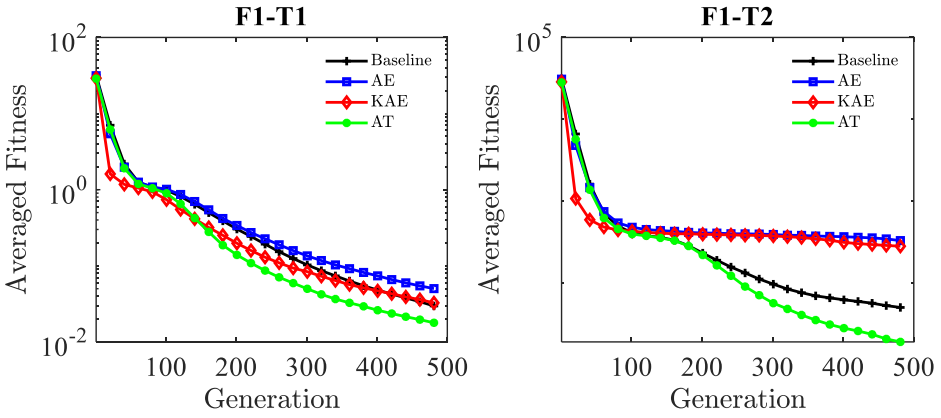
with higher similarity between two tasks. In this case, the transferred solutions through AE tend to be limited in a relatively small search region with good convergence performance for the target task, so that they can effectively guide the evolution of the population in the target task, which leads to a fast convergence speed for solving the CIMS. However, when solving the CILS with lower similarity between two tasks in Fig. 4.2(b), the KAE and AT obtain the best performance on task 1 and task 2, respectively, which indicates that their mapping biases are more effective. Similarly, considering F1 and F2, their global optima of two tasks are sampled from a uniform distribution, rather than located in the centre of the unified search space. As observed in Figs. 4.2(c) and (d), KAE and AT show faster convergence speeds than AE, which is attributed to the mapping biases that their transferred solutions can explore a wider range of promising search regions of the target task. In addition, it is observed that KAE shows a faster convergence speed than AT during the early evolutionary period, while it is outperformed by AT during the later evolutionary stage. This is because the bias of stronger exploration in the mapping of the KAE is more helpful in searching for the global optimum of the target task in the early stage, while it also leads to inefficient exploration for the target task in the later stage as the population gradually converges.



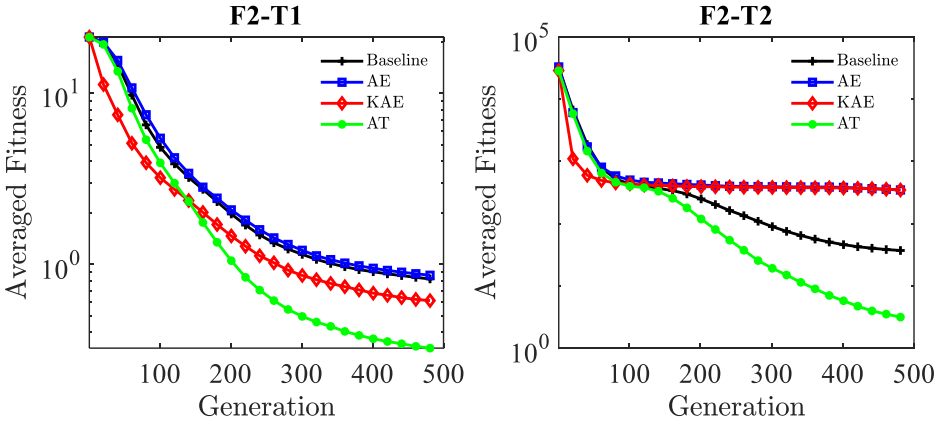
(a) CIMS



(b) CILS



(c) F1



(d) F2

Figure 4.2: Convergence curves of four DA methods on CIMS, CILS, F1, and F2.

4.2.3 Motivation

Based on the above studies of DA methods, it is observed that the differences in the mapping construction mechanisms can enable their mapping behaviours to have a specific bias in representing the connection from the source task to the target task. One specific bias can show the superiority in its preferred multitasking transfer scenario, while it has poor performance in other scenarios. Thus, this study is motivated to design an effective ensemble method to combine the strengths of multiple complementary DA methods that possess obviously distinct mapping characteristics, which will be able to enhance solution transferability from the source task to the target task on a variety of MTOPs. On the one hand, the efficacy of DA methods should be considered to ensure the effectiveness of knowledge transfer across tasks. On the other hand, the diversity of DA methods should be emphasized to enable each DA method to reach its full potential for conducting knowledge transfer. Thus, our ensemble method is developed to consider both the efficacy and diversity of various DA methods in the corresponding situations, which aims to improve the solution transferability across different tasks.

4.3 Methodology

This section presents the details of the proposed ensemble method (DAE), including two auxiliary components and domain adaptation selection (Algorithm 4.2 to Algorithm 4.4). In particular, the overall flowchart of DAE in a general EMT framework is illustrated in Fig. 4.3.

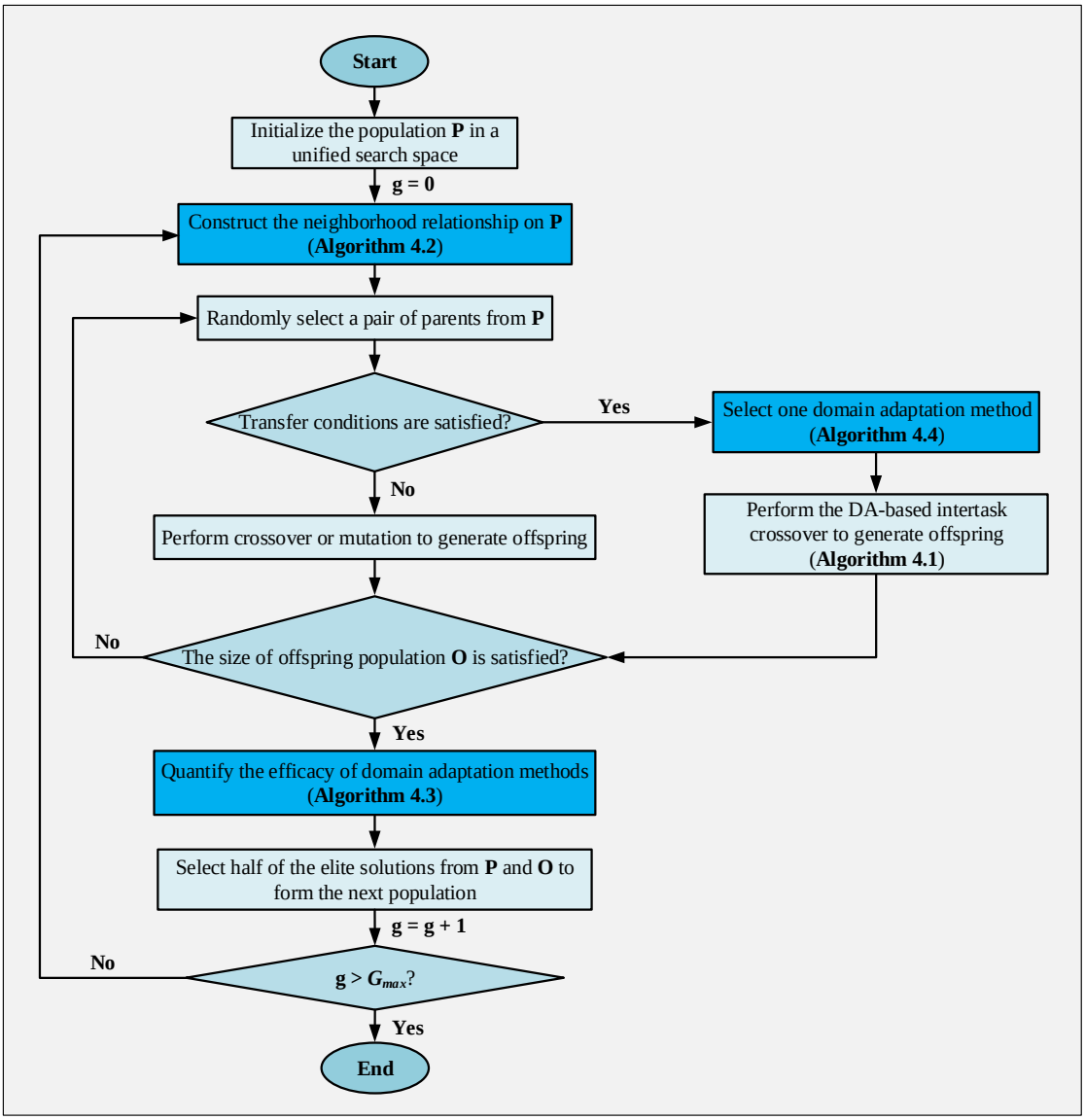


Figure 4.3: Flowchart of the DAE method in one general EMT framework.

To optimize one MTOP with K tasks, the population $\mathbf{P}$ is first initialized in a unified search space. Then, Algorithm 4.2 is run to construct the neighbourhood relationship among solutions in $\mathbf{P}$ by using the clustering method to divide those solutions associated with the same task into clusters. After that, for each pair of parent solutions

that are randomly selected from $\mathbf{P}$, offspring are generated by the DA-based intertask crossover (Algorithm 4.1) if the transfer conditions are satisfied, where the adopted DA method is selected by running Algorithm 4.4 in advance. Otherwise, offspring are generated by directly performing crossover or mutation on the parent solutions, which are assigned to the associated tasks of their parents. Once the size of offspring population $\mathbf{O}$ is satisfied, Algorithm 4.3 will be run to quantify the efficacy of each used DA method by calculating the performance improvement of their generated offspring in the evolutionary optimization process. Finally, half of the elite solutions are selected from $\mathbf{P}$ and $\mathbf{O}$ to form the next population. The above evolutionary process will be iteratively run until the stopping conditions are satisfied, and the final population $\mathbf{P}$ is outputted as the approximated solution set. In the following subsections, the details of Algorithm 4.2 to Algorithm 4.4 are introduced first, and then the details of embedding the proposed DAE into MFEA are provided in subsection 4.3.4.

Algorithm 4.2 Neighbourhood Relationship Construction (NRC)

Input: $\mathbf{P}$, K, λ
Output: $\mathbf{H}$
for $k = 1: K$
  $\mathbf{P}^k \leftarrow$ Collect the solutions associated to the k-th task by Eq. (4.22)
  Calculate the number of clusters n^k by Eq. (4.23)
  // Use the clustering method (HCM) to divide $\mathbf{P}^k$ into n^k clusters
  Initialize each $\mathbf{p}_i$ in $\mathbf{P}^k$ as a cluster $\mathbf{H}_i^k$ and its centroid $\mathbf{h}_i$
  count = $|\mathbf{P}^k|$
  while $count > n^k$
    Find two closest clusters $\mathbf{H}_a^k$ and $\mathbf{H}_b^k$ by Eq. (4.19)-Eq. (4.21)
    $\mathbf{H}_a^k = \mathbf{H}_a^k \cup \mathbf{H}_b^k$, and update its centroid $\mathbf{h}_a$ by Eq. (4.21)
    $count = count - 1$ and renumber the remaining clusters
  end
end
return $\mathbf{H} = \{\mathbf{H}^1, \dots, \mathbf{H}^k\}$ where $\mathbf{H}^k = \{\mathbf{H}_1^k, \dots, \mathbf{H}_{n^k}^k\}$ for $k = \{1, \dots, K\}$

4.3.1 Neighbourhood Relationship Construction

The neighbourhood relationship among the candidate solutions for each task is represented by dividing them into multiple clusters using the hierarchical clustering method (HCM) [111], [112]. The neighbouring solutions showing high similarities in the genetic genes will be gathered into the same cluster in decision space. As suggested in [115], [116], the HCM uses the linkage criterion of Ward with the Euclidean distance. To divide the subpopulation $\mathbf{P}^k$ associated with the k-th task into n^k clusters, the HCM is run in decision space by treating each solution in $\mathbf{P}^k$ as an initial cluster and then iteratively combining the two most similar clusters into one cluster until there are exactly n^k clusters left, which are denoted by $\mathbf{H}_1^k, \ldots, \mathbf{H}_{n^k}^k$. Specifically, each solution $\mathbf{p}_i = \{p_i^1, \ldots, p_i^d\}$ in $\mathbf{P}^k$ is first initialized as a cluster $\mathbf{H}_i^k$ and its centroid $\mathbf{h}_i$. Then, the count equal to $|\mathbf{P}^k|$ indicates the number of current clusters. The procedures for combining two clusters into one cluster will be iteratively repeated until n^k clusters are left. Concretely, two nearest clusters $\mathbf{H}_a^k$ and $\mathbf{H}_b^k$ could be found by comparing the sum of squared errors for any two clusters in the cluster set, whose indices a and b are identified as follows:

$$(a, b) = \underset{a,b \in \{1, \ldots n^k\}, a \neq b}{\arg\min} \left\{ dist(\mathbf{H}_a^k, \mathbf{H}_b^k) \right\}, \tag{4.19}$$

where $dist(\mathbf{H}_a^k, \mathbf{H}_b^k)$ is the sum of their squared errors, which can be calculated by

$$dist(\mathbf{H}_a^k, \mathbf{H}_b^k) = \sqrt{\frac{2 \cdot s_a \cdot s_b}{s_a + s_b}} \cdot ||\mathbf{h}_a - \mathbf{h}_b||, \tag{4.20}$$

where s_a and s_b represent the numbers of solutions in clusters $\mathbf{H}_a^k$, $\mathbf{H}_b^k$, and $||\mathbf{h}_a - \mathbf{h}_b||$ indicates the Euclidean distance between the centroids of two clusters (i.e., $\mathbf{h}_a$ and $\mathbf{h}_b$) in decision space. Here, the centroid $\mathbf{h}_i = \{h_i^1, \ldots, h_i^d\}$ of a clus-

ter $\mathbf{H}_i^k$ is computed as follows:

$$h_i^j = \frac{\sum_{\mathbf{p}\in\mathbf{H}_k^i} p^j}{\left|\mathbf{H}_k^i\right|}, \tag{4.21}$$

where p^j is the j-th decision variable of solution $\mathbf{p}$ in $\mathbf{H}_i^k$. Next, $\mathbf{H}_b^k$ is combined into $\mathbf{H}_a^k$, and its new centroid $\mathbf{h}_a$ is updated by Eq. (4.21). The value of count is decreased by 1, and the remaining clusters are renumbered as $\mathbf{H}_1^k, \ldots, \mathbf{H}_{count}^k$.

Using the HCM, the pseudocode for constructing the neighbourhood relationship of solutions for all tasks is given in Algorithm 4.2 with the inputs $\mathbf{P}$ (the population), K (the number of tasks), and λ (the parameter for controlling the number of clusters). As shown in Lines 1-2, those solutions associated with the k-th task are first collected into the subpopulation $\mathbf{P}^k$ as follows:

$$\mathbf{P}^k = \{\mathbf{p}_i | \boldsymbol{\tau}_i = k, \mathbf{p}_i \in \mathbf{P}\}, \tag{4.22}$$

where the skill factor τ_i of $\mathbf{p}_i$ indicates the index of its associated task. In Line 3, the number of clusters n^k for the k-th task is given by

$$n^k = \lceil \lambda \cdot |\mathbf{P}^k| \rceil, \tag{4.23}$$

where $\lambda \in [0, 1]$. Then, the HCM is used to divide $\mathbf{P}^k$ into n^k clusters $\mathbf{H}_1^k, \ldots, \mathbf{H}_{n^k}^k$, which is described in Lines 4-10.

After running the above procedures, the clustering results $\mathbf{H} = \{\mathbf{H}^1, \ldots, \mathbf{H}^k\}$ on K tasks will be returned to represent the neighbourhood relationship among the solutions in $\mathbf{P}$ in Line 12, where $\mathbf{H}^k = \{\mathbf{H}_1^k, \ldots, \mathbf{H}_{n^k}^k\}$ includes the remaining n^k clusters for the k-th task ($k \in \{1, \ldots, K\}$).

Algorithm 4.3 Efficacy Quantification of DA Methods (EQM)

Input: **P**, **O**, $\boldsymbol{pool}$, β, CR^{g-1}
Output: CR^{g}
1 **for** $label$ = 1 to $|\boldsymbol{pool}|$
2 S ← Collect transferred offspring marked with $label$ from **O**
3 Calculate the average improvement rate $\overline{IR}_{label}$ by Eq. (4.24)- Eq. (4.25)
4 **end**
5 Normalize $\{\overline{IR}_1, \dots, \overline{IR}_{|pool|}\}$ by Eq. (4.26)
6 Calculate the contribution ratios $\{CR_1^{g}, \dots, CR_{|\boldsymbol{pool}|}^{g}\}$ by Eq. (4.27)
7 **return** $CR^{g} = \{CR_1^{g}, \dots, CR_{|pool|}^{g}\}$

4.3.2 Efficacy Quantification of DA Methods

The DA method with more efficacy can contribute more to the performance improvement in the evolutionary process. As inspired by [60], [71], the improvement ratio and even the count of high-quality transferred offspring have been used for estimating the efficacy of evolutionary operators. Thus, the efficacy of each DA method for knowledge transfer can also be quantified based on the improvement ratios of transferred offspring against their parents in the evolutionary process. To be clear, the pseudocode is given in Algorithm 4.3 with the inputs: **P** (the parent population), **O** (the offspring population), $\boldsymbol{pool}$ (a set including multiple DA methods), $\boldsymbol{\beta}$ (a preference coefficient in the calculation of the accumulated contribution ratio from one DA method), and $CR^{g-1} = \{CR_1^{g-1}, \dots, CR_{|pool|}^{g-1}\}$, where CR_{label}^{g-1} denotes the accumulated contribution ratio from DA_{label} over the previous $g-1$ generations ($label \in \{1, \dots, |pool|\}$ refers to the index of DA_{label} and $|pool|$ is the number of DA methods in the $pool$). As shown in Lines 1-4, to quantify the efficacy of each DA_{label}, those transferred offspring marked with the particular label in **O** are first collected into an empty set **S**, and then the average improvement rate $\overline{IR}_{label}$ is cal-

culated as follows:

$$\overline{IR}_{label} = \frac{\sum_{\mathbf{s}\in\mathbf{S}} IR(\mathbf{s})}{|\mathbf{S}|}, \tag{4.24}$$

where $IR(\mathbf{s})$ is the fitness improvement ratio of transferred offspring $\mathbf{s}$ in $\mathbf{S}$, which is calculated by

$$IR(\mathbf{s}) = \max\left\{\frac{f(\mathbf{p}_s) - f(\mathbf{s})}{|f(\mathbf{p}_s)|}, 0\right\}. \tag{4.25}$$

Here, $\mathbf{p}_s$ is the immediate parent of $\mathbf{s}$ as defined in subsection 4.3.4, and they are associated with the same task. $f(\mathbf{p}_s)$ and $|f(\mathbf{p}_s)|$ are the scalar fitness of $\mathbf{p}_s$ and the absolute value of $f(\mathbf{p}_s)$, respectively. If $\mathbf{s}$ is better than $\mathbf{p}_s$, it will contribute to Eq. (4.24) but has no impact on Eq. (4.24) when $IR(\mathbf{s}) = 0$.

Next, as shown in Line 5, by normalizing $\overline{IR}_{label}$, NIR_{label} is obtained as follows:

$$NIR_{label} = \frac{\overline{IR}_{label}}{\sum_{label=1}^{|pool|} \overline{IR}_{label}}, \tag{4.26}$$

Then, in Line 6, the contribution ratio $CR^{\mathbf{g}}_{label}$ for DA_{label} is updated as follows:

$$CR^{\mathbf{g}}_{label} = \beta CR^{\mathbf{g}-1}_{label} + (1-\beta) NIR_{label}, \tag{4.27}$$

where $0 \leq \beta \leq 1$ is a preference coefficient that determines the proportions of the previous contribution and current contribution in $CR^{\mathbf{g}}_{label}$. Thus, $CR^{\mathbf{g}}_{label}$ represents the accumulation of the contribution ratio from DA_{label} over the previous $\mathbf{g}$ generations during the evolutionary search process. The relative proportions between the previous and current contributions can be easily adjusted by changing the value of β. Finally, as shown in Line 7, the contribution ratios of all DA methods from the $pool$

($CR^{\mathrm{g}} = \left\{CR_1^{\mathrm{g}}, \ldots, CR_{|pool|}^{\mathrm{g}}\right\}$) are returned as the quantization values of their efficacy.

Algorithm 4.4 Domain Adaptation Selection (DAS)

Input: $\mathbf{p}_a$, $\mathbf{p}_b$, $pool$, I, $\mathbf{H}$, CR^g
Output: DA_{label}, I
1 Collect the neighboring solutions of $\mathbf{p}_a$, $\mathbf{p}_b$ by Eq. (4.28)-Eq. (4.29)
2 $flag$ ← Identify the scenario on efficacy or diversity by Eq. (4.30)
3 **if** $flag$ is true // The efficacy is considered
4 DA_{label} ← Select one in $pool$ by roulette wheel selection based on their contribution ratios (CR^{g})
5 **else** // The diversity is emphasized
6 DA_{label} ← Randomly select one in the $pool$
7 **end**
8 Set $I(\mathbf{p}_a) = 1$ and $I(\mathbf{p}_b) = 1$
9 **return** DA_{label}, I

4.3.3 Domain Adaptation Selection

To clarify the running of domain adaptation selection (DAS), its pseudocode is given in Algorithm 4.4 with the inputs: $\mathbf{p}_a$ and $\mathbf{p}_b$ (a pair of parent solutions), a $pool$ including m DA methods (i.e., $pool = \{DA_1, \ldots, DA_m\}$), I (the indicator vector showing the transferred status of each solution in the population), $\mathbf{H}$ (the clustering results of the solutions on their associate tasks) and CR^g (the vector consisting of the accumulated contribution ratio of each DA method over the previous g generations). In terms of $\mathbf{p}_a$ and $\mathbf{p}_b$, the neighbourhood relationship of those solutions on their associated tasks, i.e., τ_a and τ_b, can be represented by the clustering results $\mathbf{H}^{\tau_a} = \left\{\mathbf{H}_1^{\tau_a}, \ldots, \mathbf{H}_{n^{\tau_a}}^{\tau_a}\right\}$ and $\mathbf{H}^{\tau_b} = \left\{\mathbf{H}_1^{\tau_b}, \ldots, \mathbf{H}_{n^{\tau_b}}^{\tau_b}\right\}$ from $\mathbf{H}$, where n^{τ_a} and n^{τ_b} are the numbers of clusters on tasks τ_a and τ_b, respectively. $\mathbf{H}^{\tau_a}$ and $\mathbf{H}^{\tau_b}$ are obtained by the HCM beforehand, as introduced in subsection 4.3.1. As shown in Line 1, given a solution $\mathbf{p}_a$, its neighbouring solutions are first collected into the set $B(\mathbf{p}_a)$, which is defined as follows:

$$B(\mathbf{p}_a) = \left\{\mathbf{p}_i | \forall \mathbf{p}_i \in \mathbf{H}^{\tau_a}_{index(\mathbf{p}_a)}, \mathbf{p}_i \neq \mathbf{p}_a\right\}, \tag{4.28}$$

where $index(\mathbf{p}_a)$ represents the index of the cluster to which $\mathbf{p}_a$ belongs, which can be identified as follows:

$$index(\mathbf{p}_a) = \underset{j \in \{1,\ldots,n^{\tau_a}\}}{\arg} \left\{\mathbf{p}_a \in \mathbf{H}^{\tau_a}_j\right\}, \tag{4.29}$$

Similarly, the set $B(\mathbf{p}_b)$ including the neighbouring solutions of $\mathbf{p}_b$ is also obtained according to Eq. (4.28)-Eq. (4.29). Then, in Line 2, a Boolean $flag$ based on the transferred status of all solutions in $B(\mathbf{p}_a)$ and $B(\mathbf{p}_b)$ can be constructed by

$$flag = \begin{cases} true, & \text{if } \left(\sum_{\mathbf{p}_i \in B(\mathbf{p}_a) \cup B(\mathbf{p}_b)} I(\mathbf{p}_i)\right) = 0 \\ false, & \text{otherwise} \end{cases}, \tag{4.30}$$

where $I(\mathbf{p}_i) = 0$ if $\mathbf{p}_i$ has never been used for knowledge transfer before and $I(\mathbf{p}_i) = 1$ otherwise. Thus, two situations are identified to consider the efficacy and diversity of various DA methods according to the Boolean value of $flag$. As shown in Line 3, when the $flag$ is true, indicating that all the neighbouring solutions of $\mathbf{p}_a$ and $\mathbf{p}_b$ have not been used for knowledge transfer before, the DA method (i.e., DA_{label}) is selected by the RWS based on their contribution ratios (i.e., CR^g), which aims to select one DA method with better efficacy. Here, the contribution ratio of each DA method can be quantified by their induced performance improvements during the evolutionary process, as introduced in subsection 4.3.2. Otherwise, as shown in Line 6, one DA method (i.e., DA_{label}) is randomly selected with the same probability from the $pool$, which focuses on the diversity of DA methods for knowledge transfer. After that, to update the record that $\mathbf{p}_a$ and $\mathbf{p}_b$ have been used for knowledge transfer, $I(\mathbf{p}_a)$ and $I(\mathbf{p}_b)$ are set to 1 in Line 8. Finally, in Line 9,

DA_{label} is returned as the adopted DA method in the DA-based intertask crossover, while I indicates the updated transferred status of solutions.

Algorithm 4.5 The Framework of MFEA-DAE

Input: K tasks, N, G_{max}, $pool = \{DA_1, \ldots, DA_m\}$, λ, β, and rmp
Output: the best solution of each task from $\mathbf{P}$

Initialize $\mathbf{P}$ to have $N \times K$ solutions
Assign skill factor τ_i to every $\mathbf{p}_i \in \mathbf{P}$ by Eq. (4.31), and evaluate them
Set $\mathrm{g} = 1$, $CR^1 = \{1/|pool|, \ldots, 1/|pool|\}$
while $\mathrm{g} \leq G_{max}$
    $\mathbf{H} \leftarrow$ NRC ($\mathbf{P}$, K, λ) // Algorithm 4.2
    Set $I(\mathbf{p}_i) = 0$ for each $\mathbf{p}_i \in \mathbf{P}$
    Set offspring population $\mathbf{O}$ to be an empty set
    while $|\mathbf{O}| < N \times K$
        Randomly select two candidate parents $\mathbf{p}_a$ and $\mathbf{p}_b$ from $\mathbf{P}$
        if $\tau_a == \tau_b$
            $[\mathbf{c}_a, \mathbf{c}_b] \leftarrow$ Intratask crossover between $\mathbf{p}_a$ and $\mathbf{p}_b$
            Assign offspring $\mathbf{c}_a$ and $\mathbf{c}_b$ with skill factor τ_a
        else if $rand \leq rmp$
            $DA_{label} \leftarrow$ DAS ($\mathbf{p}_a$, $\mathbf{p}_b$, $pool$, I, $\mathbf{H}$, CR^g) // Algorithm 4.4 // Algorithm 4.1
            $[\mathbf{c}_a, \mathbf{c}_b] \leftarrow$ DA-Based Intertask Crossover ($\mathbf{p}_a$, $\mathbf{p}_b$, DA_{label})
            Each offspring is randomly assigned skill factor τ_a or τ_b
            Mark $\mathbf{c}_a$ and $\mathbf{c}_b$ as transferred offspring from DA_{label}
            Mark $\mathbf{p}_a$ or $\mathbf{p}_b$ as immediate parents of $\mathbf{c}_a$ and $\mathbf{c}_b$
        else
            $\mathbf{c}_a \leftarrow$ local variation (mutation) of $\mathbf{p}_a$
            Assign offspring $\mathbf{c}_a$ with skill factor τ_a
            $\mathbf{c}_b \leftarrow$ local variation (mutation) of $\mathbf{p}_b$
            Assign offspring $\mathbf{c}_b$ with skill factor τ_b
        end
        Evaluate $\mathbf{c}_a$ and $\mathbf{c}_b$ for their assigned skill factors only
        $\mathbf{O} = \mathbf{O} \cup [\mathbf{c}_a, \mathbf{c}_b]$
    end
    $\mathrm{g} = \mathrm{g} + 1$
    $CR^{\mathrm{g}} \leftarrow$EQM ($\mathbf{P}$, $\mathbf{O}$, $pool$, β, $CR^{\mathrm{g}-1}$) //Algorithm 4.3
    $\mathbf{P} \leftarrow$ Select top $N \times K$ solutions in $\mathbf{P} \cup \mathbf{O}$ based on scalar fitness
end
return the best solution of each task from $\mathbf{P}$

4.3.4 Embedding the DAE Method into MFEA

In this subsection, the proposed DAE method including the above three components is embedded into MFEA, and the resultant algorithm is called MFEA-DAE. To clarify the running of MFEA-DAE, its pseudocode is provided in Algorithm 4.5 with the inputs K tasks, N (the size of the population) and G_{max} (the pre-set maximum number of generations). In addition, four DA methods are included in the $pool$, including AE, KAE, AT, and the baseline that directly transfers solutions of the source task to the target task without any adaptation. The complementarities among AE, KAE and AT are studied in the subsection 4.2.2. The parameters λ and β are used to control the number of clusters on each task and the preference coefficient for calculating the accumulated contribution ratio of each DA method in the evolutionary process, respectively. Moreover, rmp is a pre-set random mating probability, which controls the frequency of intertask crossover. As shown in Line 1, a single population $\mathbf{P}$ is first formed by randomly sampling $N \times K$ individuals in a unified search space $Y \in [0,1]^D$, where $D = \max\{d^k\}$ and d^k is the number of decision variables of the k-th task. Then, in Line 2, the skill factor τ_i of each solution $\mathbf{p}_i$ in $\mathbf{P}$ is randomly assigned according to

$$\tau_i = mod(i, K) + 1. \tag{4.31}$$

After that, all solutions in $\mathbf{P}$ will be evaluated for their assigned skill factor only. In Line 3, the generation counter $\mathbf{g}$ is set to 1, and the initial contribution ratio CR^1_{label} for each DA method is set to $1/|pool|$, where $label = \{1, \ldots, |pool|\}$.

The main evolutionary process is shown in Lines 4-31. At the start of each generation, the neighbourhood relationship among all solutions is first built by Algorithm 4.2 in Line 5. Then, for each solution $\mathbf{p}_i$ in $\mathbf{P}$, the indicator $I(\mathbf{p}_i)$ is set to 0 to indicate

that $\mathbf{p}_i$ has not been used for knowledge transfer in Line 6, while the offspring population $\mathbf{O}$ is set to an empty set in Line 7. In the process of generating offspring, considering each pair of parent solutions $\mathbf{p}_a$ and $\mathbf{p}_b$ that are randomly selected from $\mathbf{P}$, the knowledge transfer process will be triggered in Lines 14-18 when $\tau_a \neq \tau_b$ and $rand \leq rmp$. First, Algorithm 4.4 is performed to select one DA method DA_{label} from the $pool$, which is used to map $\mathbf{p}_a$ and $\mathbf{p}_b$ to their corresponding target tasks in the DA-based intertask crossover (Algorithm 4.1). Specifically, given one solution $\mathbf{p}_a$, the transformed solution $\mathbf{p}_a'$ for its target task τ_b is obtained as follows:

$$\mathbf{p}_a' = \begin{cases} \mathbf{M}\mathbf{p}_a, & \text{if } DA_{label} = \text{AE} \\ \mathbf{M}_k\mathbf{K}(\mathbf{P}, \mathbf{p}_a), & \text{if } DA_{label} = \text{KAE} \\ \mathbf{p}_a\mathbf{A} + \mathbf{b}, & \text{if } DA_{label} = \text{AT} \\ \mathbf{p}_a, & \text{otherwise} \end{cases} \tag{4.32}$$

where $\mathbf{M}$, $\mathbf{M}_k$, $\mathbf{A}$ and $\mathbf{b}$ are the learnt mapping parameters of AE, KAE, and AT, respectively, and their derivation formulas have been introduced in subsection 4.2.1. The transformed solution $\mathbf{p}_b'$ for its target task τ_a can be obtained in the same manner by Eq. (4.32). Then, offspring solutions $\mathbf{c}_a$ and $\mathbf{c}_b$ can be generated by Algorithm 4.1. In this case, τ_a or τ_b is randomly assigned to $\mathbf{c}_a$ and $\mathbf{c}_b$. Moreover, to quantify the efficacy of the DA method DA_{label}, $\mathbf{c}_a$ and $\mathbf{c}_b$ are marked as transferred offspring from DA_{label} and $\mathbf{p}_a$ or $\mathbf{p}_b$ having the same skill factor as $\mathbf{c}_a$ and $\mathbf{c}_b$ is marked as its immediate parent in Lines 17-18. Otherwise, $\mathbf{c}_a$ and $\mathbf{c}_b$ are generated by performing intratask crossover or mutation, which directly imitate the skill factors of their parents in Lines 11-12 and Lines 20-23. Here, simulated binary crossover (SBX) [94] and polynomial-based mutation (PM) [95] are suggested. After that, both $\mathbf{c}_a$ and $\mathbf{c}_b$ are evaluated based on their assigned skill factors only and then added into $\mathbf{O}$ in Lines 25-26. The above procedures in Lines 9-26 will be run iteratively until the number of offspring reaches $N \times K$. Then, $\mathbf{g}$ is increased by 1,

the efficacy of each DA method is updated by Algorithm 4.3, and $N \times K$ solutions based on scalar fitness are selected from the combination set of $\mathbf{P}$ and $\mathbf{O}$ to form the next population $\mathbf{P}$ in Lines 28-30. While $\mathbf{g}$ is smaller than G_{max}, the above evolutionary process in Lines 5-30 will be run. Otherwise, the best solution of each task from the final population $\mathbf{P}$ will be returned in Line 32.

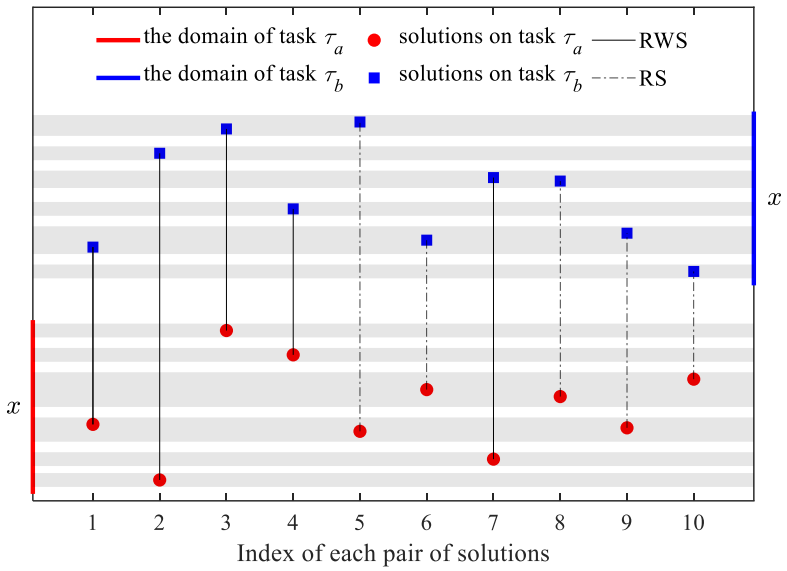


Figure 4.4: Example of running the RWS and the RS in the DAE method.

To show how our ensemble method works, a simple example is provided in Fig. 4.4, which shows the scenario of running the RWS or random selection (RS) on 10 pairs of solutions, where the solutions of two 1-D tasks τ_a and τ_b are marked with red circles and blue squares, respectively. Note that x represents the value of the solution in 1-D search space. The grey areas in Fig. 4.4 represent the clustering results of solutions on each task. Therefore, the solutions in the same grey area are regarded as neighbouring solutions. For the first 4 and 7-th pairs of solutions, $flag$ in Eq. (4.30) will be true as their neighbouring solutions have not been selected to generate transferred solutions before. Thus, the RWS is run based on the contribution ratios to se-

lect one DA method from $pool$ with relatively larger contributions in consideration of the efficacy. However, for each of the remaining pairs of candidate solutions, $flag$ will be false, as there is at least one solution from their neighbouring solutions having been used for knowledge transfer before. In this case, the RS is used to randomly select one DA method from $pool$ in consideration of the diversity. In this way, the effectiveness of the DAE method can be ensured by considering both the efficacy and diversity of complementary DA methods, which could further improve the transferability of solutions between distinct tasks.

4.4 Experimental Study

4.4.1 Test Problems and Parameter Settings

1) Test Problems

In the experiments, two synthetic single-objective multitasking test suites are used. The first benchmark suite [96] considers the task similarity and the degree of global optima intersection, which includes nine multitasking test problems, i.e., CIHS, CIMS, CILS, PIHS, PIMS, PILS, NIHS, NIMS, and NILS. Another multitasking benchmark suite [74] includes eight multitasking test problems (F1-F8). Each problem consists of two tasks, both of which are single-objective optimization problems. F1-F8 explicitly possess heterogeneous features, such as different decision spaces or fitness landscapes. To be specific, the numbers of decision variables of two tasks are equal on F1-F4, while they are different on F5-F8. Besides, the global optima of two tasks are the same on F1, F2, F5, and F6, while they are different on the remaining test problems. The multitasking benchmark suite can be categorized as follows: the numbers of decision variables of two tasks are the same on F1-F4, while they are different on F5-F8;

Moreover, the global optima of two tasks are the same on F1, F2, F5, and F6, while they are different on the rest of problems. In addition, one synthetic multiobjective multitasking test suite [117] is also used to study the effectiveness of DAE by embedding it into the multiobjective EMT algorithms. Finally, a practical case of the parameterized planar kinematic arm is adopted to study the effectiveness of DAE on solving practical optimization examples. The maximum number of function evaluations is set to 100 000 for each test problem from the synthetic multitasking test suites, while it is 60 000 for the practical optimization problems. The objective values for SOP or IGD [118] values for MOP over 20 independent runs are collected for performance comparison, where the Wilcoxon rank sum test with a 0.05 significance level is used to show the statistically significant differences in the numerical results.

First, the effectiveness of DAE method is studied by embedding it into three competitive EMT algorithms (MFEA [27], MFEA-II [49], and MFEA-AKT [60]). To study the effectiveness of DAE, MFEA-DAE as a representative algorithm is further compared to its four variants, i.e., MFEA-baseline, MFEA-AE, MFEA-KAE, and MFEA-AT, each of which adopts one DA method (the baseline, AE, KAE, and AT) from the $\boldsymbol{pool}$. Then, three variants of DAE are also designed to validate the effectiveness of the two auxiliary components. In addition, MFEA-DAE is also compared with five recently proposed EMT algorithms (MFEA-II [72], MFEA-AKT, MTEA-AD [56], MTES [119], and MKTDE [120]) and a standard single-task EA (SOEA) [8] that solves each task separately. The above experiments are conducted on two synthetic single-objective multitasking test suites. Moreover, DAE is also embedded into two multiobjective EMT algorithms (MO-MFEA [9] and MO-MFEA-II [50]) to validate its effectiveness on the multiobjective multitasking test problems. Finally, one practical case study is also conducted.

2) Parameter Settings

In the experiments, the DAE method is embedded into three single-objective evolutionary multitasking (EMT) algorithms (MFEA, MFEA-II, and MFEA-AKT). In addition, SOEA, MFEA-AE, MFEA-KAE, MFEA-AT, MTEA-AD, MTES, and MKTDE are also used as the compared algorithms. Besides, the DAE method is embedded into two multiobjective EMT algorithms (MO-MFEA and MO-MFEA). The common parameter settings of the above variants of MFEA are kept consistent, while those of MTEA-AD, MTES, and MKTDE are set the same to their original papers.

Table 4.1: Parameter settings of all compared algorithms.

Algorithm	Parameter settings
SOEA	$N = 100,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA	$N = 100,\ rmp = 0.3,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA-II	$N = 100,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA-AKT	$N = 100,\ rmp = 0.3,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA-AE	$N = 100,\ rmp = 0.3,\ NS = 100,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA-KAE	$N = 100,\ rmp = 0.3,\ NS = 100,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MFEA-AT	$N = 100,\ rmp = 0.3,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15,\ \alpha = 0.5$
MTEA-AD	$N = 100,\ \alpha = 0.1,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MTES	$n = 25$
MKTDE	$N = 100,\ F = 0.5,\ CR = 0.6$
MO-MFEA	$N = 100,\ rmp = 0.3,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
MO-MFEA-II	$N = 100,\ p_c = 1.0,\ \eta_c = 15,\ p_m = 1/d,\ \eta_m = 15$
DAE	$\lambda = 0.8,\ \beta = 0.5$

Tab. 4.1 lists the parameter settings of all algorithms. In terms of MFEA and its variants (MFEA-II, MFEA-AKT, MFEA-baseline, MFEA-AE, MFEA-KAE, MFEA-AT, MO-MFEA, MO-MFEA-II, and the proposed MFEA-DAE), the population size N for each task is set to 100, and rmp is set to 0.3 except MFEA-II and MO-MFEA-II. For a fair comparison, MFEA and its variants are configured with the same evolutionary operators, i.e., simulated binary crossover (SBX) with probability $p_c = 1.0$

and distribution index $\eta_c = 15$, and polynomial-based mutation (PM) with probability $p_m = 1/d$ and distribution index $\eta_m = 15$, where d is the number of decision variables. Note that the numbers of sampled solutions for learning $\mathbf{M}$ and $\mathbf{M}_k$ are 100 in MFEA-AE and MFEA-KAE, and the preference coefficient α is set to 0.5 in MFEA-AT. In MFEA-DAE, $\mathbf{M}$, $\mathbf{M}_k$, $\mathbf{A}$, and $\mathbf{b}$ are learned based on the same parameter settings used in MFEA-AE, MFEA-KAE and MFEA-AT. In addition, two additional parameters β and λ in MFEA-DAE are set to 0.5 and 0.8, respectively. For the single-task EA (SOEA) that solves each task separately, the population size N for each task is set to 100. Here, the same evolutionary operators (i.e., SBX and PM) are used in SOEA, in which their parameter settings are kept consistent with the above settings. In MTEA-AD, the population size N for each task is set to 100 and the knowledge transfer probability α is set to 0.1. The parameter settings of SBX and PM are kept consistent with the above settings. In MTES, the population size n is configured as 25 for each task and the learning rate α and standard deviation σ are adjusted based on the value of θ. Besides, in MKTDE, the total population size, two parameters F and CR are set to 100, 0.5, and 0.6, respectively.

4.4.2 Results of Embedding DAE into EMT Algorithms

Here, the DAE method is embedded into three competitive EMT algorithms (MFEA, MFEA-II, and MFEA-AKT), forming three enhanced algorithms, called MFEA-DAE, MFEA-II-DAE, and MFEA-AKT-DAE. The comparison results from 20 independent runs on the two test suites are listed in Tab. 4.2 and Tab. 4.3, where "~", "+", and "-" indicate the numbers of similar, better, and worse results obtained by MFEA, MFEA-II, MFEA-AKT when compared with their enhanced versions with DAE (MFEA-DAE, MFEA-II-DAE, and MFEA-AKT-DAE), respectively.

Table 4.2: Mean objective values and standard deviations obtained by three EMT algorithms and their enhanced algorithms with DAE on the first test suite.

Problem	MFEA	MFEA-DAE	MFEA-II	MFEA-II-DAE	MFEA-AKT	MFEA-AKT-DAE
CIHS-T1	2.68e-02(8.86e-03)-	**2.20e-03(5.47e-03)**	2.94e-02(1.01e-02)-	**1.14e-03(3.42e-03)**	6.09e-01(5.51e-02)-	**2.66e-01(7.59e-02)**
CIHS-T2	4.52e+01(1.83e+01)-	**2.69e+00(1.04e+01)**	5.53e+01(2.49e+01)-	**3.81e+00(1.16e+01)**	2.19e+02(3.40e+01)-	**1.21e+02(2.69e+01)**
CIMS-T1	7.22e-01(5.08e-01)-	**2.15e-03(1.37e-03)**	9.70e-01(5.21e-01)-	**6.39e-03(2.18e-03)**	4.53e+00(4.97e-01)-	**2.40e+00(5.29e-01)**
CIMS-T2	3.03e+01(2.29e+01)-	**3.89e-03(4.45e-03)**	4.86e+01(2.48e+01)-	**2.67e-02(1.66e-02)**	2.41e+02(4.78e+01)-	**1.14e+02(2.74e+01)**
CILS-T1	1.00e+01(1.02e+01)-	**1.29e-02(7.27e-03)**	1.54e+01(9.10e+00)-	**6.46e-02(4.01e-02)**	2.02e+01(6.14e-02)-	**3.83e+00(5.61e+00)**
CILS-T2	5.19e+02(5.52e+02)-	**7.34e-03(8.48e-03)**	5.94e+02(4.12e+02)-	**1.38e-01(1.49e-01)**	4.41e+03(4.19e+02)-	**4.89e+02(1.41e+03)**
PIHS-T1	2.09e+02(9.42e+01)-	**8.44e+01(1.76e+01)**	2.48e+02(1.22e+02)~	2.76e+02(1.14e+02)	4.94e+02(7.37e+01)-	**2.38e+02(6.84e+01)**
PIHS-T2	9.28e-02(2.35e-02)-	**2.25e-02(1.66e-02)**	6.32e-02(2.34e-02)~	6.08e-02(2.40e-02)	5.49e+01(1.27e+01)-	**1.73e+01(3.38e+00)**
PIMS-T1	9.58e-01(4.37e-01)-	**5.42e-02(7.54e-02)**	9.40e-01(3.72e-01)-	**6.05e-02(4.75e-02)**	3.37e+00(3.47e-01)-	**2.83e+00(3.76e-01)**
PIMS-T2	1.66e+02(4.42e+01)-	**4.96e+01(9.08e+00)**	1.55e+02(4.42e+01)-	**5.04e+01(1.07e+01)**	8.18e+02(2.24e+02)-	**1.45e+02(4.32e+01)**
PILS-T1	9.44e-01(4.76e-01)-	**1.04e-02(3.20e-03)**	7.80e-01(4.40e-01)-	**2.16e-02(4.99e-03)**	4.47e+00(6.14e-01)-	**7.01e-02(1.05e-02)**
PILS-T2	1.35e+00(6.12e-01)-	**7.22e-02(2.23e-02)**	1.07e+00(3.76e-01)-	**9.18e-02(4.02e-02)**	4.28e+00(7.51e-01)-	**3.30e-01(4.74e-02)**
NIHS-T1	1.69e+02(4.35e+01)-	**5.01e+01(1.01e+01)**	**1.79e+02(3.65e+01)+**	2.11e+02(7.29e+02)	1.75e+03(5.39e+02)-	**2.93e+02(1.36e+02)**
NIHS-T2	7.30e+01(3.18e+01)-	**1.22e+00(3.47e+00)**	9.66e+01(3.40e+01)-	**1.54e+01(6.88e+01)**	2.55e+02(3.32e+01)-	**1.37e+02(4.16e+01)**
NIMS-T1	3.63e-02(1.04e-02)-	**1.53e-02(6.75e-03)**	2.95e-02(8.20e-03)~	3.11e-02(1.10e-02)	7.40e-01(9.23e-02)-	**3.93e-01(7.19e-02)**
NIMS-T2	7.64e+00(1.58e+00)-	**1.12e+00(4.50e-01)**	8.73e+00(2.30e+00)-	**4.39e+00(1.76e+00)**	1.87e+01(2.55e+00)-	**1.13e+01(1.73e+00)**
NILS-T1	3.36e+02(7.64e+01)-	**9.12e+01(2.02e+01)**	3.37e+02(9.08e+01)~	3.06e+02(1.07e+02)	5.49e+02(7.43e+01)-	**4.73e+02(5.62e+01)**
NILS-T2	7.37e+02(2.77e+02)~	8.71e+02(3.12e+02)	6.84e+02(2.59e+02)~	6.30e+02(1.80e+02)	4.54e+03(5.82e+02)~	4.42e+03(6.14e+02)
~/+/-	1/0/17	\	5/1/12	\	1/0/17	\

Table 4.3: Mean objective values and standard deviations obtained by three EMT algorithms and their enhanced algorithms with DAE on the second test suite.

Problem	MFEA	MFEA-DAE	MFEA-II	MFEA-II-DAE	MFEA-AKT	MFEA-AKT-DAE
F1-T1	2.71e-02(1.07e-02)-	**1.89e-02(9.32e-03)**	3.33e-02(1.34e-02)-	**4.93e-04(2.02e-03)**	6.45e-01(6.58e-02)-	**2.45e-01(7.23e-02)**
F1-T2	4.78e+01(2.25e+01)-	**2.61e+01(2.18e+01)**	5.71e+01(2.13e+01)-	**1.69e+00(7.20e+00)**	2.35e+02(4.07e+01)-	**1.04e+02(2.09e+01)**
F2-T1	8.03e-01(4.13e-01)-	**2.80e-01(3.62e-01)**	8.19e-01(5.48e-01)-	**6.45e-03(1.86e-03)**	4.28e+00(4.74e-01)-	**2.60e+00(4.79e-01)**
F2-T2	3.61e+01(2.51e+01)-	**1.02e+01(2.04e+01)**	4.77e+01(3.35e+01)-	**2.67e-02(1.48e-02)**	2.66e+02(3.86e+01)-	**1.36e+02(3.92e+01)**
F3-T1	2.93e+02(3.06e+02)-	**1.57e+02(5.66e+01)**	5.05e+02(1.09e+03)~	1.07e+03(1.81e+03)	1.15e+04(3.69e+03)-	**2.71e+03(1.51e+03)**
F3-T2	6.41e+00(2.45e+00)-	**3.51e+00(2.20e+00)**	4.09e+00(2.16e+00)~	3.47e+00(2.05e+00)	2.61e+02(2.96e+01)-	**1.07e+02(1.59e+01)**
F4-T1	3.27e+02(9.32e+01)-	**1.07e+02(1.90e+01)**	2.59e+02(1.25e+02)~	2.19e+02(9.89e+01)	5.84e+02(9.33e+01)-	**4.76e+02(7.39e+01)**
F4-T2	7.33e+02(2.79e+02)~	8.03e+02(2.90e+02)	6.37e+02(2.28e+02)~	6.31e+02(2.48e+02)	4.41e+03(4.46e+02)~	4.33e+03(7.66e+02)
F5-T1	3.34e+02(9.54e+01)-	**6.95e-02(5.36e-02)**	3.75e+02(6.00e+01)-	**1.91e-01(1.20e-01)**	3.92e+02(5.77e+01)-	**2.10e+00(9.02e-01)**
F5-T2	3.21e-03(1.31e-03)-	**1.56e-05(3.33e-05)**	2.68e-03(1.73e-03)-	**5.06e-06(1.78e-05)**	3.79e-01(1.48e-01)-	**1.48e-03(1.74e-03)**
F6-T1	8.81e-01(3.60e-01)-	**1.23e-02(5.04e-03)**	9.21e-01(3.02e-01)-	**1.83e-02(7.38e-03)**	3.49e+00(6.51e-01)-	**6.36e-02(1.12e-02)**
F6-T2	4.10e-02(1.79e-02)-	**8.35e-03(7.18e-03)**	4.22e-02(1.95e-02)-	**8.66e-03(1.10e-02)**	5.66e-01(9.75e-02)-	**1.56e-01(2.78e-02)**
F7-T1	4.34e+02(9.63e+02)-	**4.81e+01(1.02e-01)**	4.43e+02(7.47e+02)-	**1.51e+02(9.01e+01)**	4.86e+03(2.24e+03)-	**2.34e+02(8.97e+01)**
F7-T2	6.35e+01(2.73e+01)-	**1.38e-02(8.94e-03)**	4.81e+01(2.74e+01)~	4.40e+01(3.74e+01)	1.14e+02(2.66e+01)-	**4.13e+01(1.51e+01)**
F8-T1	3.11e+02(1.11e+02)-	**1.16e+02(3.07e+01)**	2.56e+02(1.18e+02)~	2.96e+02(1.06e+02)	5.65e+02(8.95e+01)-	**4.62e+02(9.05e+01)**
F8-T2	1.07e+02(9.33e+01)~	1.54e+02(1.16e+02)	1.23e+02(1.27e+02)~	8.79e+01(9.16e+01)	**1.56e+03(2.60e+02)+**	2.38e+03(7.10e+02)
~/+/-	2/0/14	\	7/0/9	\	1/1/14	\

"+" (or "-") indicates the original EMT algorithm is better (or worse) than the enhanced EMT algorithm with DAE, and "~" indicates they obtain the statistically similar performance. The best result on each test problem of each task is highlighted in bold.

Regarding the results in Tab. 4.2 and Tab. 4.3, the performance of MFEA-DAE is obviously better than that of MFEA on all MTOPs. Although they obtain similar results on three cases (NILS-T2, F4-T2 and F8-T2), they further improve the performance on Task 1 (T1) of the three MTOPs. Similarly, MFEA-AKT-DAE also achieves remarkable improvement over MFEA-AKT on all test problems except F8. Although MFEA-AKT-DAE obviously improves the performance of MFEA-AKT on F8-T1, it leads to performance degradation on F8-T2. In MFEA-II, the transfer intensity (i.e., rmp) is estimated online based on similarity among tasks to mitigate the negative transfer. It is observed that the performance improvement brought by MFEA-II-DAE is not as obvious as that in MFEA-DAE and MFEA-AKT-DAE. As observed in Tab. 4.2 and Tab. 4.3, MFEA-II-DAE achieves significant performance improvements on CIHS, CIMS, CILS, PIMS, PILS, F1, F2, F5, and F6, while MFEA-II-DAE and MFEA-II have very similar performance on the rest of the test problems. The reason for the lack of obvious performance improvement on these problems is that the learnt rmp is small due to the explicit discrepancy between the global optima of two tasks, which hinders DA methods from improving the solution transferability between two distinct tasks. According to the above comparison results, it is clear that the embedding of DAE into MFEA, MFEA-II, and MFEA-AKT can bring significant improvements for solving these multitasking test problems. Thus, it is reasonable to conclude that DAE within different EMT algorithms can further improve the solution transferability between two distinct tasks.

Table 4.4: Mean objective values and standard deviations obtained by MFEA-DAE and its variants on the first test suite.

Problem	MFEA	MFEA-AE	MFEA-KAE	MFEA-AT	MFEA-DAE
CIHS-T1	4.89e-02(1.37e-02)-	4.37e-04(1.46e-04)+	3.37e-02(1.18e-02)-	2.16e-02(6.68e-03)-	2.20e-03(5.47e-03)
CIHS-T2	9.08e+01(3.60e+01)-	2.23e-01(7.26e-02)+	3.12e+02(8.64e+01)-	1.81e+01(1.40e+01)-	2.69e+00(1.04e+01)
CIMS-T1	1.97e-01(8.40e-02)-	1.78e-02(3.78e-03)-	8.64e-01(4.70e-01)-	2.18e-01(2.67e-01)-	2.15e-03(1.37e-03)
CIMS-T2	1.44e+01(6.29e+00)-	2.05e-01(7.76e-02)-	3.25e+02(7.70e+01)-	5.15e+00(9.57e+00)-	3.89e-03(4.45e-03)
CILS-T1	2.02e+01(1.80e-01)-	2.12e+01(3.88e-02)-	9.16e-01(4.38e-01)-	8.10e+00(1.01e+01)-	1.29e-02(7.27e-03)
CILS-T2	2.71e+03(6.21e+02)-	7.81e+02(2.75e+02)-	8.59e+02(3.10e+02)-	4.31e+02(5.57e+02)-	7.34e-03(8.48e-03)
PIHS-T1	1.14e+02(1.69e+01)-	2.04e+02(1.34e+02)-	3.16e+02(9.08e+01)-	9.98e+01(2.53e+01)~	8.44e+01(1.76e+01)
PIHS-T2	1.23e-01(3.09e-02)-	9.40e-02(3.12e-02)-	8.27e-02(2.28e-02)-	2.45e-02(1.33e-02)~	2.25e-02(1.66e-02)
PIMS-T1	2.39e-01(5.02e-02)-	2.83e-01(2.40e-01)-	6.10e-01(4.44e-01)-	4.80e-02(2.31e-02)~	5.42e-02(7.54e-02)
PIMS-T2	1.25e+02(1.78e+01)-	5.07e+01(1.12e+01)-	1.55e+02(3.90e+01)-	9.27e+01(5.35e+00)-	4.96e+01(9.08e+00)
PILS-T1	1.37e+00(4.38e+00)-	1.73e-02(4.18e-03)-	9.18e-01(3.74e-01)-	1.67e-01(2.21e-01)-	1.04e-02(3.20e-03)
PILS-T2	3.39e-01(3.12e-01)-	1.28e-01(4.46e-02)-	1.76e+00(6.81e-01)-	4.71e-01(3.03e-01)-	7.22e-02(2.23e-02)
NIHS-T1	1.69e+02(3.68e+01)-	4.81e+01(1.40e-01)+	1.44e+02(4.82e+01)-	1.49e+02(5.26e+01)-	5.01e+01(1.01e+01)
NIHS-T2	2.21e+02(6.22e+01)-	2.26e-01(1.08e-01)+	3.21e+02(9.40e+01)-	5.00e+01(2.90e+01)-	1.22e+00(3.47e+00)
NIMS-T1	4.63e-02(1.17e-02)-	3.73e-02(1.41e-02)-	3.16e-02(7.38e-03)-	1.72e-02(5.49e-03)~	1.53e-02(6.75e-03)
NIMS-T2	3.06e+00(5.26e-01)-	7.21e-01(1.55e-01)+	3.63e+00(7.28e-01)-	3.28e+00(1.24e+00)-	1.12e+00(4.50e-01)
NILS-T1	2.95e+02(7.12e+01)-	2.30e+02(1.25e+02)-	3.31e+02(9.50e+01)-	1.07e+02(2.47e+01)~	9.12e+01(2.02e+01)
NILS-T2	2.88e+03(6.03e+02)-	7.98e+02(2.65e+02)~	2.38e+03(1.47e+03)-	8.04e+02(2.80e+02)~	8.71e+02(3.12e+02)
~/+/-	0/0/18	1/5/12	0/0/18	6/0/12	\

Table 4.5: Mean objective values and standard deviations obtained by MFEA-DAE and its variants on the second test suite.

Problem	MFEA	MFEA-AE	MFEA-KAE	MFEA-AT	MFEA-DAE
F1-T1	2.71e-02(1.07e-02)-	4.66e-02(1.43e-02)-	3.02e-02(9.53e-03)-	1.65e-02(5.22e-03)~	1.89e-02(9.32e-03)
F1-T2	4.78e+01(2.25e+01)-	3.18e+02(9.23e+01)-	2.77e+02(1.17e+02)-	1.75e+01(1.57e+01)~	2.61e+01(2.18e+01)
F2-T1	8.03e-01(4.13e-01)-	8.48e-01(4.43e-01)-	6.01e-01(4.22e-01)-	3.14e-01(3.45e-01)~	2.80e-01(3.62e-01)
F2-T2	3.61e+01(2.51e+01)-	3.40e+02(8.35e+01)-	3.38e+02(7.14e+01)-	2.72e+00(3.85e+00)~	1.02e+01(2.04e+01)
F3-T1	2.93e+02(3.06e+02)-	9.44e+01(6.06e+01)+	1.86e+02(5.14e+01)-	4.68e+02(8.46e+02)-	1.57e+02(5.66e+01)
F3-T2	6.41e+00(2.45e+00)-	6.31e+00(2.30e+00)-	4.31e+00(1.76e+00)~	4.96e+00(2.43e+00)~	3.51e+00(2.20e+00)
F4-T1	3.27e+02(9.32e+01)-	3.46e+02(6.43e+01)-	3.30e+02(7.93e+01)-	1.08e+02(2.00e+01)~	1.07e+02(1.90e+01)
F4-T2	7.33e+02(2.79e+02)~	7.46e+02(3.78e+02)~	2.36e+03(1.29e+03)-	6.92e+02(3.06e+02)~	8.03e+02(2.90e+02)
F5-T1	3.34e+02(9.54e+01)-	2.06e-01(9.18e-02)-	3.25e+02(9.61e+01)-	1.09e+02(2.25e+01)-	6.95e-02(5.36e-02)
F5-T2	3.21e-03(1.31e-03)-	2.63e-04(1.91e-04)-	4.27e-03(2.45e-03)-	3.48e-04(3.87e-04)-	1.56e-05(3.33e-05)
F6-T1	8.81e-01(3.60e-01)-	1.75e-02(4.32e-03)-	8.79e-01(3.58e-01)-	1.15e-01(1.68e-01)-	1.23e-02(5.04e-03)
F6-T2	4.10e-02(1.79e-02)-	1.64e-02(7.37e-03)-	4.41e-02(2.54e-02)-	1.75e-02(1.11e-02)-	8.35e-03(7.18e-03)
F7-T1	4.34e+02(9.63e+02)-	4.81e+01(1.18e-01)~	2.86e+02(2.73e+02)-	3.57e+02(6.53e+02)-	4.81e+01(1.02e-01)
F7-T2	6.35e+01(2.73e+01)-	2.79e-02(1.60e-02)-	4.83e+01(1.44e+01)-	4.39e+01(2.45e+01)-	1.38e-02(8.94e-03)
F8-T1	3.11e+02(1.11e+02)-	3.52e+02(7.15e+01)-	3.35e+02(9.73e+01)-	1.15e+02(1.97e+01)~	1.16e+02(3.07e+01)
F8-T2	1.07e+02(9.33e+01)~	1.35e+02(1.63e+02)~	1.52e+02(1.51e+02)~	9.48e+01(9.09e+01)~	1.54e+02(1.16e+02)
~/+/-	2/0/14	3/1/12	2/0/14	9/0/7	\

"+" (or "-") indicates the corresponding variant is better (or worse) than MFEA-DAE, and "~" indicates they obtain the statistically similar performance.

Table 4.6: Mean objective values and standard deviations obtained by MFEA-DAE and its variants on the first test suite.

Problem	Variant-I	Variant-II	Variant-III	MFEA-DAE
CIHS-T1	3.67e-03(1.20e-02)-	2.59e-04(8.24e-04)~	9.29e-04(2.25e-03)~	2.20e-03(5.47e-03)
CIHS-T2	2.12e+01(8.32e+01)-	4.17e-01(1.21e+00)~	9.41e-01(2.12e+00)~	2.69e+00(1.04e+01)
CIMS-T1	7.31e-02(2.58e-01)-	3.58e-03(2.41e-03)-	3.63e-03(2.00e-03)-	2.15e-03(1.37e-03)
CIMS-T2	1.71e+00(6.95e+00)-	1.09e-02(1.70e-02)-	1.02e-02(1.19e-02)-	3.89e-03(4.45e-03)
CILS-T1	2.61e+00(6.20e+00)-	1.35e-02(8.37e-03)~	1.48e-02(1.26e-02)~	1.29e-02(7.27e-03)
CILS-T2	5.12e+02(5.04e+02)-	8.57e-03(9.05e-03)~	1.21e-02(1.94e-02)~	7.34e-03(8.48e-03)
PIHS-T1	2.96e+02(1.15e+02)-	8.97e+01(2.52e+01)~	8.83e+01(1.65e+01)~	8.44e+01(1.76e+01)
PIHS-T2	7.23e-02(4.34e-02)-	3.50e-02(1.97e-02)-	2.52e-02(7.98e-03)~	2.25e-02(1.66e-02)
PIMS-T1	3.38e-01(3.29e-01)-	1.14e-01(1.50e-01)-	2.49e-01(3.00e-01)-	5.42e-02(7.54e-02)
PIMS-T2	9.45e+01(7.00e+01)-	6.46e+01(2.41e+01)-	6.39e+01(2.07e+01)-	4.96e+01(9.08e+00)
PILS-T1	1.73e-02(4.06e-03)-	1.50e-02(4.91e-03)-	1.59e-02(5.08e-03)-	1.04e-02(3.20e-03)
PILS-T2	1.24e-01(3.25e-02)-	9.72e-02(3.07e-02)-	1.16e-01(3.23e-02)-	7.22e-02(2.23e-02)
NIHS-T1	4.81e+01(8.85e-02)+	6.50e+01(3.86e+01)~	6.14e+01(3.06e+01)~	5.01e+01(1.01e+01)
NIHS-T2	1.63e-01(6.22e-02)+	1.23e+01(2.04e+01)-	1.37e+01(2.27e+01)-	1.22e+00(3.47e+00)
NIMS-T1	3.38e-02(1.58e-02)-	2.40e-02(7.03e-03)-	2.00e-02(7.23e-03)~	1.53e-02(6.75e-03)
NIMS-T2	2.59e+00(1.37e+00)-	1.44e+00(6.04e-01)-	1.51e+00(4.92e-01)-	1.12e+00(4.50e-01)
NILS-T1	3.60e+02(3.05e+01)-	1.04e+02(2.47e+01)~	1.12e+02(2.39e+01)-	9.12e+01(2.02e+01)
NILS-T2	1.62e+03(1.27e+03)-	7.47e+02(2.91e+02)~	9.15e+02(3.49e+02)~	8.71e+02(3.12e+02)
~/+/-	0/2/16	8/0/10	9/0/9	\

Table 4.7: Mean objective values and standard deviations obtained by MFEA-DAE and its variants on the second test suite.

Problem	Variant-I	Variant-II	Variant-III	MFEA-DAE
F1-T1	2.86e-02(1.49e-02)~	2.41e-02(7.34e-03)~	2.26e-02(9.05e-03)~	2.10e-02(6.42e-03)
F1-T2	1.14e+02(1.40e+02)-	2.77e+01(1.91e+01)~	3.34e+01(2.42e+01)~	2.54e+01(1.92e+01)
F2-T1	8.26e-01(4.62e-01)-	2.37e-01(3.04e-01)~	2.92e-01(3.18e-01)~	2.12e-01(2.69e-01)
F2-T2	2.45e+02(1.61e+02)-	4.25e+00(6.10e+00)~	4.87e+00(5.58e+00)~	8.44e+00(2.33e+01)
F3-T1	1.87e+02(8.73e+01)-	1.79e+02(4.48e+01)-	2.65e+02(5.31e+02)-	1.34e+02(5.06e+01)
F3-T2	4.05e+00(2.19e+00)-	4.09e+00(2.24e+00)-	3.98e+00(1.69e+00)-	2.94e+00(1.41e+00)
F4-T1	2.56e+02(1.34e+02)-	1.20e+02(2.52e+01)-	1.14e+02(2.38e+01)~	1.06e+02(2.35e+01)
F4-T2	9.28e+02(5.47e+02)~	8.99e+02(3.19e+02)~	7.86e+02(3.02e+02)~	1.00e+03(4.98e+02)
F5-T1	2.22e-01(1.15e-01)-	2.21e-01(1.12e-01)-	1.51e-01(1.06e-01)~	1.05e-01(7.80e-02)
F5-T2	3.46e-04(3.43e-04)-	3.08e-05(4.82e-05)~	1.37e-05(1.59e-05)~	2.97e-05(4.57e-05)
F6-T1	1.83e-02(5.20e-03)-	1.84e-02(6.84e-03)-	1.65e-02(6.32e-03)-	1.22e-02(3.86e-03)
F6-T2	1.67e-02(1.04e-02)-	1.38e-02(7.97e-03)-	1.33e-02(8.07e-03)-	6.35e-03(4.37e-03)
F7-T1	4.81e+01(1.54e-01)~	4.81e+01(1.53e-01)~	4.81e+01(1.20e-01)~	4.81e+01(1.24e-01)
F7-T2	2.63e-02(2.04e-02)-	2.64e-02(3.59e-02)~	2.04e-02(1.21e-02)~	1.65e-02(1.28e-02)
F8-T1	3.01e+02(1.15e+02)-	1.16e+02(2.39e+01)~	1.17e+02(2.25e+01)~	1.12e+02(2.65e+01)
F8-T2	1.22e+02(1.14e+02)~	9.87e+01(1.04e+02)~	7.90e+01(9.50e+01)~	1.60e+02(1.42e+02)
~/+/-	4/0/12	10/0/6	12/0/4	\

"+" (or "-") indicates the corresponding variant is better (or worse) than MFEA-DAE, and "~" indicates they obtain the statistically similar performance.

4.4.3 Further Study on the Effectiveness of DAE

1) Comparison with Single DA Method

As four domain adaptation (DA) methods are included in the $pool$ of DAE, MFEA-DAE is selected as one representative algorithm to compare with its four variants, i.e., MFEA-baseline, MFEA-AE, MFEA-KAE, and MFEA-AT, each of which uses only one DA method. As mentioned in the parameter settings, the common parameters are kept consistent. The effectiveness of DAE is validated, as it performs better or similarly on most test problems when compared with each of its variants. The results in Tab. 4.4 show the significant superiority of MFEA-DAE in most cases. On both tasks of CIHS and NIHS, MFEA-DAE is outperformed by MFEA-AE because AE has more effective transferability between two tasks with high similarity. Considering the performance on the second test suite, the experimental results in Tab. 4.5 also show the superiority of MFEA-DAE over other competitors. However, MFEA-DAE falls slightly behind MFEA-AT on the two tasks of F1 and F8. The degraded performance on these problems could be attributed to the ineffectiveness of other DA methods in the $pool$, leading to the waste of some computational resources for knowledge transfer across tasks.

2) Effectiveness of Two Auxiliary Components

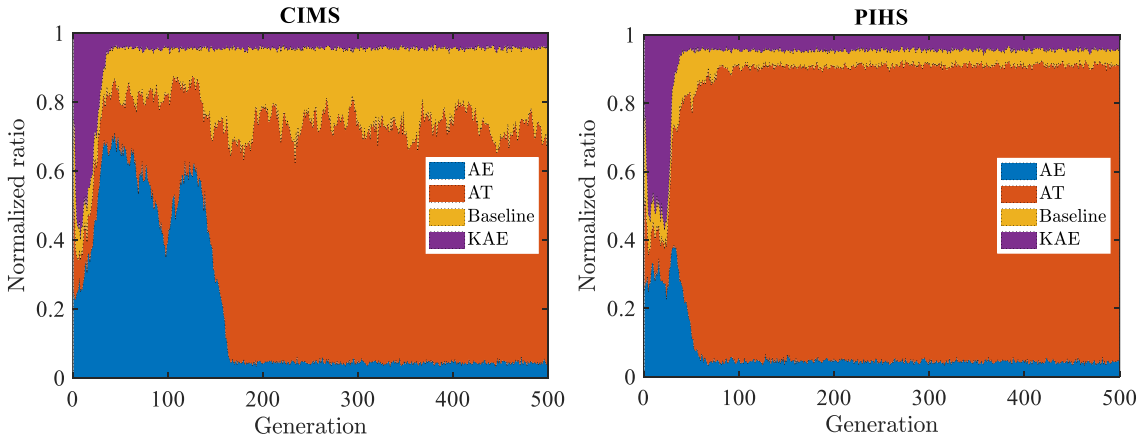
To validate the effectiveness of the two auxiliary components in MFEA-DAE, ablation experiments are conducted on two test suites by comparing DAE to its various variants. Here, three different variants of DAE (Variant-I, Variant-II, and Variant-III) are designed for performance comparison. The first variant (Variant-I) uses the RWS based on the quantified efficacy of DA methods to select one DA method in DAE,

while each DA method has an equal probability of being selected in the second variant (Variant-II). The detailed numerical results can be found in Tab. 4.6 and Tab. 4.7. Considering the first multitasking benchmark suite, MFEA-DAE outperforms Variant-I on 16 out of 18 cases, while it is outperformed by Variant-I on 2 cases. In addition, compared with Variant-II, MFEA-DAE obtains better and similar results on 10 and 8 cases, respectively. The superiority of MFEA-DAE can also be observed on another multitasking benchmark suite according to the comparison results. The comparisons not only validate the effectiveness of the RWS based on the quantified efficacy of DA methods but also show the necessity of combining random selection and the RWS to select suitable DA methods for knowledge transfer. Moreover, to further study and analyse the reasonability of combining two selection methods, another variant (Variant-III) is designed. Both random selection and the RWS are adopted in Variant-III and MFEA-DAE, while the strategies for their use are different. Specifically, Variant-III swaps the usage scenarios of two selection methods in MFEA-DAE. As summarized in Tab. 4.6 and Tab. 4.7, the comparison results of the two multitasking test suites show that the overall performance of MFEA-DAE is better than that of Variant-III, as MFEA-DAE always shows either better or similar results in each case when compared with its competitor.

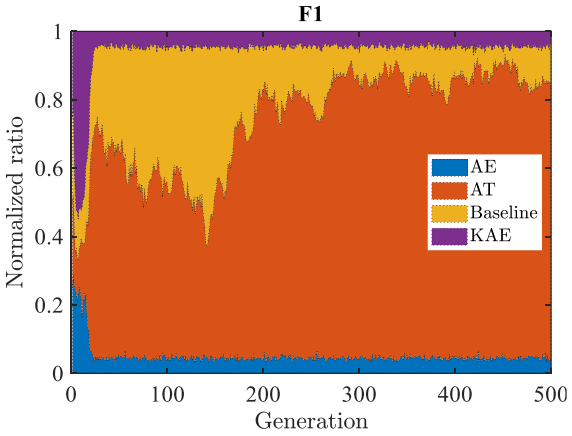
3) Further Discussion and Analysis

Moreover, to intuitively observe the dynamic selection behaviour in MFEA-DAE, the count of using each DA method is recorded at each generation. For clarity, their changing utilization ratios are calculated by normalizing the counts of using AE, KAE, AT, and the baseline method. As the evolutionary process progresses, their changing utilization ratios show that MFEA-DAE can effectively assign suitable DA methods

to candidate solutions. The utilization ratio curves are plotted for some representative test problems in Figs. 4.5 (a)-(c). As observed in Fig. 4.5 (a), at the earlier optimization stage of CIMS, the utilization ratio of AE is larger than that of the other methods, which means that AE plays a significant role in effective knowledge transfer. As the evolutionary process continues, AT and the baseline method begin to make more performance improvements than AE and KAE. In terms of PIHS in Fig. 4.5 (b), the utilization ratios of AE and KAE are larger when compared with the other two DA methods at the beginning of the evolutionary process, while AT plays a key role in effective knowledge transfer, as its utilization ratio is much higher than that of the others during the later evolutionary process. In addition, as shown in Fig. 4.5 (c), the utilization ratios of AT and the baseline method are relatively larger than those of AE and KAE on F1. The above observations show that MFEA-DAE can adaptively adjust the usage ratio of each DA method to realize effective knowledge transfer and mitigate negative transfer from ineffective DA methods to some extent during the evolutionary process.



(a) CIMS (b) PIHS



(c) F1

Figure 4.5: Normalized utilization ratios of different methods during the evolutionary search process on CIMS, PIHS, and F1.

4.4.4 Comparison with State-of-the-art EMT Algorithms

Here, five state-of-the-art EMT algorithms are further used for performance comparison, namely MFEA-II, MFEA-AKT, MTEA-AD, MTES, and MKTDE. Moreover, a single-task EA (SOEA), which solves each task separately, is used as the baseline to investigate the effectiveness of MFEA-DAE in solving the adopted MTOPs. The summarized results on the two test suites are collected in Tab. 4.8. Compared with SOEA, MFEA-DAE achieves better performance on most test problems, which demonstrates the superiority of multitasking optimization. However, all EMT algorithms are outperformed by SOEA on task 2 of F5 and F6 (F5-T2 and F6-T2), which shows that the occurrence of negative transfer causes their performance degradation in solving MTOPs. In MFEA-II, $rmp$is dynamically adjusted to alleviate negative transfer, while the adaptive configuration of multiple crossovers for knowledge transfer is designed in MFEA-AKT. Although both of them could improve multitasking performance to some extent, the obvious superiority of MFEA-DAE on almost all test

problems reveals that the proposed DAE can further enhance solution transferability across different tasks. In addition, MFEA-DAE performs better than MTEA-AD on most cases, while it is outperformed on only 5 cases. This demonstrates the superiority of DAE for improving multitasking optimization performance when compared with MTEA-AD. In addition, when compared with MFEA-DAE, MTES fails in all cases, even though it uses an approximated gradient to provide the search direction and dynamically adjusts the transfer intensity of information between tasks. The reason may be attributed to the problem of becoming trapped in a local optimum that gradient-based methods often suffer from, especially in problems with multimodality. Instead of transferring solutions, meta-knowledge that can evolve task-specific knowledge is shared across different tasks in MKTDE, which aims to optimize multiple tasks effectively. As shown in Tab. 4.8, MFEA-DAE achieves better performance in most cases, while MKTDE only obtains better results on five test problems (CIHS, CIMS, NIMS, F1, and F2). In fact, the effectiveness of MKTDE may be partly attributed to the stronger exploitation ability of differential evolution.

Table 4.8: Summarized results of MFEA-DAE and its competitors.

Algorithm Comparison	Test suite 1(~/+/-)	Test suite 2(~/+/-)
SOEA vs MFEA-DAE	1/0/17	1/2/13
MFEA-II vs MFEA-DAE	1/0/17	3/0/13
MFEA-AKT vs MFEA-DAE	0/0/18	0/0/16
MTEA-AD vs MFEA-DAE	0/1/17	0/4/12
MTES vs MFEA-DAE	0/0/18	0/0/18
MKTDE vs MFEA-DAE	3/6/9	2/4/10

“+”, “-”, and “~” indicate that the corresponding competitor is better than, worse than and similar to MFEA-DAE.

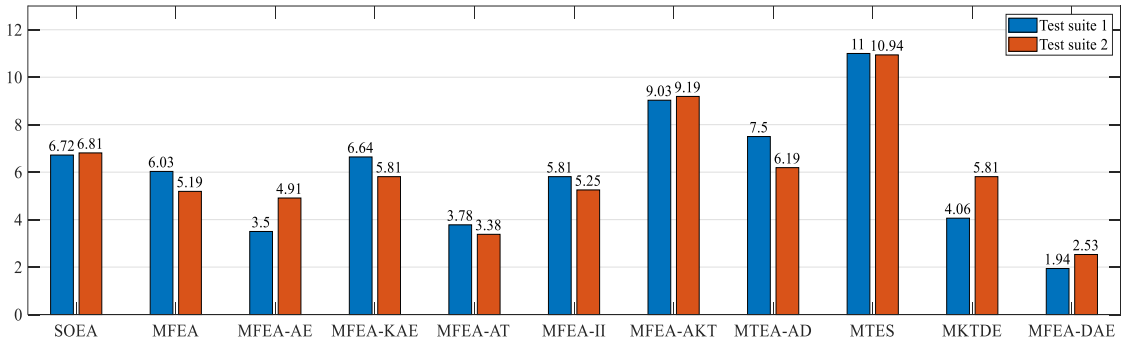


Figure 4.6: Average ranks of the test of Friedman for all compared algorithms.

Moreover, based on the test of Friedman [121], the average performance ranks of all compared algorithms on two test suites are plotted in Fig. 4.6, where blue and red bars represent the ranks on the first and second test suites, respectively. The ranks of MFEA-DAE on the two benchmark suites are smallest (1.94 and 2.53), which shows that the overall performance achieved by MFEA-DAE is the best on both test suites. According to the comparison results, it can be concluded that MFEA-DAE is very competitive when compared to these state-of-the art EMT algorithms on tackling the MTOPs adopted.

4.4.5 Parameter Sensitivity Analysis

The proposed DAE requires two additional parameters λ (determining the number of clusters in clustering candidate solutions) and β (determining the proportions of previous contribution and current contribution in quantifying the efficacy of DA methods). To study the impact of λ and β, the further comparative experiments are conducted by setting λ and β to different values. The summarized comparison results are collected in Tab. 4.9.

Table 4.9: Summarized results of MFEA-DAE with different parameter values.

Different parameter setting	Test suite 1(~/+/-)		Test suite 2(~/+/-)	
Gen	200	500	100	500
λ = 0.3 vs λ = 0.8	12/0/6	15/0/3	9/0/7	15/0/1
λ = 0.5 vs λ = 0.8	13/0/5	18/0/0	9/0/7	15/0/1
λ = 0.7 vs λ = 0.8	14/2/2	17/0/1	15/0/1	15/0/1
λ = 0.9 vs λ = 0.8	15/0/3	16/0/2	16/0/0	15/0/1
λ = 0.95 vs λ = 0.8	11/0/7	14/0/4	13/2/1	14/0/2
λ = 1.0 vs λ = 0.8	3/4/11	1/3/14	7/2/7	4/0/12
β = 0.0 vs β = 0.5	\	16/0/2	\	16/0/0
β = 0.3 vs β = 0.5	\	16/0/2	\	15/1/0
β = 0.7 vs β = 0.5	\	15/1/2	\	16/0/0
β = 1.0 vs β = 0.5	\	10/0/8	\	10/0/6
rmp = 0.1 vs rmp = 0.3	\	4/0/14	\	8/1/7
rmp = 0.5 vs rmp = 0.3	\	15/1/2	\	11/1/4
rmp = 0.7 vs rmp = 0.3	\	6/6/6	\	5/3/8
rmp = 0.9 vs rmp = 0.3	\	7/3/8	\	4/2/10

"+", "-", and "~" indicate that the corresponding variant is better than, worse than and similar to MFEA-DAE.

To study the impact of λ, MFEA-DAE with different λ values from {0.3, 0.5, 0.7, 0.8, 0.9, 0.95, 1.0} are compared. As shown in Tab. 4.9, compared with a smaller λ (e.g., 0.3 and 0.5), MFEA-DAE with λ = 0.8 speeds up the convergence on some test problems without any degradation on other test problems. This is because a too small number of clusters will give more opportunity for the candidate solutions to randomly select one DA method with equal probability. This lack of the effective competition among the DA methods will cause inefficient knowledge transfer in high probability. Compared to the variant with a larger λ (e.g., 0.9, 0.95), just the slight performance improvement can be observed on some test problems. This is because the efficacy is achieved by using the roulette wheel selection (RWS) strategy to select one DA method with higher quantified efficacy while the diversity is encouraged by using random selection (RS) strategy in DAE. As other DA methods with lower quantified efficacy also have the opportunity to be selected by using RWS strategy, the

effect of using RS to encourage the diversity of the DA methods on performance improvement is weaken to some extent, which leads to the situation that the overall performance improvement is not very significant in comparison to a larger λ. In addition, as the population converges to the global or local optimum as the evolutionary search process proceeds, the effect of cluster information with an optimal λ is further weakened during the later evolutionary stage. Although the cluster information is little in this case that λ is close to 1 (e.g., 0.9, 0.95), the gradually accumulative effect of the cluster information could achieve the similar performance to the optimal λ at the end of the evolutionary search process. Furthermore, MFEA-DAE with λ = 0.8 achieves the obvious performance improvement on most of test problems when compared with λ = 1.0, which indicates that the lack of diversity of the DA methods will miss the opportunity to exploit potential strengths of other DA methods. In summary, a too large or a too small λ value will cause the degradation of performance to some extent. Thus, setting λ around 0.8 is suggested in MFEA-DAE, which strikes the balance between efficacy and diversity of the DA methods.

In addition, to study the impact of β, MFEA-DAE with different β values from {0, 0.3, 0.5, 0.7, 1.0} are compared. As observed from Tab. 4.9, MFEA-DAE with β = 0.5 achieves better performance on 8 cases and 6 cases of two test suites when compared with β = 1.0 while its performance is similar to other values of β (e.g., 0, 0.3, and 0.7) on most of test problems. The comparison results show that the current contribution of DA methods on multitasking performance plays a critical role in the efficacy quantification of the DA methods. Thus, β is suggested to be set to a value less than 1 in MFEA-DAE. In this study, β is set to 0.5.

Moreover, to study the impact of the intensity of knowledge transfer on DAE, MFEA-DAE with different values of rmp from {0.1, 0.3, 0.5, 0.7, 0.9} are compared.

Keeping the same parameter setting introduced before, the summarized results are collected in Tab. 4.9, which show that MFEA-DAE with rmp = 0.3 achieves better performance on most cases when compared to MFEA-DAE with rmp = 0.1. The comparison with rmp = 0.3 indicates that a too small rmp could not provide sufficient genetic material exchange, which will degrade optimization performance. In addition, when rmp is increased from 0.3 to 0.5, they show similar performance on most cases. However, when rmp is further increased to 0.7 or 0.9, performance degradation is obvious on most cases, which indicates that the high intensity of knowledge transfer among tasks will waste some computational resources, especially at the later evolutionary stages. Thus, rmp is recommended as 0.3 in MFEA-DAE.

4.4.6 Computational Complexity Analysis

As the calculation of the mapping parameters from AE, KAE and AT ($\mathbf{M}$, $\mathbf{M}_k$, $\mathbf{A}$, and $\mathbf{b}$) can be expressed as a closed-form solution, their additional computational costs are from the matrix operations. Using the big O notation, the required time complexity of AE, KAE and AT can be expressed by $O(d^2N)$, $O(N^3)$, and $O(d^2N)$ where N is the population size and d is the number of decision variables. The DAE method includes three components (e.g., domain adaptation selection and two auxiliary components), where the computational complexity in one generation is mainly dominated by the two auxiliary components. Concretely, DAE needs a time complexity of $O(dN^2)$ to construct the neighborhood relation for the population and a time complexity of $O(N)$ to quantify the efficacy of various DA methods, respectively. Thus, the overall worst time complexity of DAE is $O(dN^2)$.

Moreover, the running times of all the compared EMT algorithms from 20 runs on

the first synthetic single-objective multitasking test suite are collected to compare their running efficiency. Here, various DA methods (AE, KAE and AT) and DAE are incorporated into MFEA for comparison, respectively. In addition, the ratios of running time of MFEA-DAE and other compared algorithms are also calculated to show the relative running speed of DAE against its competitors more intuitively. As observed from Tab. 4.10, the running speed of MFEA-DAE is slightly slower than MFEA, MFEA-AE, MFEA-KAE and MFEA-AT (the time ratios are 1.41, 1.37, 1.33 and 1.18, respectively), which show that DAE indeed does not bring much computational time cost when compared with each single domain adaptation method. In addition, MFEA-DAE achieves faster running speed when compared with MFEA-II while it is beaten by MFEA-AKT. In terms of the three algorithms implemented in their own frameworks (MTEA-AD, MTES and MKTDE), MFEA-DAE performs worse than them in the running speed. Actually, the difference of the basic EMT framework and the specific implementation could cause a large discrepancy of the running time. Based on the above theoretical complexity analyses and actual running time comparisons, it can be concluded that the extra computational burden of the use of DAE is affordable, and its actual running speed is very competitive when compared with its competitors based on the same EMT framework.

Table 4.10: Running times of all compared EMT algorithm.

Algorithm	Running time (s)	Time ratio (MFEA-DAE/competitor)
MFEA	141.31	1.41
MFEA-AE	144.76	1.37
MFEA-KAE	148.92	1.33
MFEA-AT	167.53	1.18
MFEA-II	629.26	0.32
MFEA-AKT	131.87	1.51
MTEA-AD	10.32	24.19
MTES	83.18	3.25
MKTDE	66.84	4.11
MFEA-DAE	196.11	\

4.4.7 Practical Case Study

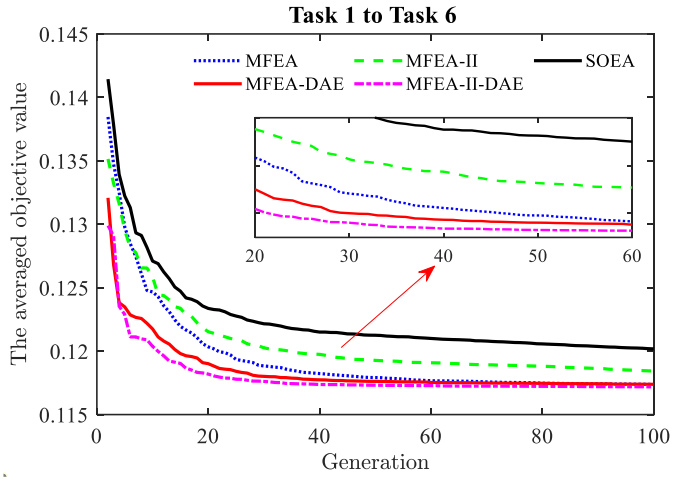
To further analyse the effectiveness of DAE in real-world applications, the parameterized planar kinematic arm problem [98] is used in the comparative experiments. The planar kinematic arm consists of d links with the same length L, which are connected by d joints with the same angle limit α_{max}. A task T_i is defined by the parameters L and α_{max}, which aims to optimize the angle of each joint $\boldsymbol{\alpha}^i = (\boldsymbol{\alpha}_1^i, \dots, \boldsymbol{\alpha}_d^i)$ to make the Euclidean distance between tip position $\mathbf{p}^d$ and a predefined target γ as close as possible. Using these notations, an MTOP with K tasks can be formulated as follows:

$$\min_{\boldsymbol{\alpha}^i} f_i(\boldsymbol{\alpha}^i; [L, \alpha_{max}]) = ||\mathbf{p}^d - \gamma||, i = \{1, \dots, K\}, \tag{4.33}$$

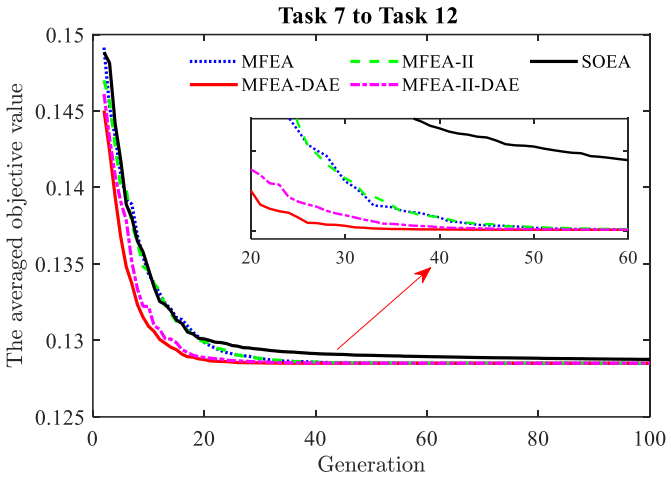
where the fitness function $f_i(\cdot)$ of each task is distinguished by a particular parameter combination $[L, \alpha_{max}]$, which determines the kinematics calculation of the arm. Note that the real-value encoding is used to represent the solutions. More details of the kinematics used to calculate $\mathbf{p}^d$ can be found in [98].

Table 4.11: Parameter settings of the practical optimization examples.

Parameterized Planar Arm Problem	Task Group 1: T_1 to T_6						Task Group 2: T_7 to T_{12}					
Task	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}
L (m)	0.10	0.10	0.10	0.10	0.10	0.10	0.08	0.08	0.07	0.07	0.06	0.06
α_{max} (rad)	0.80	0.75	0.70	0.65	0.60	0.55	0.40	0.35	0.40	0.35	0.40	0.35



(a) $\mathbf{T_1}$ to $\mathbf{T_6}$



(b) $\mathbf{T_7}$ to $\mathbf{T_{12}}$

Figure 4.7: Averaged convergence curves of MFEA, MFEA-II, MFEA-DAE, MFEA-II-DAE and SOEA.

As suggested in [119], $\boldsymbol{d}$ is set to 10 and $\boldsymbol{\gamma}$ is set to (0.5, 0.5) in this experiment. The MTOP with many tasks can be constructed by configuring each task with a particular combination $[L, \alpha_{max}]$. Here, using the same parameter settings in [56], two task groups are adopted as practical optimization examples, each of which includes six different tasks. The details of the parameter settings of the 12 tasks from the two

task groups can be seen in Tab. 4.11. In particular, one task group consists of tasks T_1 to T_6, while another task group includes tasks T_7 to T_{12}. Each task group is used as one MTOP to be solved by each algorithm in one run. In the experiment, the enhanced EMT algorithms with DAE (MFEA-DAE and MFEA-II-DAE) are compared with MFEA and MFEA-II on solving T_1 to T_6 and T_7 to T_{12}, respectively. In addition, SOEA is used as the baseline to solve each task separately, which aims to investigate the effectiveness of EMT algorithms. The comparison results are collected from 10 independent runs on each task group. Figs. 4.7(a) and (b) show the averaged convergence curves of all algorithms on T_1 to T_6 and T_7 to T_{12}. It is observed that all EMT algorithms achieve better average convergence performance than SOEA, and the enhanced EMT algorithms with DAE further improve the average convergence performance of MFEA and MFEA-II on T_1 to T_6 and T_7 to T_{12}. The above comparison results validate the effectiveness of DAE on the parameterized planar kinematic arm problem.

4.5 Conclusion

This chapter has proposed an effective DAE method to combine the strengths of various complementary DA methods by considering their efficacy and diversity, which can effectively assign suitable DA methods to candidate solutions. First, the neighbourhood relationship is constructed for all candidate solutions by using the hierarchical clustering method. Then, if none of the neighbours of two selected parent solutions has been selected for knowledge transfer, the efficacy of DA methods should be considered first by running roulette wheel selection based on their contributions to the

performance improvement in knowledge transfer. Otherwise, one DA method is randomly selected for knowledge transfer to emphasize the diversity of DA methods, which can potentially combine their advantages. In this way, both the efficacy and diversity of complementary DA methods have been considered to enhance the solution transferability across distinct tasks. The experimental results not only validated the effectiveness of the DAE method for knowledge transfer in EMT but also demonstrated the very competitive performance of MFEA-DAE when compared with other state-of-the-art EMT algorithms.

Chapter 5

Fuzzy Logic-Based Method for Adaptively Deciding When and How to Transfer in EMT

5.1 Introduction

On the one hand, in most existing EMT algorithms, the extent of knowledge transfer is predefined during the multitasking search process [36], [51]. For example, in MFEA and most of its variants [27], [60], [72], the prespecified random mating probability (rmp) was employed to determine the extent of knowledge transfer by allowing the solutions between different tasks to undergo crossover in a fixed probability. In addition, in [46], [48], [68], the extent of knowledge transfer was determined by

predefining the number of transferred solutions and the generation interval of knowledge transfer. In fact, the performance of EMT is intricately linked to the degree of underlying similarity among tasks, as evidenced in prior studies [49], [50]. Generally, the appropriate extent of knowledge transfer tends to vary when the tasks exhibit different degrees of similarity. For example, the appropriate transfer extent for highly similar tasks may be excessive in such scenarios that tasks are dissimilar or unrelated. In this case, the predefined extent of knowledge transfer will lead to ineffective knowledge exchange among unrelated tasks. To mitigate the risk of negative transfer, it is common practice to preset the transfer parameters to some small values. However, such parameter settings may potentially undermine the multitasking performance when solving highly similar tasks due to repeated searches to find their global optima [49], [50].

On the other hand, several advanced methods of knowledge transfer have been proposed to enhance knowledge transferability across tasks in solving various multitasking optimization problems (MTOPs) [36], [51]. For example, considering the tasks with different optima and dimensionalities, two strategies were proposed in G-MFEA [74], which facilitate effective knowledge transfer by translating and shuffling decision variables. In addition, to efficiently solve multiple tasks possessing unique properties, the denoising autoencoder (AE) [48] was proposed to build the mapping across tasks, which allows knowledge transfer to take place across multiple evolution mechanisms. However, in terms of some complex tasks, their solutions may be nonlinearly correlated. Hence, the kernelized AE (KAE) [68] was proposed to capture the nonlinearity between the solutions of two tasks by constructing the mapping in a reproducing kernel Hilbert space (RKHS). Moreover, for the tasks sharing different fitness landscapes, an affine transformation (AT) [67] was proposed to learn the mapping

between two tasks by building the probability representation models on their populations, which avoids the mismatch of solutions. However, these methods are specifically designed for the characteristics of their solved problems, severely limiting the effectiveness of knowledge transfer to their respective preferred problems [126].

Without any prior knowledge about the tasks, adaptive knowledge transfer (AKT) can further enhance the effectiveness and robustness of EMT by dynamically adapting the transfer extent and selecting promising transfer methods. Recently, several ATK approaches have been proposed for EMT by building complex mathematical models [49], [50], or formulating deterministic judgments [60], [126]. However, the available information or data are usually imprecise and fuzzy due to the uncertainty and randomness brought by the evolutionary mechanism [88]-[92]. Therefore, existing deterministic approaches may make unwise or wrong decisions, which will potentially lead to the performance slowdowns of EMT. Due to its superior uncertainty and noise-handling ability from the usage of human-like linguistic variables, fuzzy logic (FL) has attracted much attention [127], [128], [129], [130]. Furthermore, it has been employed to assist EAs to solve various complex problems with nonlinear constraints [131], time-varying optima [132], and many local optima [133], [134]. Inspired by the above studies, this chapter proposes a new MFEA with FL-based AKT for more effective and robust EMT. Different existing deterministic methods, it is the first attempt to employ FL to implement AKT for performing implicit EMT. The main contributions of this study are summarized as follows.

1) To effectively adapt the transfer extent along the multitasking search process, an FL-based parameter adaption mechanism is developed to dynamically adjust the value of the transfer parameter, thereby alleviating the risk of negative transfer.

2) To adaptively select the most promising method for knowledge transfer, an FL-based selection mechanism is developed to select the best transfer method from multiple candidates, thereby enhancing knowledge transferability across tasks.

3) By incorporating the above two FL-based mechanisms into an EMT framework, this study presents the implementation of a new MFEA (called MFEA-FLM). The experimental results validate the effectiveness of the proposed method and show the competitive performance of MFEA-FLM when compared with other EMT algorithms.

5.2 Background and Motivation

5.2.1 Mamdani Fuzzy Inference System

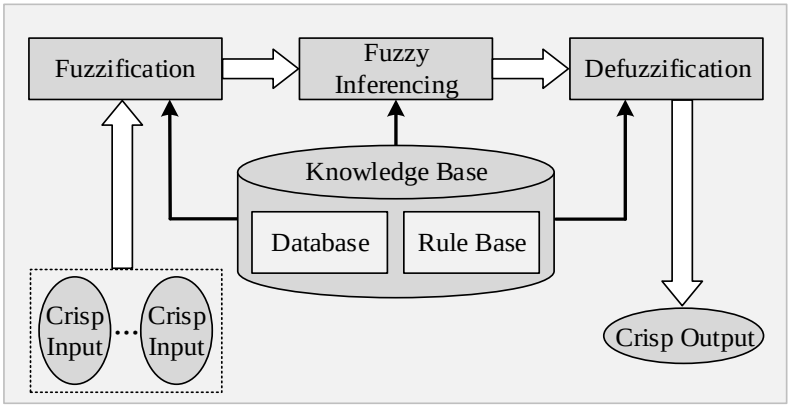


Figure 5.1: Framework of a Mamdani FIS with multiple inputs and one output.

The fuzzy inference system (FIS) is a type of artificial intelligence system, which utilizes fuzzy logic and fuzzy set theory to address the issue of imprecise information [135], [136]. Due to simplicity and interpretability, Mamdani FISs have been widely utilized in various practical applications [137], [138]. The framework of a Mamdani FIS with multiple inputs and one output is provided in Fig. 5.1. In general, the

knowledge base consists of a database and a rule base. The linguistic terms and their corresponding membership functions (MFs) are called the database, while the set of a series of IF-THEN rules is known as the rule base. For Mamdani FISs [139], the input-output relations are defined in the form of IF-THEN rules based on linguistic terms, which can be expressed as follows:

$$R^i: \begin{cases} \textbf{IF } v_1 \text{ is } L_1^i \textbf{ AND } v_2 \text{ is } L_2^i \textbf{ AND } \dots v_n \text{ is } L_n^i, \\ \textbf{THEN } y \text{ is } L_{n+1}^i \end{cases} \tag{5.1}$$

where R^i is the i-th IF-THEN rule, $\{L_1^i, L_2^i, \dots, L_n^i\}$ are the antecedent linguistic terms corresponding to n input variables $\{v_1, v_2, \dots, v_n\}$, and L_{n+1}^i is the consequent linguistic term related to the output variable y. These linguistic IF-THEN rules allow human experts within that domain to incorporate their knowledge and experience into the system in an effective manner [140]. In addition to the knowledge base, a Mamdani FIS involves three general steps, which are as follows:

Step 1 (Fuzzification): Converting crisp inputs into fuzzy sets through their corresponding MFs.

Step 2 (Fuzzy Inferencing): Mapping fuzzy inputs to fuzzy outputs by computing the firing strengths of activated rules. By aggregating the firing strengths of all the rules, the fuzzy output set can be obtained by

$$\mu(y) = \max_{i=1,\dots,|R|} \left\{ \min_{j=1,\dots,n} \left(\mu_{ij}(v_j), \mu_{i(n+1)}(y) \right) \right\}, \tag{5.2}$$

where $|R|$ is the number of rules, $\mu_{ij}(v_j)$ and $\mu_{i(n+1)}(y)$ are the MFs associated with L_j^i and L_{n+1}^i.

Step 3 (Defuzzification): Converting the fuzzy output set into the crisp output. The

center of area (COA) [140] is a commonly used defuzzification method, which computes the crisp output in the form of algebraic integration as follows:

$$\bar{y} = \frac{\int y\mu(y)dy}{\int \mu(y)dy}. \tag{5.3}$$

5.2.2 Motivation

In EMT, determining the appropriate transfer extent and the promising transfer method for conducting knowledge transfer across tasks are very challenging in handling MTOPs. Without any prior knowledge about the characteristics of tasks and their relationships, predefining the fixed transfer extent and adopting one certain transfer method cannot ensure the effectiveness and robustness of EMT on a wide range of test problems. On the one hand, dynamically adapting the extent of knowledge transfer can effectively alleviate the threat of negative transfer. On the other hand, collaboratively employing multiple methods of knowledge transfer with distinct complementarities can further enhance the knowledge transferability across tasks. While several AKT approaches have been proposed for EMT, they rarely tackle the above two challenges at the same time. More importantly, the uncertainty and randomness of the evolutionary mechanism will lead to the prevalence of imprecise information or data in the multitasking search process, which brings difficulties in developing reliable AKT approaches. Particularly, due to some random values in the reproduction operator, the offspring generated by the same parent population may have different objective values. Thus, the data or information by quantifying the performance improvement of offspring against their parents is usually imprecise and fuzzy. As a successful application of FL, the Mamdani FIS can incorporate expert knowledge into the model-building process through fuzzy sets and IF-THEN rules, enhancing the interpreta-

bility and flexibility of the models in practical applications. Moreover, due to its superior ability to handle uncertainty and noise, employing the Mamdani FIS can facilitate reliable and stable decision-making in complex and dynamic environments. Therefore, instead of building exact mathematical models or customizing deterministic statements, this study is motivated to employ FL to tackle the two challenges in EMT.

5.3 Methodology

5.3.1 Fuzzy Logic-Based Transfer Parameter Adaption

The transfer parameter determines the extent of conducting knowledge transfer among tasks during the evolutionary search process. In MFEA, the extent of knowledge exchange among tasks is controlled by a prespecified scalar parameter labelled as the random mating probability (rmp). It has a big impact on the effectiveness of the multitasking search process. However, choosing its appropriate value depends on the problem and may be difficult.

In this study, one Mamdani FIS with multiple inputs and one output is designed to estimate the change of the transfer parameter (i.e., Δrmp). For each generation (i.e., $\mathbf{g}$), the FIS takes the current value of the transfer parameter (i.e., rmp) and the quantified improvement value of knowledge transfer (i.e., $I^{\mathbf{g}}$) in terms of optimization performance as its two inputs. Note that the value of rmp is in the range of [0, 1]. The performance improvement along the evolutionary search process can be quantified by computing the improvement ratio of the objective values of offspring against their parents. Specifically, given the offspring and its parent population at the generation g, the normalized performance improvement brought by employing rmp to trigger knowledge transfer can be quantified by

$$\Delta I = \frac{\frac{1}{|\mathbf{Q}|}\sum_{\mathbf{x}\in\mathbf{Q}} \Delta F(\mathbf{x})}{\frac{1}{|\mathbf{O}|}\sum_{\mathbf{x}\in\mathbf{O}} \Delta F(\mathbf{x})}, \tag{5.4}$$

where $\mathbf{Q}$ and $\mathbf{O}$ are two solution sets. Specifically, $\mathbf{Q}$ consists of offspring generated by triggering knowledge transfer at the current generation, while $\mathbf{O}$ includes all offspring at the current generation. Here, $\Delta F(\mathbf{x})$ represents the improvement ratio of the objective values of offspring against its immediate parent, which is computed by

$$\Delta F(\mathbf{x}) = \sum_{i=1}^{m} \max\left\{\frac{f_i(\mathbf{p}) - f_i(\mathbf{x})}{f_i(\mathbf{p})}, 0\right\}, \tag{5.5}$$

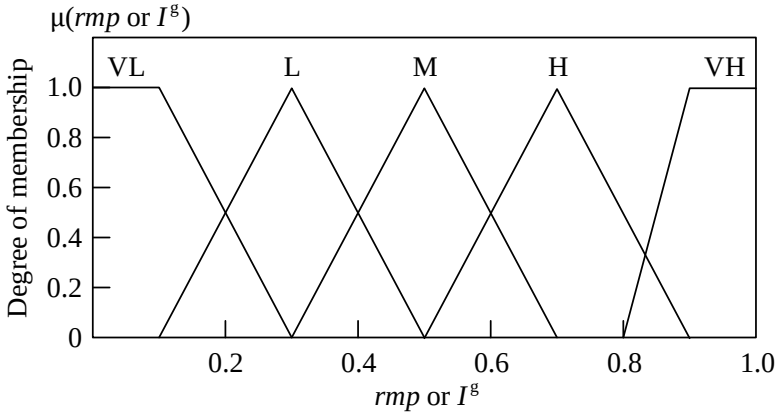
where $\mathbf{p}$ is the immediate parent of $\mathbf{x}$ and $f_i(\cdot)$ is the i-th objective value. To better represent the performance improvement brought by knowledge transfer, the preference coefficient α is introduced to determine the proportions of the performance improvement at the previous and current generations. Thus, the quantified performance improvement can be computed by

$$I^{\mathrm{g}} = \begin{cases} 1/2, & \text{if } \mathrm{g} = 1 \\ \alpha I^{\mathrm{g}-1} + (1-\alpha)\Delta I, & \text{if } \mathrm{g} \geq 2 \end{cases}, \tag{5.6}$$

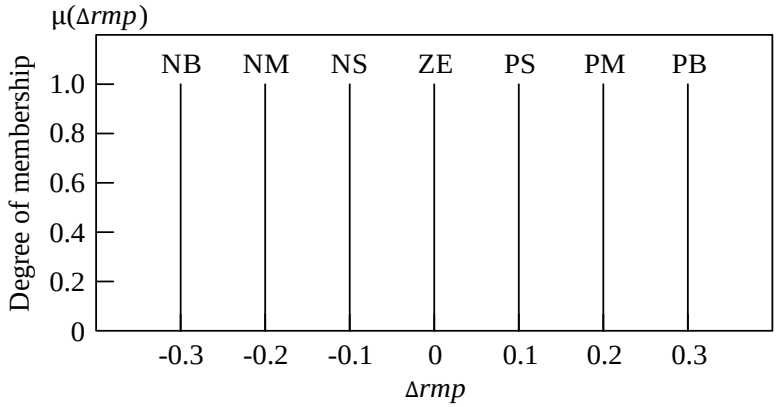
where α is a value in [0, 1] and it can be adjusted based on the specific requirement.

Now, the two inputs, i.e., rmp and I^{g}, are in [0, 1]. Next, the fuzzy partitions of the antecedent space and consequence space are defined. Specifically, for rmp and I^{g}, there are five fuzzy sets, which denote very low (VL), low (L), medium (M), high (H), and very high (VH), respectively. Here, L, M, and H are defined by three membership functions of class triangular with their associated parameter tuples being (0.1, 0.3, 0.5), (0.3, 0.5, 0.7), and (0.5, 0.7, 0.9), respectively. Besides, VL and VH are defined

by the membership function of class L with the parameter tuple (0.0, 0.0, 0.1, 0.3) and the membership function of class γ with the parameter tuple (0.7, 0.9, 1.0, 1.0), respectively. For Δrmp, there are seven fuzzy sets, which denote negative big (NB), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), and positive big (PB), respectively. Here, the seven fuzzy sets are defined by seven singleton membership functions with the parameter being -0.3, -0.2, -0.1, 0, 0.1, 0.2, and 0.3, respectively. Fig. 5.2 shows the graphical illustration of the fuzzy sets defined for each input and output.



(a) Membership functions of rmp and I^g



(b) Membership functions of Δrmp

Figure 5.2: Graphical illustration of membership functions applied in the fuzzy inference system for transfer parameter adaption.

Table 5.1: Set of rules of fuzzy inference system for estimating the change in the transfer parameter.

Δrmp		Performance Improvement (I^g)				
		VL	L	M	H	VH
Transfer Parameter (rmp)	VL	ZE	PS	PM	PB	PB*(#1)
	L	NS	ZE	PS	PM	PB
	M	NM	NS	ZE	PS	PM
	H	NB	NM	NS	ZE	PS
	VH	NB*(#2)	NB	NM	NS	ZE

In the FIS, the inputs are first converted from real values into fuzzy values, and then the fuzzy inference process is performed based on a set of fuzzy rules as shown in Tab. 5.1. To illustrate the fuzzy inference process, two fuzzy rules marked with the star symbol are given as follows:

Rule #1: IF rmp is VL and I^g is VH, THEN Δrmp is PB.

Rule #2: IF rmp is VH and I^g is VL, THEN Δrmp is NB.

In Rule #1, the extent of knowledge transfer is very low, while its resultant performance improvement is very high. In such a case, conducting knowledge transfer across tasks can bring significant performance improvement during the multitasking search process. Thus, the extent of knowledge transfer should be significantly increased, which allows more frequent positive knowledge exchange among tasks in the subsequent evolutionary search process. On the contrary, in Rule #2, the extent of knowledge transfer is very high, while its resultant performance improvement is very low. In such a situation, conducting knowledge transfer contributes little to the performance improvement in optimizing multiple tasks. Thus, the extent of knowledge transfer should be significantly decreased to reduce negative transfer.

After the fuzzy inference process, the obtained results are a series of fuzzy values. Finally, by applying the defuzzification operator in Eq. (5.3), the crisp value of Δrmp is computed as the final output of the FIS. Thus, the extent of knowledge transfer can be dynamically adjusted by modifying the value of Δrmp according to the following formula:

$$rmp = \begin{cases} 0.3, & \text{if g} = 1 \\ rmp + \Delta rmp, & \text{if g} \geq 2 \end{cases}. \tag{5.7}$$

5.3.2 Fuzzy Logic-Based Transfer Method Selection

In terms of a $pool$ consisting of k complementary transfer methods, which are denoted by $pool = \{\mathcal{TM}_1, \mathcal{TM}_2, \ldots, \mathcal{TM}_k\}$, they have different biases in conducting knowledge transfer among tasks. One specific transfer method can show superiority in its preferred transfer scenario, while it has poor performance in other scenarios. The lack of prior knowledge of back-box optimization problems causes difficulty in determining the appropriate method for knowledge transfer. Thus, adaptively selecting promising methods for knowledge transfer is very critical to improving the optimization performance and efficacy in handling multiple tasks. The current applicability of one specific method for knowledge transfer should not only depend on its efficacy quantified by its actual performance during the multitasking search process but also should have sensitivity to the number of triggering it in the current population.

In this study, another Mamdani FIS with multiple inputs and one output is designed to estimate the applicability of each transfer method for the current transfer scenario. In this way, the most promising method can be selected from multiple candidates by comparing the estimated values of their applicability. Specifically, the normalized ef-

ficacy and the usage ratio of one transfer method are considered as two inputs of the FIS. The normalized efficacy of one transfer method can be reflected by quantifying how much it contributes to accelerating the evolutionary search process. Considering one transfer method, i.e., $\mathcal{TM}_i$, the ratio of its efficacy can be computed by

$$\Delta E_i = \frac{\frac{1}{|\mathbf{S}_i|}\sum_{\mathbf{x}\in\mathbf{S}_i}\Delta F(\mathbf{x})}{\frac{1}{|\mathbf{S}_1 \cup \ldots \cup \mathbf{S}_k|}\sum_{\mathbf{x}\in\mathbf{S}_1\cup\ldots\cup\mathbf{S}_k}\Delta F(\mathbf{x})}, \tag{5.8}$$

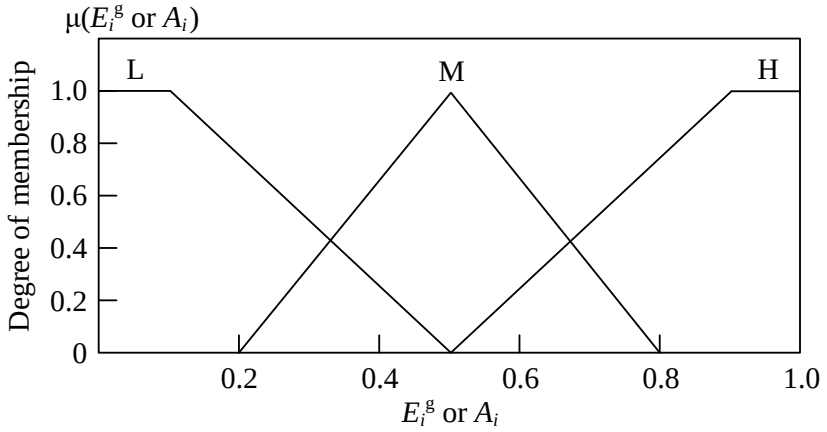
where $\mathbf{S}_i$ denotes the solution set. Here, $\mathbf{S}_i$ consists of all offspring solutions that are generated by performing $\mathcal{TM}_i$ for knowledge transfer at the current generation. Here, $\Delta F(\mathbf{x})$ represents the improvement ratio of the objective values of the offspring in $\mathbf{S}_i$ against the immediate parent, which is computed by Eq. (5.8). The normalized efficacy of performing $\mathcal{TM}_i$ for knowledge transfer can be defined by

$$E_i^{\mathrm{g}} = \begin{cases} 1/k, & \text{if } \mathrm{g} = 1 \\ \alpha E_i^{\mathrm{g}-1} + (1-\alpha)\Delta E_i, & \text{if } \mathrm{g} \geq 2 \end{cases}, \tag{5.9}$$

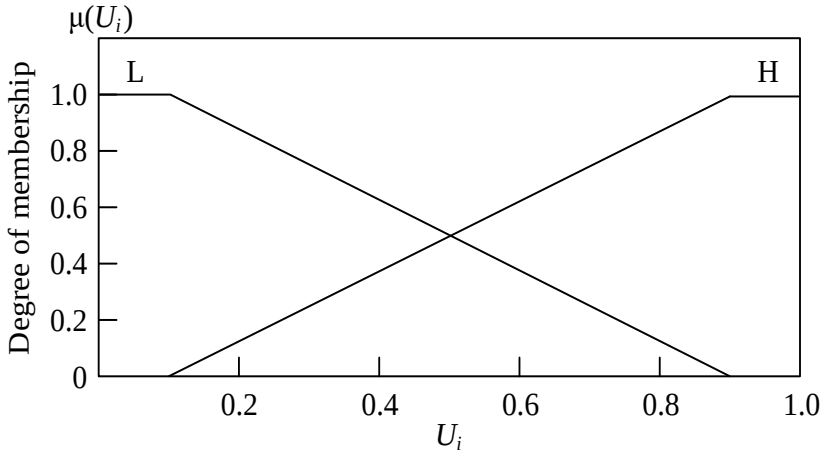
where α is a preference coefficient to determine the proportions of the previous efficacy and current efficacy of $\mathcal{TM}_i$ in estimating E_i^{g}. Note that ΔE_i is set to $1/k$ for all $i = \{1, \cdots, k\}$ if $\Delta F(\mathbf{x})$ is 0 for all $\mathbf{x} \in \mathbf{S}_1 \cup \ldots \cup \mathbf{S}_k$, where k is the number of all candidates in the $pool$. In addition, the usage ratio of triggering $\mathcal{TM}_i$ for knowledge transfer is computed as follows:

$$U_i = \begin{cases} 0, & \text{if } \sum_{i=1}^{k} n_i = 0 \\ \frac{n_i}{\sum_{i=1}^{k} n_i}, & \text{otherwise} \end{cases}, \tag{5.10}$$

where n_i denotes the number of triggering $\mathcal{TM}_i$ for knowledge transfer at the current generation.



(a) Membership functions of $\mathbf{E_i^g}$ and $\mathbf{A_i}$



(b) Membership functions of $\mathbf{U_i}$

Figure 5.3: Graphical illustration of membership functions applied in the fuzzy inference system for transfer method selection.

Now, the two inputs, i.e., E_i^g and U_i, are in [0, 1]. Then, the fuzzy partitions for the antecedent space and consequence space are defined. In terms of E_i^g, there are three fuzzy sets, which denote low (L), medium (M), and high (H). Here, L, M, and H are defined by three membership functions, i.e., class L with the parameter tuple (0.0, 0.0, 0.1, 0.5), class triangular with the parameter tuple (0.2, 0.5, 0.8), and class γ with the parameter tuple (0.5, 0.9, 1.0, 1.0), respectively. For U_i, there are two fuzzy

sets, which denote low (L) and high (H) by employing two membership functions, i.e., class L with the parameter tuple (0.0, 0.0, 0.1, 0.9) and class γ with the parameter tuple (0.1, 0.9, 1.0, 1.0), respectively. The output is the current applicability of the method (i.e., A_i). Here, there are three fuzzy sets, which denote low (L), medium (M), and high (H). Their membership functions are defined to be the same as that of $E_i^{\mathbf{g}}$. Fig. 5.3 shows the graphical illustration of the fuzzy sets defined for each input and output.

Table 5.2: Set of rules of fuzzy inference system for estimating the applicability of each transfer method.

Rule No.	Normalized Efficacy ($E_i^{\mathbf{g}}$)	Usage Ratio (U_i)	Applicability (A_i)
1	Low	Low	Medium
2	Low	High	Low
3	Medium	-	Medium
4	High	Low	High
5	High	High	Medium

Taking the fuzzy values of $E_i^{\mathbf{g}}$ and U_i as the inputs, the fuzzy inference process is performed based on a series of fuzzy rules as shown in Tab. 5.2. As displayed in Rule 1, when both the normalized efficacy of $\mathcal{TM}_i$ and its usage ratio are low, the diversity should be encouraged when selecting the method for knowledge transfer. Thus, the applicability of $\mathcal{TM}_i$ is medium, which allows $\mathcal{TM}_i$ to be competitive as the promising method. In terms of Rules 2, 3, and 4, the applicability of $\mathcal{TM}_i$ completely depends on its normalized efficacy regardless of its usage ratio. In these situations, the efficacy of the method for knowledge transfer should be a priority, thereby encouraging the use of the method with better efficacy for conducting knowledge trans-

fer. However, the high efficacy of $\mathcal{TM}_i$ may be attributed to its frequent use rather than its true applicability for knowledge transfer. To alleviate this issue, Rule 5 is given by setting the applicability to medium rather than high.

With the above fuzzy rules, a series of fuzzy values are obtained by performing the fuzzy inference process. By applying the defuzzification operator by Eq. (5.3), the crisp variable of A_i is computed as the final output of the FIS. After obtaining the values of the applicability of all candidate methods, the most promising method of knowledge transfer (i.e., $\mathcal{TM}_{index}$) from $\{\mathcal{TM}_1, \ldots, \mathcal{TM}_k\}$ is identified by the following formula:

$$index = \begin{cases} r \in \{1, \ldots, k\}, & \text{if } \mathrm{g} = 1 \\ \underset{i \in \{1, \ldots, k\}}{\arg\max}\{A_i\}, & \text{if } \mathrm{g} \geq 2, \end{cases} \tag{5.11}$$

where r is an integer randomly selected from $\{1, \cdots, k\}$ when g is 1.

5.3.3 Main Framework

The details of MFEA-FLM is presented in this subsection. To clarify the running of MFEA-FLM, its pseudocode is provided in Algorithm 5.1 with the inputs: an MTOP with K tasks, N (the size of the population), G_{max} (the preset maximum number of generations), α (the preference coefficient), and $pool = \{\mathcal{TM}_1, \mathcal{TM}_2, \ldots, \mathcal{TM}_k\}$ (the set of multiple candidate methods of knowledge transfer). As studied in [31], AE [25], KAE [29], and AT [30] have strong complementarities in transferring the solutions across tasks. Thus, they are selected as candidate methods for knowledge transfer. Besides, one baseline method that directly transfers the solutions among tasks without adaptation is also included in the $pool$.

Algorithm 5.1 The Main Framework of MFEA-FLM

Input: an MTOP with K tasks, N, G_{max}, α, $pool = \{\mathcal{TM}_1, \mathcal{TM}_2, \ldots, \mathcal{TM}_k\}$
Output: **P**

1 Initialize **P** to have $N \times K$ solutions
2 Assign skill factor τ_i to every $\mathbf{p}_i$ in **P** by Eq. (5.12) and evaluate them
3 Set g = 1, $E_i^{\mathrm{g}} = 1/k$ for all i = {1, 2, ⋯, k}, and I^1 = 1/2
4 **while** g ≤ G_{max}
5 Set $\mathbf{O} = \emptyset$, $\mathbf{S}_i = \emptyset$, and n_i = 0 for all i = {1, 2, ⋯, k}
6 rmp ← Adapt the transfer parameter by Eq. (5.7) // subsection 5.3.1
7 **while** $|\mathbf{O}| < N \times K$
8 Randomly select two parents [$\mathbf{p}_a$, $\mathbf{p}_b$] with $\tau_a \neq \tau_b$ from **P**
9 **if** $rand < rmp$
10 $\mathcal{TM}_{index}$ ← Select one from $pool$ by Eq. (5.11) // subsection 5.3.2
11 $\mathbf{p}_b'$ ← Transfer $\mathbf{p}_b$ to task τ_a via $\mathcal{TM}_{index}$
12 $\mathbf{c}_a$ ← *Inter-task crossover* + *mutation* ($\mathbf{p}_a$, $\mathbf{p}_b'$)
13 $\mathbf{p}_a'$ ← Transfer $\mathbf{p}_a$ to task τ_b via $\mathcal{TM}_{index}$
14 $\mathbf{c}_b$ ← *Inter-task crossover* + *mutation* ($\mathbf{p}_b$, $\mathbf{p}_a'$)
15 Each offspring is randomly assigned skill factor τ_a or τ_b
16 $\mathbf{S}_{index} = \mathbf{S}_{index} \cup [\mathbf{c}_a, \mathbf{c}_b]$, $n_{index} = n_{index} + 1$
17 Compute $\{U_1, \cdots, U_k\}$ by Eq. (5.10)
18 **else**
19 Randomly select one parent $\mathbf{p}'$ with skill factor τ_a from **P**
20 $\mathbf{c}_a$ ← *Intra-task crossover* + *mutation* ($\mathbf{p}_a$, $\mathbf{p}'$)
21 Randomly select one parent $\mathbf{p}''$ with skill factor τ_b from **P**
22 $\mathbf{c}_b$ ← *Intra-task crossover* + *mutation* ($\mathbf{p}_b$, $\mathbf{p}''$)
23 Assign $\mathbf{c}_a$ skill factor τ_a, and $\mathbf{c}_b$ skill factor τ_b, respectively
24 **end**
25 Mark $\mathbf{p}_a$ or $\mathbf{p}_b$ as immediate parents of $\mathbf{c}_a$ and $\mathbf{c}_b$
26 Evaluate [$\mathbf{c}_a$, $\mathbf{c}_b$] for their assigned skill factors only
27 $\mathbf{O} = \mathbf{O} \cup [\mathbf{c}_a, \mathbf{c}_b]$
28 **end**
29 $\mathbf{Q} = \mathbf{S}_1 \cup \ldots \cup \mathbf{S}_k$
30 Compute $I^{\mathrm{g}+1}$ and $\{E_1^{\mathrm{g}+1}, \ldots, E_k^{\mathrm{g}+1}\}$ by Eq. (5.6) and Eq. (5.9), respectively
31 **P** ← Environmental Selection on $\mathbf{P} \cup \mathbf{O}$
32 g = g + 1
33 **end**
34 **return P**

As shown in Line 1, a single population **P** is first formed by randomly sampling

$N \times K$ solutions in the unified search space $Y \in [0, 1]^{D_{max}}$, where D_{max} is the maximal dimensionality among K tasks. Then, in Line 2, the skill factor τ_i of each solution $\mathbf{p}_i$ in $\mathbf{P}$ is randomly assigned according to

$$\tau_i = mod(i, K) + 1. \tag{5.12}$$

After that, all solutions in $\mathbf{P}$ will be evaluated for their assigned skill factor only. In Line 3, the generation counter $\mathbf{g}$ is set to 1, $E_i^{\mathbf{g}}$ is initialized to $\mathbf{1}/k$ for all i = {1, 2, ⋯, k}, and $I^{\mathbf{1}}$ is initialized to 1/2.

The main evolutionary process is shown in Lines 4-33. A solution set $\mathbf{O}$ is used to collect all generated offspring at the current generation. Each $\mathcal{TM}_i$ is configured with one set and a counter, i.e., $\mathbf{S}_i$ and n_i. Here, $\mathbf{S}_i$ collects all offspring whose parents are generated by performing $\mathcal{TM}_i$. Additionally, n_i records the number of triggering $\mathcal{TM}_i$ for knowledge transfer. Before the start of each generation, $\mathbf{O}$ and $\mathbf{S}_i$ are set to empty, while n_i is set to 0 for all i = {1, ⋯, k} in Line 5. Then, in Line 6, the transfer parameter rmp is adapted by Eq. (5.7), which is elaborated in detail in subsection 5.3.1. After that, $N \times K$ offspring are generated by randomly selecting parents from $\mathbf{P}$. Specifically, for each pair of parents [$\mathbf{p}_a$, $\mathbf{p}_b$] with $\tau_a \neq \tau_b$, the component of knowledge transfer will be triggered when $rand < rmp$ ($rand$ is a random number in [0, 1]), as shown in Lines 10-17. First, the most promising method $\mathcal{TM}_{index}$ is selected from the $pool$ by Eq. (5.11), which is elaborated in detail in subsection 5.3.2. Next, $\mathcal{TM}_{index}$ is employed to transfer $\mathbf{p}_b$ and $\mathbf{p}_a$ to two different tasks (i.e., τ_a and τ_b), respectively. The corresponding transferred solutions of $\mathbf{p}_b$ and $\mathbf{p}_a$ are denoted by $\mathbf{p}_b{}'$ and $\mathbf{p}_a{}'$. Suppose $\mathcal{TM}_1$, $\mathcal{TM}_2$, $\mathcal{TM}_3$, and $\mathcal{TM}_4$ represent AE, KAE, AT, and Baseline, respectively. Then, $\mathbf{p}_a{}'$ is derived from $\mathbf{p}_a$ via $\mathcal{TM}_{index}$ as follows:

$$\mathbf{p}_a{}' = \begin{cases} \mathbf{M}\mathbf{p}_a, & \text{if } index = 1 \\ \mathbf{M}_k\mathbf{K}(\mathbf{P}, \mathbf{p}_a), & \text{if } index = 2 \\ \mathbf{p}_a\mathbf{A} + \mathbf{b}, & \text{if } index = 3 \\ \mathbf{p}_a, & \text{if } index = 4 \end{cases} \tag{5.13}$$

where $\mathbf{M}$, $\mathbf{M}_k$, and $[\mathbf{A}, \mathbf{b}]$ are the mappings of AE, KAE, and AT, respectively. Here, $\mathbf{K}(\cdot,\cdot)$ is the matrix where the (i,j)-th element is computed by the polynomial kernel function [29]. The details of expressing these mappings in the closed-form solutions can be found in [25], [29], and [30], respectively. Similarly, $\mathbf{p}_b{}'$ is derived from $\mathbf{p}_b$ via $\mathcal{TM}_{index}$ in the same manner by Eq. (5.13). Then, two offspring $\mathbf{c}_a$ and $\mathbf{c}_b$ are generated by applying inter-task crossover and mutation on two pairs of parents, i.e., $[\mathbf{p}_a, \mathbf{p}_b{}']$ and $[\mathbf{p}_b, \mathbf{p}_a{}']$, respectively. In this case, the skill factor τ_a or τ_b is randomly assigned to $\mathbf{c}_a$ and $\mathbf{c}_b$. Moreover, $\mathbf{c}_a$ and $\mathbf{c}_b$ are added into Sindex while nindex is increased by 1 due to the triggering of $\mathcal{TM}_{index}$. The usage ratios of all methods $\{U_1, \cdots, U_k\}$ are computed by Eq. (5.10). Otherwise, as shown in Lines 19-23, $\mathbf{c}_a$ and $\mathbf{c}_b$ are generated by performing intra-task crossover and mutation on two pairs of parents, i.e., $[\mathbf{p}_a, \mathbf{p}']$ and $[\mathbf{p}_b, \mathbf{p}'']$. Here, $\mathbf{p}'$ and $\mathbf{p}''$ are randomly selected from $\mathbf{P}$, which have the same skill factor with $\mathbf{p}_a$ and $\mathbf{p}_b$, respectively. Thus, $\mathbf{c}_a$ and $\mathbf{c}_b$ can directly imitate the skill factors of their parents. In this study, simulated binary crossover (SBX) [94] and polynomial-based mutation (PM) [95] are suggested. After generating $\mathbf{c}_a$ and $\mathbf{c}_b$, $\mathbf{p}_a$ or $\mathbf{p}_b$ having the same skill factor as each offspring is marked as its immediate parent in Line 25, which aims to quantify the improvement ratio of the objective values of offspring against its parent. Next, both $\mathbf{c}_a$ and $\mathbf{c}_b$ are evaluated based on their assigned skill factors only and then added into $\mathbf{O}$ in Lines 26-27. The above procedures in Lines 7-28 will be run iteratively until the number of offspring in $\mathbf{O}$ reaches $N \times K$.

After that, in Line 29, the solution sets, i.e., $\mathbf{S}_1$, $\cdots$, and $\mathbf{S}_k$, are combined to form

the solution set $\mathbf{Q}$, which is used to quantify the performance improvement brought by performing knowledge transfer. Then, I^{g+1} and $\{E_1^{g+1}, ..., E_k^{g+1}\}$ are computed by Eq. (5.6) and Eq. (5.9), respectively. Next, in Lines 31-32, the next population $\mathbf{P}$ is formed by selecting top $N \times K$ solutions from the combination set of $\mathbf{P}$ and $\mathbf{O}$ based on scalar fitness and $\mathbf{g}$ is increased by 1. The above evolutionary process in Lines 4-33 will be run when $\mathbf{g}$ is no more than G_{max}. Otherwise, the final population $\mathbf{P}$ is returned in Line 34.

5.4 Experimental Study

5.4.1 Experimental Settings

1) Test Problems

First, two multitasking test suites are employed as the test problems for performance comparison in the experiments. The first test suite includes nine MTOPs (i.e., CIHS, CIMS, CILS, PIHS, PIMS, PILS, NIHS, NIMS, and NILS) [96]. In addition, the second test suite includes eight MTOPs (i.e., F1-F8), which explicitly possess heterogeneous features on the problem dimensionality and the location of the global optima [74], [126]. The dimensions of the search spaces of two tasks are equal, while they are unequal on the rest of the problems. Additionally, two global optima are the same on F1, F2, F5, and F6, while they are different on the rest of the problems. The details of F1-F8 can be found in [126]. Furthermore, the effectiveness of the proposed method in handling some more complex problems is also studied. Therefore, one test suite including ten complex MTOPs from CEC 2021 Competition on Evolutionary Multitask Optimization[1] and another test suite consisting of nine multiobjective MTOPs

[117] are considered as the test problems for effectiveness validation. More details of the two test suites are provided in subsection 5.4.6.

Table 5.3: Parameter settings of all compared algorithms.

Algorithm	Parameter settings
MFEA	$N = 100,\ rmp = 0.3$
MFEA-II	$N = 100$
MFEA-AKT	$N = 100,\ rmp = 0.3$
MFEA-AE	$N = 100,\ rmp = 0.3,\ S = 100$
MFEA-KAE	$N = 100,\ rmp = 0.3,\ S = 100$
MFEA-AT	$N = 100,\ rmp = 0.3,\ \alpha = 0.5$
MFEA-DAE	$N = 100,\ rmp = 0.3,\ \lambda = 0.8,\ \beta = 0.5$
MKTDE	$N = 100,\ F = 0.5,\ CR = 0.6$
LCB-EMT	$N = 100,\ TG = 50,\ N_t = 50\times \lambda_g$
MFEA-FLM	$N = 100,\ \alpha = 0.1$

2) Parameter Settings

The common parameters are set to the same for all compared algorithms. Specifically, the population size (N) for each task is set to 100. The maximum number of function evaluations is set to 40 000 for each test problem. Additionally, SBX and PM are employed as the evolutionary operator for generating the offspring during the evolutionary search process for all compared algorithms except MKTDE [120]. In SBX, two parameters, i.e., p_c and η_c are set to 1.0 and 15, respectively. In terms of PM, two parameters, i.e., p_m and η_m are set to $1/d$ (d is the number of decision variables) and 15, respectively.

[1]http://www.bdsc.site/websites/MTO_competition_2021/MTO_Competition_CEC_2021.html

The private parameters of these algorithms are listed in Tab. 5.3. In MFEA and its variants (i.e., MFEA-AKT, MFEA-AE, MFEA-KAE, MFEA-AT, and MFEA-DAE [126]), rmp is 0.3. For MFEA-AE and MFEA-KAE, the numbers of sampled solutions (S) for learning $\mathbf{M}$ and $\mathbf{M}_k$ offline are set to 100. In MFEA-AT, the preference coefficient α is set to 0.5 for learning the mapping parameters [$\mathbf{A}$, $\mathbf{b}$]. In MFEA-DAE, two parameters, i.e., β and λ, are 0.5 and 0.8, respectively. MKTDE adopts DE, where two parameters (i.e., F and CR in DE) is set to 0.5 and 0.6, respectively. For LCB-EMT [76], the transfer interval generation (TG) is 50 while the maximal number of transferred solutions (N_t) is $50 \times \lambda_{\mathrm{g}}$ (λ_{g} is the similar factor at the current generation g). In terms of MFEA-FLM, the preference coefficient α is set to 0.1, while the mapping parameters (i.e., $\mathbf{M}$, $\mathbf{M}_k$, and [$\mathbf{A}$, $\mathbf{b}$]) are learned based on the same parameter settings used in MFEA-AE, MFEA-KAE, and MFEA-AT. The numerical results of independently running each algorithm 30 times on each test problem are collected for comparison. The Wilcoxon rank sum test with a 0.05 significance level is used to show the statistically significant differences.

5.4.2 Comparison with Recent EMT Algorithms

To demonstrate the competitive performance of MFEA-FLM relative to five state-of-the-art EMT algorithms, including MFEA-II, MFEA-AKT, MFEA-DAE, MTKDE, and LCB-EMT. The detailed numerical results of all compared algorithms on two test suites are given in Tab. 5.4 and Tab. 5.5, respectively.

Table 5.4: Mean objective values and standard deviations obtained by MFEA-FLM and compared algorithms on nine MTOPs.

Problem	MFEA-II	MFEA-AKT	MFEA-DAE	MTKDE	LCB-EMT	MFEA-FLM
CIHS-T1	3.68e-01(6.19e-02)-	1.04e+00(2.48e-02)-	1.12e-02(2.84e-02)-	9.34e-01(9.54e-02)-	3.04e-01(4.55e-02)-	**9.79e-03(2.35e-02)**
CIHS-T2	2.09e+02(2.93e+01)-	3.55e+02(3.42e+01)-	1.29e+01(2.19e+01)-	4.11e+02(2.43e+01)-	2.54e+02(9.41e+01)-	**1.04e+01(2.32e+01)**
CIMS-T1	2.41e+00(2.24e-01)-	5.71e+00(4.75e-01)-	6.88e-02(1.35e-02)-	4.25e+00(4.85e-01)-	2.09e+00(4.88e-01)-	**3.59e-02(1.73e-02)**
CIMS-T2	2.30e+02(4.10e+01)-	3.91e+02(3.25e+01)-	2.13e+00(6.59e-01)-	4.10e+02(2.91e+01)-	2.72e+02(1.18e+02)-	**7.16e-01(7.04e-01)**
CILS-T1	2.03e+01(2.61e+00)-	2.06e+01(4.81e-02)-	5.73e-01(1.24e-01)-	2.12e+01(4.66e-02)-	3.47e+00(5.62e+00)-	**4.08e-01(1.36e-01)**
CILS-T2	8.84e+02(2.13e+02)-	8.48e+03(5.31e+02)-	3.55e+00(1.01e+00)-	1.38e+04(4.17e+02)-	4.35e+02(3.62e+02)-	**2.25e+00(1.15e+00)**
PIHS-T1	4.10e+02(1.69e+01)-	6.72e+02(1.09e+02)-	2.98e+02(6.15e+01)+	5.57e+02(5.60e+01)-	**1.99e+02(7.80e+01)**+	3.75e+02(2.67e+01)
PIHS-T2	3.08e+00(6.39e-01)-	6.31e+02(1.64e+02)-	2.24e+00(7.31e-01)-	3.90e+02(1.82e+02)-	3.95e+00(1.34e+00)-	**1.36e+00(5.41e-01)**
PIMS-T1	2.42e+00(2.36e-01)-	4.66e+00(3.39e-01)-	**5.50e-01(1.63e-01)**~	3.92e+00(7.50e-01)-	2.53e+00(1.33e-01)-	6.88e-01(3.34e-01)
PIMS-T2	6.62e+02(1.85e+02)-	6.50e+03(2.09e+03)-	6.39e+01(1.77e+01)-	5.56e+03(6.92e+03)-	5.53e+02(4.50e+02)-	**5.34e+01(1.44e+01)**
PILS-T1	2.49e+00(2.75e-01)-	7.37e+00(1.08e+00)-	**3.12e-02(7.10e-03)**~	1.77e+01(2.44e+00)-	2.59e+00(6.25e-01)-	3.43e-02(9.85e-03)
PILS-T2	2.55e+00(7.66e-01)-	7.33e+00(1.07e+00)-	1.17e-01(3.81e-02)~	1.62e+01(5.51e+00)-	3.12e+00(1.03e+00)-	**1.10e-01(4.70e-02)**
NIHS-T1	8.41e+02(1.97e+02)-	2.76e+04(1.02e+04)-	4.98e+01(8.33e-01)-	1.14e+04(7.58e+03)-	6.73e+02(6.15e+02)-	**4.87e+01(4.84e-01)**
NIHS-T2	2.57e+02(4.76e+01)-	4.09e+02(3.05e+01)-	2.93e+00(1.70e+00)-	4.38e+02(2.22e+01)-	3.53e+02(8.42e+01)-	**8.79e-01(8.73e-01)**
NIMS-T1	2.62e-01(5.86e-02)-	1.13e+00(3.59e-02)-	1.36e-01(3.19e-02)-	1.14e+00(5.85e-02)-	2.43e-01(6.59e-02)-	**9.33e-02(2.71e-02)**
NIMS-T2	1.07e+01(1.99e+00)-	2.20e+01(2.72e+00)-	2.65e+00(2.35e-01)~	1.89e+01(2.88e+00)-	1.57e+01(2.81e+00)-	**2.59e+00(3.32e-01)**
NILS-T1	4.10e+02(1.63e+01)-	9.48e+02(1.05e+02)-	**3.27e+02(5.23e+01)**+	3.37e+03(6.83e+02)-	4.15e+02(1.76e+01)-	3.82e+02(3.75e+01)
NILS-T2	8.42e+02(2.93e+02)+	8.41e+03(5.77e+02)-	2.23e+03(1.47e+03)+	1.36e+04(4.47e+02)-	7.06e+02(2.38e+02)+	5.31e+03(8.60e+02)
Best/All	0/18	0/18	3/18	0/18	1/18	13/18
+/-/~	1/17/0	0/18/0	3/11/4	0/18/0	2/16/0	\

Table 5.5: Mean objective values and standard deviations obtained by MFEA-FLM and compared algorithms on F1 to F8.

Problem	MFEA-II	MFEA-AKT	MFEA-DAE	MTKDE	LCB-EMT	MFEA-FLM
F1-T1	3.78e-01(6.89e-02)-	1.11e+00(2.16e-02)-	2.50e-01(5.90e-02)-	9.56e-01(6.83e-02)-	3.10e-01(4.80e-02)-	**1.55e-01(3.59e-02)**
F1-T2	**2.03e+02(3.34e+01)**~	4.68e+02(2.90e+01)-	2.41e+02(4.04e+01)-	4.03e+02(2.39e+01)-	4.25e+02(1.11e+01)-	2.08e+02(4.37e+01)
F2-T1	2.42e+00(2.44e-01)-	8.35e+00(5.89e-01)-	1.76e+00(2.38e-01)-	4.53e+00(6.28e-01)-	3.46e+00(5.91e-01)-	**1.13e+00(2.20e-01)**
F2-T2	2.20e+02(3.58e+01)-	6.08e+02(5.99e+01)-	2.29e+02(3.69e+01)-	4.26e+02(3.10e+01)-	4.17e+02(1.45e+01)-	**1.17e+02(2.57e+01)**
F3-T1	1.18e+03(8.45e+02)-	5.04e+05(2.97e+05)-	**2.87e+02(7.24e+01)**~	6.03e+05(4.85e+05)-	9.83e+02(4.12e+02)-	3.07e+02(7.27e+01)
F3-T2	5.15e+01(1.03e+01)-	9.63e+03(1.29e+03)-	4.34e+01(1.04e+01)-	7.35e+02(1.42e+02)-	5.70e+01(9.24e+00)-	**3.18e+01(4.52e+00)**
F4-T1	4.10e+02(2.33e+01)~	1.62e+03(2.81e+02)-	**3.85e+02(3.81e+01)**~	3.30e+03(6.89e+02)-	4.22e+02(1.18e+01)-	4.03e+02(1.82e+01)
F4-T2	**8.03e+02(2.90e+02)**+	8.51e+03(6.95e+02)-	1.55e+03(8.77e+02)~	1.35e+04(4.41e+02)-	8.27e+02(3.42e+02)+	1.59e+03(1.05e+03)
F5-T1	4.19e+02(2.06e+01)-	5.53e+02(8.63e+01)-	5.73e-01(2.83e-01)-	1.77e+03(3.56e+02)-	3.05e+01(2.92e+01)-	**4.22e-01(2.44e-01)**
F5-T2	3.06e-02(1.99e-02)-	1.62e+01(5.72e+00)-	1.52e-04(2.69e-04)~	1.38e+02(6.31e+01)-	8.52e-02(4.01e-02)-	**1.01e-04(2.00e-04)**
F6-T1	2.85e+00(2.00e-01)-	5.73e+00(9.33e-01)-	3.28e-02(1.06e-02)-	1.60e+01(1.57e+00)-	**6.17e-03(5.55e-03)**+	**2.58e-02(9.43e-03)**
F6-T2	1.05e-01(3.85e-02)-	2.58e+00(4.57e-01)-	1.71e-02(1.00e-02)-	8.77e+00(1.58e+00)-	7.73e-02(2.86e-02)-	**9.33e-03(5.79e-03)**
F7-T1	1.30e+03(7.97e+02)-	1.20e+05(1.34e+05)-	4.86e+01(2.29e-01)-	2.67e+07(1.33e+07)-	1.23e+03(9.77e+02)-	**4.84e+01(2.07e-01)**
F7-T2	1.26e+02(4.29e+01)-	1.55e+02(2.92e+01)-	2.19e-02(1.86e-02)~	2.89e+02(3.88e+01)-	1.22e+02(3.99e+01)-	**1.71e-02(1.09e-02)**
F8-T1	4.13e+02(1.95e+01)-	1.22e+03(2.07e+02)-	**3.59e+02(4.32e+01)**+	3.48e+03(7.32e+02)-	4.15e+02(9.20e+00)-	3.99e+02(1.52e+01)
F8-T2	1.61e+02(1.62e+02)~	3.48e+03(4.51e+02)-	3.42e+02(2.57e+02)-	5.07e+03(3.24e+02)-	**9.48e+01(1.05e+02)**~	2.01e+02(1.86e+02)
Best/All	2/16	0/16	3/16	0/16	2/16	9/16
+/-/~	1/12/3	0/16/0	1/10/5	0/16/0	2/13/1	\

"+" (or "-") indicates the variant is better (or worse) than MFEA-FLM, and "~" indicates they obtain the statistically similar performance. The best result on each task is highlighted in bold.

The comparison results demonstrate that MFEA-FLM achieves better performance than its competitors on most test problems. In MFEA-II, the extent of knowledge transfer is adjustable by dynamically modifying the value of rmp. Additionally, MFEA-AKT employs multiple crossovers with different search biases for knowledge transfer, while MFEA-DAE employs multiple domain adaptation methods to transfer solutions across tasks. Compared to the three EMT algorithms, significantly outperforms them on most test problems from the first test suite (i.e., 17, 18, and 11 out of 18 cases, respectively. Similarly, the performance superiority of MFEA-FLM is also shown on the second test suite. MFEA-FLM obtains better results than MFEA-II, MFEA-AKT, and MFEA-DAE on 12, 16, and 10 out of 16 cases, respectively. Besides, MTKDE adopts differential evolution-based framework, where the meta-knowledge is transferred across tasks. In LCB-EMT, a solution selection method based on the lower confidence bound is proposed for knowledge transfer. It is observed that MFEA-FLM archives better results than MTKDE on all test cases from the two test suites. In addition, MFEA-FLM outperforms LCB-EMT on most test problems (i.e., 16 out of 18 cases from the first test suite and 13 out of 16 cases from the second test suite, respectively). In summary, the above comparison results demonstrate the competitive performance of MFEA-FLM.

5.4.3 Effectiveness Validation of MFEA-FLM

1) Effectiveness Validation of FL-Based Transfer Parameter Adaption

To study the effectiveness of dynamically adapting the value of rmp via FL-based transfer parameter adaption introduced in subsection 5.3.1, some variants of MFEA-FLM are designed by presetting rmp to a fixed value from {0, 0.3, 0.6, 0.9}.

The overall performance comparisons among these variants are computed based on the test of Friedman [121] as shown in Fig. 5.4 while the detailed numerical results are provided in Tab. 5.6 and Tab. 5.7. It is observed that MFEA-FLM achieves the smallest ranks among these variants on the two test suites (i.e., 1.5 and 2.03, respectively), which demonstrate that MFEA-FLM performs the best among all variants in terms of the overall performance on the two test suites. The above comparison results show that employing the proposed FL-based parameter adaption mechanism to adapt the transfer extent significantly improves the multitasking performance of MFEA when compared to its variants with a fixed $\boldsymbol{rmp}$.

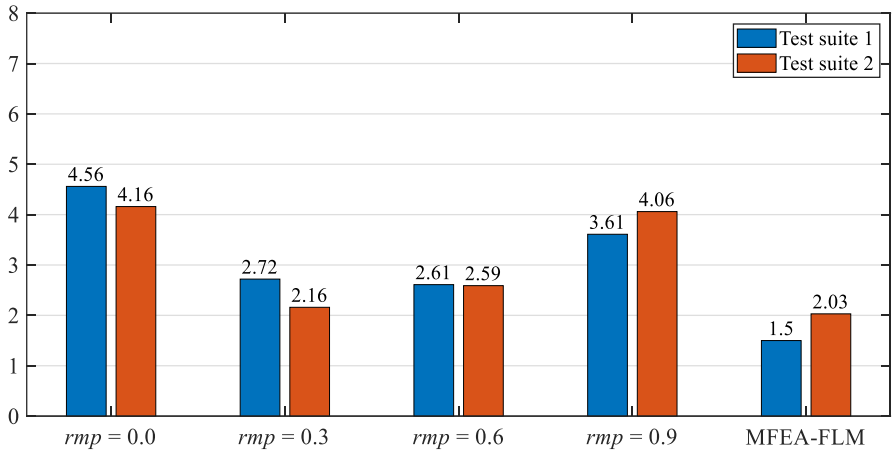


Figure 5.4: Performance ranks of MFEA-FLM and its variants with fixed values of $\boldsymbol{rmp}$ on two test suites.

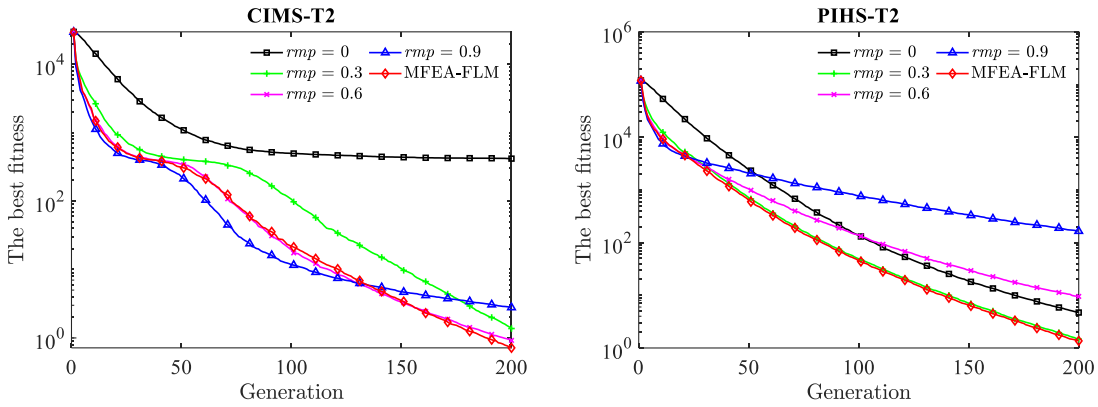
Table 5.6: Mean objective values and standard deviations obtained by MFEA-FLM and its variants with fixed values of rmp on nine MTOPs.

Problem	Variant-I (rmp= 0)	Variant-II (rmp= 0.3)	Variant-III (rmp= 0.6)	Variant-IV (rmp= 0.9)	MFEA-FLM
CIHS-T1	2.98e-01(5.24e-02)-	1.98e-02(3.27e-02)-	9.87e-03(2.80e-02)~	1.48e-02(4.58e-02)-	**9.79e-03(2.35e-02)**
CIHS-T2	4.15e+02(1.87e+01)-	2.30e+01(3.60e+01)-	1.27e+01(3.39e+01)~	1.59e+01(4.19e+01)-	**1.04e+01(2.32e+01)**
CIMS-T1	2.25e+00(2.42e-01)-	5.35e-02(1.16e-02)-	4.28e-02(1.04e-02)-	8.09e-02(2.21e-02)-	**3.59e-02(1.73e-02)**
CIMS-T2	4.17e+02(2.10e+01)-	1.37e+00(5.49e-01)-	9.20e-01(3.62e-01)-	2.78e+00(1.39e+00)-	**7.16e-01(7.04e-01)**
CILS-T1	2.12e+01(4.09e-02)-	4.92e-01(1.93e-01)~	7.38e-01(2.39e-01)-	1.86e+00(5.43e-01)-	**4.08e-01(1.36e-01)**
CILS-T2	1.13e+03(3.71e+02)-	2.99e+00(1.87e+00)~	5.44e+00(2.70e+00)-	2.90e+01(1.99e+01)-	**2.25e+00(1.15e+00)**
PIHS-T1	4.23e+02(2.12e+01)-	3.80e+02(1.90e+01)~	3.58e+02(3.66e+01)+	**3.28e+02(4.25e+01)**+	3.75e+02(2.67e+01)
PIHS-T2	4.71e+00(1.16e+00)-	1.47e+00(4.04e-01)~	9.53e+00(4.43e+00)-	1.65e+02(4.37e+01)-	**1.36e+00(5.41e-01)**
PIMS-T1	2.41e+00(2.21e-01)-	**4.22e-01(1.52e-01)**+	1.09e+00(4.49e-01)-	2.74e+00(2.39e-01)-	6.88e-01(3.34e-01)
PIMS-T2	1.25e+03(5.51e+02)-	5.98e+01(1.36e+01)-	**5.17e+01(2.08e+00)**+	5.47e+01(4.86e+00)-	5.34e+01(1.44e+01)
PILS-T1	2.47e+00(1.87e-01)-	**3.24e-02(9.42e-03)**~	4.44e-02(8.58e-03)-	8.03e-02(1.21e-02)-	3.43e-02(9.85e-03)
PILS-T2	5.97e+00(1.62e+00)-	**1.01e-01(3.35e-02)**~	1.47e-01(3.69e-02)-	3.05e-01(5.82e-02)-	1.10e-01(4.70e-02)
NIHS-T1	1.48e+03(1.27e+03)-	5.10e+01(8.90e+00)-	4.89e+01(5.44e-01)-	5.03e+01(7.94e-01)-	**4.87e+01(4.84e-01)**
NIHS-T2	4.17e+02(2.65e+01)-	4.90e+00(1.40e+01)-	1.28e+00(1.09e+00)-	3.67e+00(1.54e+00)-	**8.79e-01(8.73e-01)**
NIMS-T1	3.17e-01(6.07e-02)-	1.04e-01(2.49e-02)~	1.94e-01(5.38e-02)-	5.55e-01(9.57e-02)-	**9.33e-02(2.71e-02)**
NIMS-T2	1.56e+01(2.88e+00)-	2.83e+00(4.93e-01)-	3.26e+00(4.03e-01)-	5.17e+00(1.22e+00)-	**2.59e+00(3.32e-01)**
NILS-T1	4.15e+02(1.74e+01)-	4.03e+02(2.31e+01)-	3.93e+02(5.09e+01)~	**3.35e+02(6.42e+01)**+	3.82e+02(3.75e+01)
NILS-T2	**9.42e+02(3.97e+02)**+	4.28e+03(1.35e+03)+	5.66e+03(5.81e+02)-	6.09e+03(7.95e+01)-	5.31e+03(8.60e+02)
Best/All	1/18	3/18	1/18	2/18	11/18
+/-/~	1/17/0	2/9/7	2/13/3	2/16/0	\

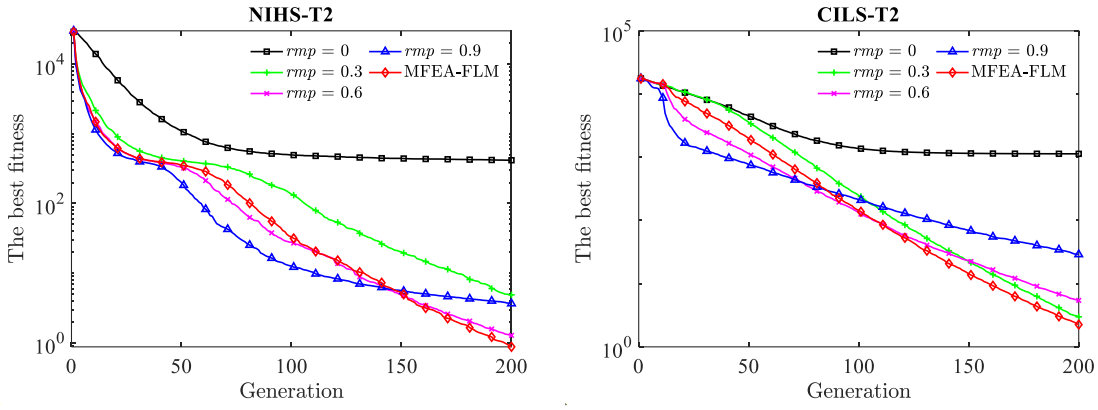
Table 5.7: Mean objective values and standard deviations obtained by MFEA-FLM and its variants with fixed values of rmp on F1 to F8.

Problem	Variant-I (rmp= 0)	Variant-II (rmp= 0.3)	Variant-III (rmp= 0.6)	Variant-IV (rmp= 0.9)	MFEA-FLM
F1-T1	2.59e-01(6.94e-02)-	1.56e-01(3.05e-02)~	1.99e-01(4.94e-02)-	2.84e-01(7.28e-02)-	**1.55e-01(3.59e-02)**
F1-T2	4.12e+02(1.86e+01)-	2.24e+02(6.63e+01)~	2.08e+02(4.87e+01)~	2.21e+02(5.37e+01)~	**2.08e+02(4.37e+01)**
F2-T1	2.56e+00(4.47e-01)-	**9.07e-01(3.10e-01)**+	1.27e+00(3.29e-01)~	1.96e+00(2.30e-01)-	1.13e+00(2.20e-01)
F2-T2	4.11e+02(1.65e+01)-	**1.05e+02(5.47e+01)**~	1.39e+02(5.04e+01)~	2.32e+02(4.71e+01)-	1.17e+02(2.57e+01)
F3-T1	1.48e+03(1.40e+03)-	3.35e+02(9.59e+01)~	**2.72e+02(7.41e+01)**~	3.77e+02(9.28e+01)-	3.07e+02(7.27e+01)
F3-T2	5.02e+01(7.95e+00)-	3.46e+01(5.37e+00)-	4.22e+01(6.39e+00)-	5.52e+01(7.60e+00)-	**3.18e+01(4.52e+00)**
F4-T1	4.13e+02(2.08e+01)~	4.06e+02(2.11e+01)~	4.19e+02(1.65e+01)-	4.46e+02(2.06e+01)-	**4.03e+02(1.82e+01)**
F4-T2	**8.02e+02(3.81e+02)**+	1.14e+03(5.60e+02)~	2.41e+03(1.38e+03)-	4.42e+03(1.35e+03)-	1.59e+03(1.05e+03)
F5-T1	4.17e+02(1.67e+01)-	**3.47e-01(1.88e-01)**~	3.88e-01(2.23e-01)~	4.96e-01(2.21e-01)~	4.22e-01(2.44e-01)
F5-T2	6.14e-02(3.54e-02)-	9.75e-05(1.91e-04)~	**5.46e-05(8.20e-05)**~	1.39e-04(2.71e-04)~	1.01e-04(2.00e-04)
F6-T1	2.16e+00(3.65e-01)-	**2.55e-02(6.55e-03)**~	2.93e-02(7.74e-03)~	3.49e-02(9.59e-03)-	2.58e-02(9.43e-03)
F6-T2	7.07e-02(3.08e-02)-	1.18e-02(9.10e-03)~	**8.25e-03(5.38e-03)**~	9.56e-03(6.31e-03)~	9.33e-03(5.79e-03)
F7-T1	9.58e+02(3.26e+02)-	4.85e+01(1.71e-01)~	4.85e+01(2.66e-01)~	4.86e+01(1.40e-01)-	**4.84e+01(2.07e-01)**
F7-T2	1.54e+02(4.89e+01)-	2.04e-02(1.29e-02)~	1.56e-02(1.11e-02)~	**1.31e-02(1.21e-02)**~	1.71e-02(1.09e-02)
F8-T1	4.09e+02(1.52e+01)-	**3.98e+02(1.72e+01)**~	4.09e+02(1.51e+01)-	4.20e+02(2.44e+01)-	3.99e+02(1.52e+01)
F8-T2	**1.34e+02(1.19e+02)**~	1.84e+02(1.59e+02)~	3.36e+02(2.55e+02)-	1.81e+03(7.74e+02)-	2.01e+02(1.86e+02)
Best/All	2/16	5/16	3/16	1/16	5/16
+/-/~	1/13/2	1/1/14	0/6/10	0/11/5	\

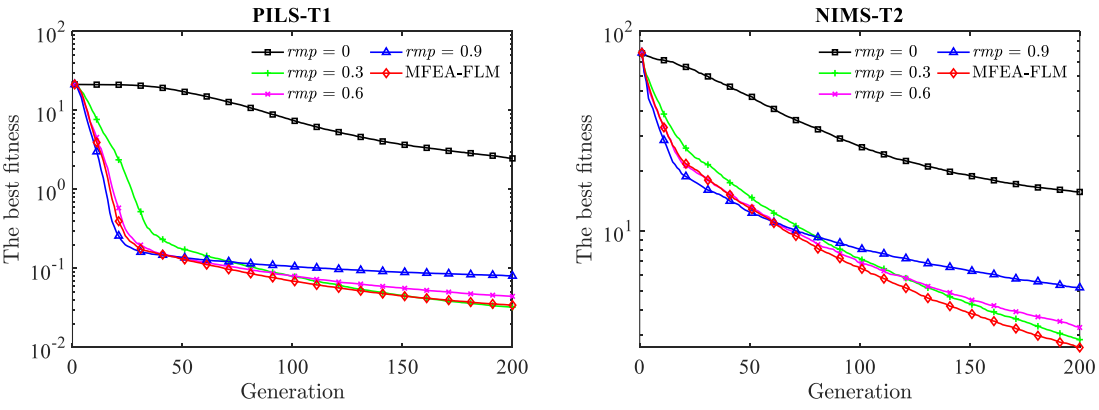
"+" (or "-") indicates the variant is better (or worse) than MFEA-FLM, and "~" indicates they obtain the statistically similar performance. The best result on each task is highlighted in bold.



(a) CIMS-T2 (b) PIHS-T2



(c) NIHS-T2 (d) CILS-T2



(e) PILS-T1 (f) NIMS-T2

Figure 5.5: Convergence curves of MFEA-FLM and its variants with the fixed values of rmp during the multitasking search process.

Moreover, the convergence curves of MFEA-FLM and these variants with different values of rmp on some representative problems are presented in Fig. 5.5. For CIMS, the two tasks have median similarity, and their global optima are identical in the unified search space with respect to all variables. As observed from Fig. 5.5(a), the variant with rmp = 0.9 achieves better convergence performance than other variants at the early evolutionary stage, while it is gradually outperformed by two variants with rmp = 0.3 and rmp = 0.6. This shows that setting rmp to 0.9 will lead to excessive knowledge transfer during the later evolutionary stage, which causes performance degradation, called negative transfer. By adaptively adjusting the value of rmp, MFEA-FLM finally achieves better performance, as shown in Fig. 5.5(a). Similarly, in terms of PIHS and NIMS, their two tasks have high and median similarity, respectively. As observed in Fig. 5.5(b) and Fig. 5.5(c), the variant with rmp = 0.3 and the variant with rmp = 0.6 achieve better performance on PIHS-T2 and NIMS-T2 when compared these variants with other fixed values of rmp, respectively. MFEA-FLM finally achieves the best performance when compared to all variants with fixed values of rmp, which demonstrates that employing the proposed FLM to adapt rmp effectively alleviates the threat of negative transfer. Additionally, as shown in Figs. 5.5(d), (e), and (f), MFEA-FLM finally achieves the best convergence performance when compared to its variants with fixed values of rmp on CILS-T2, PILS-T1, and NIMS-T2. The above observations demonstrate that employing the proposed method to dynamically adapt the extent of knowledge transfer can effectively alleviate the threat of negative transfer during the multitasking search process.

2) Effectiveness Validation of FL-Based Transfer Method Selection

To study the effectiveness of adaptively selecting the promising transfer method via

FL-based transfer method selection introduced in subsection 5.3.2, some variants of MFEA-FLM are designed as the compared algorithms by only employing one certain method (i.e., AE, KAE, AT, or Baseline) for conducting knowledge transfer. Note that Baseline is a simple method, which directly transfers the solutions from one task to another task without any adaptation. Additionally, another variant is designed as the compared algorithm by randomly selecting one from the $\boldsymbol{pool}$, aiming to validate the effectiveness and reasonability of the proposed FL-based selection mechanism. Note that the value of $\boldsymbol{rmp}$ is dynamically adapted in all compared algorithms. The overall performance comparisons among these variants are done based on the test of Friedman as shown in Fig. 5.6, while the detailed numerical results are provided in Tab. 5.8 and Tab. 5.9.

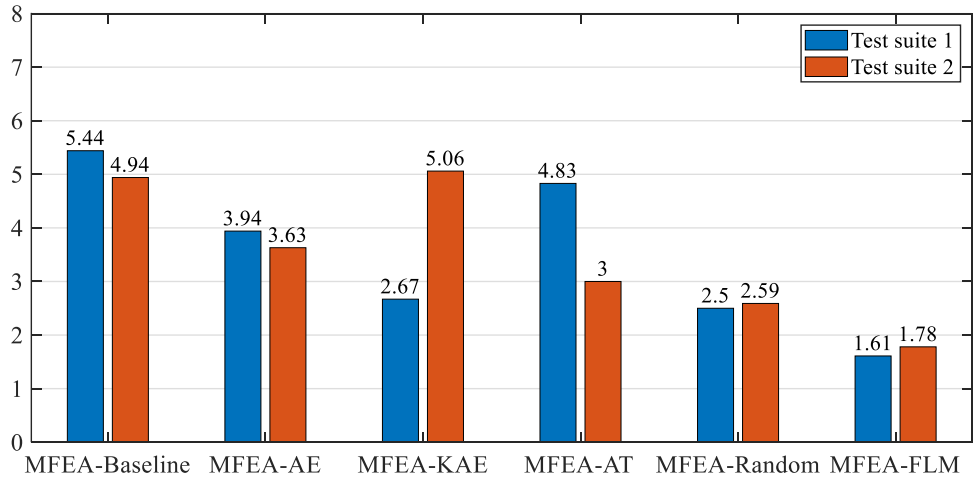


Figure 5.6: Performance ranks of MFEA-FLM and its variants with fixed transfer methods on two test suites.

Table 5.8: Mean objective values and standard deviations obtained by MFEA-FLM and its variants with fixed transfer methods on Nine MTOPs.

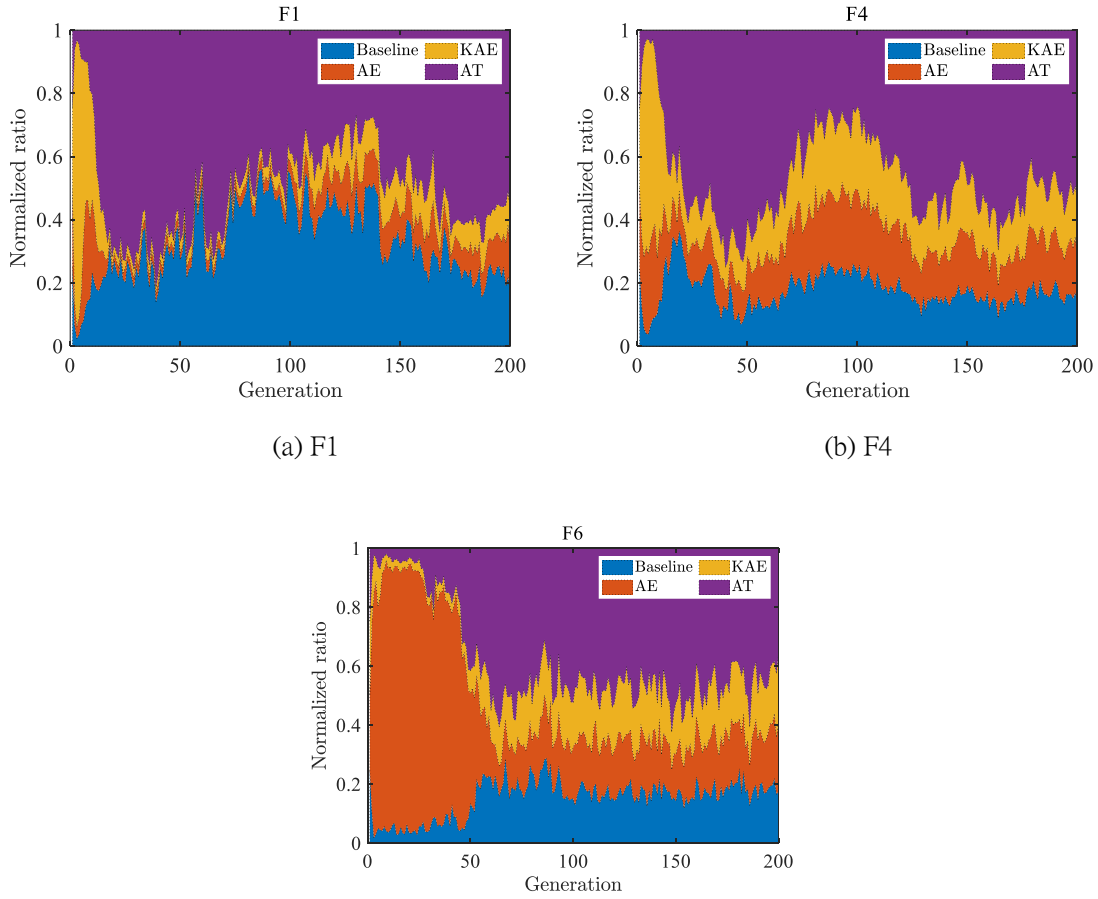
Problem	MFEA-Baseline	MFEA-AE	MFEA-KAE	MFEA-AT	MFEA-Random	MFEA-FLM
CIHS-T1	4.06e-01(7.80e-02)-	**2.29e-03(8.28e-04)**+	2.64e-01(6.13e-02)-	2.07e-01(4.73e-02)-	7.97e-03(3.33e-03)+	9.79e-03(2.35e-02)
CIHS-T2	2.20e+02(3.39e+01)-	**2.19e+00(9.11e-01)**~	3.94e+02(1.97e+01)-	2.05e+02(6.47e+01)-	1.18e+01(4.40e+00)-	1.04e+01(2.32e+01)
CIMS-T1	2.39e+00(2.19e-01)-	5.42e-02(9.59e-03)-	1.73e+00(2.68e-01)-	1.47e+00(3.45e-01)-	1.07e-01(1.97e-02)-	**3.59e-02(1.73e-02)**
CIMS-T2	2.44e+02(4.78e+01)-	1.50e+00(4.38e-01)-	3.98e+02(1.35e+01)-	1.47e+02(5.03e+01)-	4.38e+00(1.44e+00)-	**7.16e-01(7.04e-01)**
CILS-T1	1.97e+01(3.08e+00)-	2.12e+01(3.64e-02)-	3.28e+00(2.47e+00)-	1.98e+01(3.57e+00)-	5.63e-01(1.44e-01)-	**4.08e-01(1.36e-01)**
CILS-T2	1.44e+03(3.52e+02)-	1.24e+03(4.38e+02)-	1.67e+03(9.30e+02)-	1.17e+03(3.47e+02)-	3.36e+00(1.27e+00)-	**2.25e+00(1.15e+00)**
PIHS-T1	4.37e+02(1.99e+01)-	**3.73e+02(1.40e+01)**~	4.02e+02(1.48e+01)-	3.98e+02(1.41e+01)-	3.78e+02(2.19e+01)~	3.75e+02(2.67e+01)
PIHS-T2	1.15e+01(2.73e+00)-	9.75e+00(2.31e+00)-	7.76e+00(1.78e+00)-	2.41e+00(5.80e-01)-	1.90e+00(6.17e-01)-	**1.36e+00(5.41e-01)**
PIMS-T1	2.69e+00(2.45e-01)-	1.10e+00(1.94e-01)-	1.82e+00(2.84e-01)-	1.14e+00(2.01e-01)-	**5.94e-01(1.22e-01)**~	6.88e-01(3.34e-01)
PIMS-T2	8.08e+02(2.59e+02)-	**5.15e+01(1.22e+00)**+	4.04e+02(1.05e+02)-	2.78e+02(4.79e+01)-	6.98e+01(1.98e+01)-	5.34e+01(1.44e+01)
PILS-T1	3.04e+00(2.52e-01)-	4.10e-02(1.26e-02)-	2.42e+00(2.29e-01)-	1.63e+00(3.93e-01)-	3.71e-02(7.84e-03)~	**3.43e-02(9.85e-03)**
PILS-T2	2.86e+00(5.43e-01)-	1.56e-01(3.72e-02)-	2.45e+00(9.63e-01)-	1.27e+00(5.86e-01)-	1.26e-01(4.36e-02)~	**1.10e-01(4.70e-02)**
NIHS-T1	1.03e+03(2.66e+02)-	4.92e+01(2.36e-01)-	4.44e+02(9.21e+01)-	4.45e+02(9.53e+01)-	5.63e+01(1.38e+01)-	**4.87e+01(4.84e-01)**
NIHS-T2	3.06e+02(3.51e+01)-	1.74e+00(5.80e-01)-	3.96e+02(1.45e+01)-	2.96e+02(6.06e+01)-	1.37e+01(1.85e+01)-	**8.79e-01(8.73e-01)**
NIMS-T1	4.61e-01(6.23e-02)-	3.97e-01(6.52e-02)-	2.91e-01(5.64e-02)-	1.66e-01(4.26e-02)-	1.46e-01(4.11e-02)-	**9.33e-02(2.71e-02)**
NIMS-T2	1.41e+01(2.57e+00)-	2.71e+00(2.76e-01)~	5.69e+00(7.76e-01)-	8.37e+00(1.61e+00)-	2.92e+00(2.71e-01)-	**2.59e+00(3.32e-01)**
NILS-T1	4.27e+02(1.71e+01)-	3.96e+02(1.64e+01)~	4.04e+02(1.83e+01)-	4.34e+02(1.79e+01)-	3.92e+02(1.99e+01)~	**3.82e+02(3.75e+01)**
NILS-T2	1.36e+03(3.66e+02)+	1.27e+03(4.21e+02)+	5.27e+03(4.79e+02)~	**1.12e+03(3.52e+02)**+	1.81e+03(1.08e+03)+	5.31e+03(8.60e+02)
Best/All	0/18	4/18	0/18	1/18	1/18	12/18
+/-/~	1/17/0	3/11/4	0/17/1	1/17/0	2/11/5	\

Table 5.9: Mean objective values and standard deviations obtained by MFEA-FLM and its variants with fixed transfer methods on F1 to F8.

Problem	MFEA-Baseline	MFEA-AE	MFEA-KAE	MFEA-AT	MFEA-Random	MFEA-FLM
F1-T1	2.88e-01(4.52e-02)-	2.57e-01(4.74e-02)-	2.49e-01(4.52e-02)-	**1.54e-01(2.69e-02)**~	1.60e-01(4.46e-02)~	1.55e-01(3.59e-02)
F1-T2	2.11e+02(4.44e+01)~	4.15e+02(1.69e+01)-	4.19e+02(1.67e+01)-	**1.79e+02(6.11e+01)**~	2.59e+02(6.05e+01)-	2.08e+02(4.37e+01)
F2-T1	2.00e+00(2.77e-01)-	1.83e+00(2.44e-01)-	1.86e+00(3.14e-01)-	1.16e+00(3.01e-01)~	**1.10e+00(2.17e-01)**~	1.13e+00(2.20e-01)
F2-T2	2.42e+02(4.42e+01)-	4.12e+02(2.25e+01)-	4.20e+02(1.84e+01)-	1.20e+02(3.54e+01)~	1.26e+02(3.53e+01)~	**1.17e+02(2.57e+01)**
F3-T1	1.63e+03(1.21e+03)-	4.46e+02(1.44e+02)-	4.48e+02(1.26e+02)-	5.33e+02(5.22e+02)-	3.56e+02(5.75e+01)-	**3.07e+02(7.27e+01)**
F3-T2	5.49e+01(9.32e+00)-	6.33e+01(1.35e+01)-	2.70e+01(5.18e+00)+	4.89e+01(8.42e+00)-	3.61e+01(6.39e+00)-	**3.18e+01(4.52e+00)**
F4-T1	4.14e+02(1.82e+01)~	4.19e+02(1.87e+01)-	4.20e+02(1.65e+01)-	4.04e+02(1.60e+01)~	4.08e+02(1.48e+01)~	**4.03e+02(1.82e+01)**
F4-T2	8.77e+02(3.06e+02)+	9.71e+02(2.72e+02)+	3.43e+03(1.28e+03)-	**8.08e+02(2.58e+02)**+	9.42e+02(3.43e+02)+	1.59e+03(1.05e+03)
F5-T1	4.10e+02(1.95e+01)-	6.22e-01(2.92e-01)-	4.06e+02(2.34e+01)-	3.93e+02(1.68e+01)-	5.88e-01(2.83e-01)-	**4.22e-01(2.44e-01)**
F5-T2	4.52e-02(2.30e-02)-	4.95e-04(4.39e-04)-	5.77e-02(2.85e-02)-	2.53e-03(5.48e-03)-	2.17e-04(2.11e-04)-	**1.01e-04(2.00e-04)**
F6-T1	2.05e+00(2.83e-01)-	3.13e-02(8.16e-03)-	1.72e+00(2.76e-01)-	1.01e+00(2.21e-01)-	3.21e-02(8.93e-03)-	**2.58e-02(9.43e-03)**
F6-T2	7.59e-02(2.72e-02)-	2.50e-02(1.59e-02)-	7.89e-02(3.73e-02)-	3.04e-02(1.98e-02)-	1.93e-02(9.53e-03)-	**9.33e-03(5.79e-03)**
F7-T1	1.42e+03(8.45e+02)-	**4.84e+01(1.82e-01)**~	6.64e+02(5.72e+02)-	8.00e+02(6.13e+02)-	4.86e+01(3.06e-01)-	4.84e+01(2.07e-01)
F7-T2	9.58e+01(1.90e+01)-	2.97e-02(2.19e-02)-	1.33e+02(2.95e+01)-	7.85e+01(2.97e+01)-	4.42e-02(5.47e-02)-	**1.71e-02(1.09e-02)**
F8-T1	4.19e+02(1.81e+01)-	4.12e+02(2.11e+01)-	4.13e+02(2.02e+01)-	4.10e+02(1.30e+01)-	4.02e+02(1.86e+01)~	**3.99e+02(1.52e+01)**
F8-T2	1.84e+02(1.45e+02)~	1.78e+02(1.11e+02)~	3.44e+02(2.23e+02)-	**1.50e+02(1.35e+02)**~	1.78e+02(1.55e+02)~	2.01e+02(1.86e+02)
Best/All	0/16	1/16	0/16	4/16	1/16	10/16
+/-/~	1/12/3	1/13/2	1/15/0	1/9/6	1/9/6	\

"+" (or "-") indicates the variant is better (or worse) than MFEA-FLM, and "~" indicates they obtain the statistically similar performance. The best result on each task is highlighted in bold.

It is observed that MFEA-FLM achieves the smallest ranks among these variants on the two test suites (i.e., 1.61 and 1.78, respectively), which demonstrate that MFEA-FLM performs best among these variants in terms of the overall performance. The above comparison results show that employing the proposed method to adaptively select the promising method for knowledge transfer can further enhance the multitasking performance.



(a) F1

(b) F4

(c) F6

Figure 5.7: Normalized utilization ratios of four methods of knowledge transfer during the multitasking search process on F1, F4, and F6.

Moreover, to study the use of AE, KAE, AT, and Baseline during the multitasking evolutionary search process, Fig. 5.7 shows the normalized utilization ratio of each method during the multitasking search process on F1, F4, and F6. Here, the normalized utilization ratio is defined as the ratio of the number of using each method for knowledge transfer and the total number of triggering knowledge transfer. It can be observed that MFEA-FLM can dynamically adjust the utilization ratios of different methods for knowledge transfer during the multitasking search process. As shown in Fig. 5.7(a), the utilization ratios of AT and Baseline are obviously higher than that of AE and KAE on F1, which demonstrate that both AT and Baseline play a vital role in performing effective knowledge transfer on F1. However, in terms of F4, the two tasks have low similarity, and their global optima are in different locations. In such a case, compared to Baseline, both KAE and AE can enhance the transferability of solutions to some extent by learning the mapping between tasks. Thus, the utilization ratios of KAE and AE on F4 are relatively higher than that on F1, as shown in Fig. 5.7(b). In addition, it is observed from Fig. 5.7(c) that the utilization ratio of AE is significantly higher than that of other methods at the early evolutionary stage. This demonstrates that AE plays a vital role in significantly enhancing the multitasking performance on F6. In fact, the global optima of two tasks of F6 are located at the same point in the unified search space. In such cases, employing AE for knowledge transfer at the early evolutionary stage can achieve fast convergence. The above observations further validate that the proposed FL-based transfer method selection can effectively select the most promising method for conducting knowledge transfer in different scenarios.

5.4.4 Results of Incorporating FLM into EMT Algorithms

The proposed FL-based method is incorporated into three EMT algorithms (such as MFEA, MFEA-II, and MFEA-AKT), which form three new enhanced algorithms (called MFEA-FLM, MFEA-II-FLM, and MFEA-AKT-FLM, respectively). Note that MFEA-II-FLM adaptively selects the method of knowledge transfer by FL-based transfer method selection while the transfer parameter estimation scheme proposed in MFEA-II [24] is used to estimate the value of rmp. The details numerical results of the three EMT algorithms and their corresponding enhanced algorithms on the two test suites are represented in Tab. 5.10 and Tab. 5.11.

It is observed that MFEA-FLM, MFEA-II-FLM, and MFEA-AKT-FLM significantly outperform their original EMT algorithms in solving most test problems, respectively. Specifically, in terms of the first test suite, MFEA-FLM obtains better results than MFEA on 17 out of 18 cases, while it is beat by MFEA on 1 case. Additionally, MFEA-II-FLM and MFEA-AKT-FLM outperform MFEA-II and MFEA-AKT on all cases out of 18 cases, respectively. Similarly, for the second test suite, MFEA-FLM and MFEA-AKT-FLM also achieve significantly better results than MFEA and MFEA-AKT on most test problems (i.e., 13 and 16 out of 16 cases, respectively). Besides, MFEA-II-FLM outperforms MFEA-II on 9 out of 16 cases while they achieve similar results on the rest of the test problems. The above comparison results show that incorporating the proposed method into these EMT algorithms can significantly enhance the multitasking performance on most test problems when compared to their original EMT algorithms, respectively.

Table 5.10: Mean objective values and standard deviations obtained by three EMT algorithms and their enhanced algorithms on nine MTOPs.

Problem	MFEA	MFEA-FLM	MFEA-II	MFEA-II-FLM	MFEA-AKT	MFEA-AKT-FLM
CIHS-T1	2.77e-01(4.43e-02)-	**7.57e-03(1.13e-02)**	3.68e-01(6.19e-02)-	**3.39e-02(5.20e-02)**	1.04e+00(2.48e-02)-	**3.02e-01(9.56e-02)**
CIHS-T2	2.61e+02(3.67e+01)-	**1.09e+01(1.57e+01)**	2.09e+02(2.93e+01)-	**4.77e+01(7.58e+01)**	3.55e+02(3.42e+01)-	**1.49e+02(2.24e+01)**
CIMS-T1	1.87e+00(2.76e-01)-	**4.15e-02(1.03e-02)**	2.41e+00(2.24e-01)-	**6.60e-02(2.29e-02)**	5.71e+00(4.75e-01)-	**1.61e+00(8.95e-01)**
CIMS-T2	2.45e+02(3.63e+01)-	**8.76e-01(3.86e-01)**	2.30e+02(4.10e+01)-	**2.10e+00(1.43e+00)**	3.91e+02(3.25e+01)-	**1.22e+02(6.38e+01)**
CILS-T1	1.65e+01(6.56e+00)-	**4.17e-01(1.60e-01)**	2.03e+01(2.61e+00)-	**5.07e+00(5.43e+00)**	2.06e+01(4.81e-02)-	**5.38e+00(6.00e+00)**
CILS-T2	9.32e+02(4.25e+02)-	**2.41e+00(1.42e+00)**	8.84e+02(2.13e+02)-	**4.74e+02(4.68e+02)**	8.48e+03(5.31e+02)-	**1.31e+03(3.09e+03)**
PIHS-T1	4.22e+02(2.08e+01)-	**3.82e+02(2.51e+01)**	4.10e+02(1.69e+01)~	4.11e+02(1.69e+01)	6.72e+02(1.09e+02)-	**2.31e+02(3.63e+01)**
PIHS-T2	4.45e+00(1.18e+00)-	**7.96e-01(2.95e-01)**	3.08e+00(6.39e-01)~	3.31e+00(9.97e-01)	6.31e+02(1.64e+02)-	**1.01e+02(2.97e+01)**
PIMS-T1	2.18e+00(2.49e-01)-	**4.32e-01(1.99e-01)**	2.42e+00(2.36e-01)-	**1.01e+00(3.30e-01)**	4.66e+00(3.39e-01)-	**3.12e+00(2.53e-01)**
PIMS-T2	5.06e+02(1.19e+02)-	**6.34e+01(2.05e+01)**	6.62e+02(1.85e+02)-	**1.19e+02(1.68e+02)**	6.50e+03(2.09e+03)-	**1.71e+02(8.39e+01)**
PILS-T1	2.47e+00(2.11e-01)-	**2.72e-02(7.22e-03)**	2.49e+00(2.75e-01)-	**5.63e-02(3.99e-02)**	7.37e+00(1.08e+00)-	**9.72e-02(1.71e-02)**
PILS-T2	2.14e+00(3.81e-01)-	**1.05e-01(4.17e-02)**	2.55e+00(7.66e-01)-	**1.82e-01(1.29e-01)**	7.33e+00(1.07e+00)-	**3.88e-01(5.12e-02)**
NIHS-T1	6.42e+02(1.96e+02)-	**5.77e+01(2.75e+01)**	8.41e+02(1.97e+02)-	**7.10e+01(1.05e+02)**	2.76e+04(1.02e+04)-	**2.42e+02(1.52e+02)**
NIHS-T2	3.25e+02(4.13e+01)-	**1.46e+01(3.92e+01)**	2.57e+02(4.76e+01)-	**2.05e+01(7.52e+01)**	4.09e+02(3.05e+01)-	**1.22e+02(6.29e+01)**
NIMS-T1	2.84e-01(4.71e-02)-	**8.80e-02(2.71e-02)**	2.62e-01(5.86e-02)-	**2.24e-01(4.55e-02)**	1.13e+00(3.59e-02)-	**5.11e-01(8.90e-02)**
NIMS-T2	1.00e+01(1.85e+00)-	**2.60e+00(3.19e-01)**	1.07e+01(1.99e+00)-	**7.50e+00(2.56e+00)**	2.20e+01(2.72e+00)-	**8.18e+00(1.79e+00)**
NILS-T1	4.19e+02(2.25e+01)-	**3.91e+02(2.26e+01)**	4.10e+02(1.63e+01)~	4.06e+02(1.58e+01)	9.48e+02(1.05e+02)-	**3.95e+02(1.19e+02)**
NILS-T2	**9.43e+02(3.44e+02)**+	3.13e+03(1.54e+03)	8.42e+02(2.93e+02)~	7.27e+02(2.71e+02)	8.41e+03(5.77e+02)-	**7.15e+03(1.17e+03)**
+/-/~	1/17/0	\	0/18/0	\	0/18/0	\

Table 5.11: Mean objective values and standard deviations obtained by three EMT algorithms and their enhanced algorithms on F1 to F8.

Problem	MFEA	MFEA-FLM	MFEA-II	MFEA-II-FLM	MFEA-AKT	MFEA-AKT-FLM
F1-T1	2.96e-01(5.97e-02)-	**1.72e-01(4.25e-02)**	3.78e-01(6.89e-02)-	**2.76e-01(6.58e-02)**	1.11e+00(2.16e-02)-	**1.03e+00(1.69e-02)**
F1-T2	2.59e+02(4.11e+01)-	**2.22e+02(5.34e+01)**	**2.03e+02(3.34e+01)**~	2.18e+02(4.95e+01)	4.68e+02(2.90e+01)-	**3.64e+02(2.04e+01)**
F2-T1	1.99e+00(2.43e-01)-	**1.10e+00(2.44e-01)**	2.42e+00(2.44e-01)-	**1.89e+00(3.98e-01)**	8.35e+00(5.89e-01)-	**6.33e+00(5.30e-01)**
F2-T2	2.52e+02(3.97e+01)-	**1.16e+02(3.57e+01)**	2.20e+02(3.58e+01)~	**2.11e+02(4.79e+01)**	6.08e+02(5.99e+01)-	**4.46e+02(4.41e+01)**
F3-T1	1.70e+03(6.97e+02)-	**3.24e+02(9.11e+01)**	1.18e+03(8.45e+02)-	**8.79e+02(3.06e+02)**	5.04e+05(2.97e+05)-	**8.60e+03(1.99e+04)**
F3-T2	6.33e+01(8.94e+00)-	**2.98e+01(5.29e+00)**	5.15e+01(1.03e+01)~	**5.10e+01(7.79e+00)**	9.63e+03(1.29e+03)-	**1.46e+02(2.37e+01)**
F4-T1	4.17e+02(2.09e+01)~	**4.12e+02(1.55e+01)**	**4.10e+02(2.33e+01)**~	4.17e+02(1.88e+01)	1.62e+03(2.81e+02)-	**9.75e+02(1.00e+02)**
F4-T2	**9.50e+02(3.72e+02)**+	1.84e+03(1.12e+03)	**8.03e+02(2.90e+02)**~	8.38e+02(2.88e+02)	8.51e+03(6.95e+02)-	**6.77e+03(1.84e+03)**
F5-T1	4.14e+02(1.71e+01)-	**3.78e-01(1.81e-01)**	4.19e+02(2.06e+01)-	**9.23e-01(4.22e-01)**	5.53e+02(8.63e+01)-	**3.23e+00(9.49e-01)**
F5-T2	3.78e-02(1.81e-02)-	**8.62e-05(1.58e-04)**	3.06e-02(1.99e-02)-	**2.24e-04(4.31e-04)**	1.62e+01(5.72e+00)-	**9.12e-03(4.17e-03)**
F6-T1	2.27e+00(2.77e-01)-	**2.51e-02(8.45e-03)**	2.85e+00(2.00e-01)-	**4.51e-02(1.22e-02)**	5.73e+00(9.33e-01)-	**8.96e-02(1.53e-02)**
F6-T2	7.17e-02(2.28e-02)-	**1.02e-02(7.76e-03)**	1.05e-01(3.85e-02)-	**2.23e-02(8.76e-03)**	2.58e+00(4.57e-01)-	**2.38e-01(4.19e-02)**
F7-T1	1.73e+03(1.06e+03)-	**4.84e+01(1.90e-01)**	1.30e+03(7.97e+02)-	**4.74e+02(7.79e+02)**	1.20e+05(1.34e+05)-	**6.91e+01(4.89e+01)**
F7-T2	8.98e+01(1.96e+01)-	**2.04e-02(1.76e-02)**	1.26e+02(4.29e+01)-	**7.10e+01(7.29e+01)**	1.55e+02(2.92e+01)-	**6.83e+00(1.79e+01)**
F8-T1	4.13e+02(2.40e+01)-	**3.97e+02(2.18e+01)**	4.13e+02(1.95e+01)~	**4.06e+02(2.28e+01)**	1.22e+03(2.07e+02)-	**5.56e+02(9.07e+01)**
F8-T2	**1.42e+02(1.22e+02)**~	2.72e+02(2.02e+02)	1.61e+02(1.62e+02)~	**1.61e+02(1.32e+02)**	3.48e+03(4.51e+02)-	**2.87e+03(2.01e+02)**
+/-/~	1/13/2	\	0/9/7	\	0/16/0	\

"+" (or "-") indicates the original EMT algorithm is better (or worse) than its enhanced algorithm, and "~" indicates they obtain the statistically similar performance.

Table 5.12: Summarized results of MFEA-FLM and its variants with different parameters.

Algorithm comparison	Test suite 1 (+/-/~)	Test suite 2 (+/-/~)
α = 0 vs α = 0.1	0/0/17	0/1/15
α = 0.2 vs α = 0.1	1/2/15	0/0/16
α = 0.4 vs α = 0.1	3/4/11	1/1/14
α = 0.6 vs α = 0.1	2/9/7	1/2/13
α = 0.8 vs α = 0.1	0/11/7	0/8/7
α = 1.0 vs α = 0.1	0/18/0	0/16/0
Variant-V vs MFEA-FLM	0/3/15	0/0/16
Variant-VI vs MFEA-FLM	0/1/17	1/0/15
Variant-VII vs MFEA-FLM	0/3/15	0/0/16
Variant-VIII vs MFEA-FLM	0/0/18	0/0/16

"+", "-", and "~" denote the numbers of better, worse, and similar results obtained by the corresponding competitor when compared to MFEA-FLM, respectively.

5.4.5 Parameter Sensitivity Analysis

1) The Effect of α

To study the impact of α, MFEA-FLM with α = 0.1 is compared to MFEA-FLM with different values of α from {0, 0.2, 0.4, 0.6, 0.8, 1.0}. The comparison results on the two test suites are summarized in Tab. 5.12. It is observed that MFEA-FLM with some smaller values of α (i.e., 0 and 0.2) achieves similar multitasking performance on most test problems. However, setting α to some larger values (i.e., 0.8 and 1.0) will lead to the obvious performance degradation of MFEA-FLM when solving most test problems. The observations demonstrate that the current improvements of multitasking performance should occupy a high proportion in quantifying the efficacy of knowledge transfer. Thus, setting α to 0.1 is suggested in this study.

2) The Effect of Membership Functions (MFs)

Two variants (called Variant-V and Variant-VI) are designed to study the effect of MFs with different parameters on the performance of FL-based parameter adaption mechanism. Specifically, in Variant-V, the parameters of the singleton MFs are set to some larger values, i.e., -0.6, -0.4, -0.2, 0, 0.2, 0.4, and 0.6, respectively. In Variant-VI, the parameters of the singleton MFs are set to some smaller values, i.e., -0.15, -0.10, -0.05, 0, 0.05, 0.10, and 0.15, respectively. Additionally, two variants (called Variant-VII and Variant-VIII) are designed to study the effect of MFs with different parameters on the performance of FL-based transfer method selection mechanism. Specifically, in Variant VII, three MFs with larger degree of intersection, i.e., class L with the parameter tuple (0.0, 0.0, 0.1, 0.8), class triangular with the parameter tuple (0.0, 0.5, 1.0), and class γ with the parameter tuple (0.2, 0.9, 1.0, 1.0), are used to define fuzzy sets of the consequence space. In Variant VII, three MFs with smaller degree of intersection are used, i.e., class L with (0.0, 0.0, 0.1, 0.3), class triangular with (0.3, 0.5, 0.7), and class γ with (0.7, 0.9, 1.0, 1.0). The summarized results on the two test suites are collected in Tab. 5.12. It is observed that these variants achieve similar results on most test problems when compared to MFEA-FLM. According to the comparison results, it is concluded that the multitasking performance of MFEA-FLM on most test problems is not sensitive to the MFs with different parameters.

5.4.6 Further Study in Handling More Complex Problems

1) Comparison Results on Complex MTOPs

Ten complex MTOPs from CEC 2021 Competition on Evolutionary Multi-task Optimization are used as the test problems (called C-MTOP1 to C-MTOP10). Each MTOP consists of two single-objective continuous optimization tasks, which bear cer-

tain commonality and complementarity in terms of the global optimum and the fitness landscape. To further examine the effectiveness of the proposed FL-based mechanisms in solving C-MTOP1 to C-MTOP10, FL-based transfer parameter adaption and transfer method selection are incorporated into MFEA and MFEA-AKT, forming two enhanced algorithms (i.e., MFEA-FLM and MFEA-AKT-FLM). The detailed numerical results of all compared algorithms are provided in Tab. 5.13.

Table 5.13: Mean objective values and standard deviations obtained by two EMT algorithms and their enhanced algorithms on ten complex MTOPs.

C-MTOPs	MFEA	MFEA-FLM	MFEA-AKT	MFEA-AKT-FLM
C-MTOP1-T1	6.2213e+02(1.6e-01)-	**6.2199e+02(1.3e-01)**	6.2387e+02(1.7e-01)~	**6.2382e+02(2.0e-01)**
C-MTOP1-T2	6.2501e+02(1.8e-01)~	**6.2496e+02(1.7e-01)**	6.2697e+02(1.8e-01)-	**6.2684e+02(1.8e-01)**
C-MTOP2-T1	7.1127e+02(5.7e-03)-	**7.1126e+02(7.1e-03)**	7.1195e+02(5.7e-02)-	**7.1154e+02(7.0e-02)**
C-MTOP2-T2	7.1760e+02(1.5e-02)-	**7.1758e+02(9.3e-03)**	7.1855e+02(8.1e-02)-	**7.1800e+02(9.3e-02)**
C-MTOP3-T1	2.8841e+06(1.9e+04)-	**2.8709e+06(1.3e+04)**	4.1901e+06(1.0e+05)-	**3.4404e+06(9.1e+04)**
C-MTOP3-T2	3.5071e+07(1.1e+05)-	**3.4976e+07(1.0e+05)**	4.5364e+07(6.6e+05)-	**3.9234e+07(1.0e+06)**
C-MTOP4-T1	1.3042e+03(2.3e-04)-	**1.3042e+03(2.0e-04)**	1.3043e+03(1.7e-03)-	**1.3043e+03(1.8e-03)**
C-MTOP4-T2	1.3047e+03(4.7e-04)-	**1.3047e+03(1.8e-04)**	1.3048e+03(1.3e-03)-	**1.3048e+03(1.7e-03)**
C-MTOP5-T1	3.3750e+05(2.8e+02)-	**3.3713e+05(2.4e+02)**	3.6149e+05(2.3e+03)-	**3.4626e+05(1.7e+03)**
C-MTOP5-T2	8.5128e+05(7.9e+02)-	**8.5056e+05(7.0e+02)**	9.0568e+05(5.7e+03)-	**8.7383e+05(4.0e+03)**
C-MTOP6-T1	1.8741e+08(2.1e+05)-	**1.8706e+08(1.7e+05)**	1.9987e+08(9.7e+05)-	**1.9294e+08(1.1e+06)**
C-MTOP6-T2	2.6582e+09(1.7e+06)-	**2.6562e+09(1.2e+06)**	2.7379e+09(1.0e+07)-	**2.6960e+09(6.8e+06)**
C-MTOP7-T1	6.2238e+04(7.6e+01)-	**6.2134e+04(5.0e+01)**	6.8513e+04(6.2e+02)-	**6.4988e+04(5.4e+02)**
C-MTOP7-T2	1.4821e+04(1.7e+01)-	**1.4808e+04(1.3e+01)**	1.5930e+04(1.1e+02)-	**1.5263e+04(1.2e+02)**
C-MTOP8-T1	5.2015e+02(1.1e-01)-	**5.2008e+02(6.6e-02)**	5.2058e+02(7.9e-02)-	**5.2047e+02(1.0e-01)**
C-MTOP8-T2	5.2016e+02(9.1e-02)~	**5.2013e+02(8.7e-02)**	5.2061e+02(7.8e-02)-	**5.2049e+02(8.1e-02)**
C-MTOP9-T1	1.8983e+04(4.5e+00)-	**1.8977e+04(3.6e+00)**	1.9296e+04(3.5e+01)-	**1.9138e+04(3.3e+01)**
C-MTOP9-T2	**1.6222e+03(5.1e-02)**~	1.6222e+03(6.8e-02)	1.6230e+03(1.4e-01)-	**1.6227e+03(1.0e-01)**
C-MTOP10-T1	1.9481e+09(1.3e+06)-	**1.9463e+09(6.9e+05)**	2.0401e+09(8.9e+06)-	**1.9873e+09(7.7e+06)**
C-MTOP10-T2	6.7436e+08(3.9e+05)-	**6.7367e+08(3.3e+05)**	7.1113e+08(4.2e+06)-	**6.8978e+08(2.5e+06)**
+/-/~	0/17/3	\	0/19/1	\

"+" (or "-") indicates the original multi-objective EMT algorithm is better (or worse) than the enhanced algorithm, and "~" indicates they obtain the statistically similar performance.

It is observed that MFEA-FLM and MFEA-AKT-FLM significantly outperform their original algorithms on most test problems, respectively. Additionally, they do not lead to any performance degradation on these test problems. The comparison results show that employing FL-based mechanisms to modify the value of rmp and select the most promising method for knowledge transfer can further enhance the multitasking performance of MFEA and MFEA-AKT in handling the ten complex MTOPs.

2) Comparison Results on Multiobjective MTOPs

To study the effectiveness of FL-based mechanisms on the multiobjective MTOPs (called MO-MTOPs), one common test suite including nine test problems [55] is used. Each MO-MTOP has two multiobjective optimization tasks. In the experiment, FL-based transfer parameter adaption and transfer method selection are incorporated into three multiobjective EMT algorithms (i.e., MO-MFEA, MO-MFEA-II, and MO-MFEA-DAE). Specifically, two FL-based mechanisms are incorporated into MO-MFEA, and the enhanced algorithm is called MO-MFEA-FLM. Additionally, MFEA-DAE-FLM is formed by incorporating FL-based transfer parameter adaption into MO-MFEA-DAE to dynamically modify the value of rmp. However, MO-MFEA-II-FLM is formed by incorporating FL-based transfer method selection into MO-MFEA-II to adaptively select the promising transfer method from multiple candidates (i.e., Baseline, AE, KAE, and AT). The detailed numerical results are provided in Tab. 5.14, showing that the three enhanced algorithms with the proposed FL-based mechanism(s) significantly outperform their original algorithms on most test problems, respectively. The comparison results show that incorporating FL-based mechanism(s) into the three EMT algorithms can significantly improve the multitasking performance in handling the multiobjective MTOPs.

Table 5.14: Mean objective values and standard deviations obtained by two EMT algorithms and their enhanced algorithms on nine multobjective MTOPs.

MO-MTOP	MO-MFEA	MO-MFEA-FLM	MO-MFEA-II	MO-MFEA-II-FLM	MO-MFEA-DAE	MO-MFEA-DAE-FLM
CIHS-T1	2.67e-02(6.69e-03)-	**1.79e-03(6.26e-04)**	8.88e-03(3.24e-03)-	**3.65e-03(2.08e-03)**	5.00e-02(1.55e-02)-	**9.66e-03(2.95e-03)**
CIHS-T2	2.73e-02(3.80e-03)-	**6.41e-03(1.22e-03)**	1.52e-02(2.89e-03)-	**9.19e-03(2.24e-03)**	3.85e-02(5.96e-03)-	**1.61e-02(2.37e-03)**
CIMS-T1	1.71e-01(7.30e-02)-	**1.17e-01(9.22e-02)**	1.93e-01(5.45e-02)~	1.68e-01(7.96e-02)	1.54e-01(7.90e-02)~	1.61e-01(8.12e-02)
CIMS-T2	1.48e-02(1.04e-02)-	**8.65e-03(6.41e-03)**	1.00e-02(6.59e-03)~	1.27e-02(9.57e-03)	1.37e-02(9.48e-03)~	1.39e-02(9.16e-03)
CILS-T1	2.16e-02(6.28e-03)-	**4.09e-03(1.11e-03)**	9.82e-02(9.50e-02)-	**7.39e-02(1.86e-01)**	4.64e-02(1.13e-02)-	**1.47e-02(4.76e-03)**
CILS-T2	4.33e-04(4.20e-05)-	**2.54e-04(1.36e-05)**	6.90e-04(2.01e-04)-	**4.01e-04(2.21e-04)**	6.21e-04(6.78e-05)-	**3.56e-04(4.05e-05)**
PIHS-T1	2.01e-01(5.29e-02)-	**3.33e-02(1.39e-02)**	1.54e-01(4.74e-02)-	**4.65e-02(1.39e-02)**	3.36e-01(1.00e-01)-	**1.37e-01(5.03e-02)**
PIHS-T2	2.33e+00(3.90e-01)-	**1.78e+00(2.80e-01)**	2.02e+00(2.88e-01)~	1.92e+00(3.24e-01)	3.87e+00(6.05e-01)-	**2.98e+00(5.44e-01)**
PIMS-T1	3.18e-02(1.33e-02)-	**5.31e-03(1.49e-03)**	6.31e-02(1.38e-02)~	5.12e-02(2.37e-02)	1.06e-02(2.87e-03)-	**8.70e-03(2.02e-03)**
PIMS-T2	1.74e+01(2.67e+00)-	**7.29e-01(9.06e-01)**	1.98e+01(3.76e+00)-	**8.73e+00(7.56e+00)**	1.98e+00(3.82e-01)-	**1.51e+00(8.44e-01)**
PILS-T1	5.42e-03(1.29e-03)-	**1.15e-03(3.56e-04)**	5.37e-03(1.33e-03)-	**4.42e-03(1.38e-03)**	7.94e-03(2.08e-03)-	**5.94e-03(2.02e-03)**
PILS-T2	1.79e-01(2.30e-02)-	**8.71e-02(1.09e-02)**	6.69e-01(3.92e-03)-	**3.19e-01(2.19e-01)**	1.67e-01(1.75e-02)-	**1.48e-01(1.55e-02)**
NIHS-T1	1.37e+01(3.54e+00)-	**2.26e+00(3.12e-01)**	**5.98e+00(1.69e+00)**+	8.75e+00(2.39e+01)	1.67e+01(4.27e+00)-	**4.23e+00(1.04e+00)**
NIHS-T2	8.95e-02(2.75e-02)-	**5.16e-03(1.89e-03)**	2.90e-02(1.30e-02)-	**2.61e-02(6.10e-02)**	1.16e-01(3.89e-02)-	**1.94e-02(8.46e-03)**
NIMS-T1	5.70e-01(2.82e-01)-	**2.93e-01(2.44e-01)**	7.06e-01(3.34e-01)~	6.11e-01(2.85e-01)	7.56e-01(3.90e-01)-	**4.56e-01(3.16e-01)**
NIMS-T2	1.41e-01(3.04e-01)-	**2.40e-02(3.41e-02)**	2.35e-01(1.36e-01)~	2.69e-01(2.17e-01)	1.41e-01(1.30e-01)-	**5.55e-02(6.34e-02)**
NILS-T1	1.05e-02(7.54e-05)-	**9.98e-03(6.61e-04)**	1.06e-02(6.02e-05)~	1.06e-02(9.69e-05)	1.06e-02(1.53e-04)~	1.06e-02(1.14e-04)
NILS-T2	6.52e-01(3.75e-03)-	**6.44e-01(1.20e-03)**	6.52e-01(3.46e-03)~	6.50e-01(2.72e-03)	6.44e-01(1.07e-03)~	6.44e-01(9.58e-04)
+/-/~	0/18/0	\	1/9/8	\	0/14/4	\

"+" (or "-") indicates the original multi-objective EMT algorithm is better (or worse) than the enhanced algorithm, and "~" indicates they obtain the statistically similar performance.

5.5 Conclusion

This chapter has proposed a new MFEA with FL-based AKT for performing more effective and robust EMT. The proposed method includes two main components, which are employed to dynamically adapt the extent of knowledge transfer and adaptively select the promising method for knowledge transfer, respectively. Specifically, in the first component, one fuzzy inference system is employed to estimate the change in the transfer parameter, and then the value of the transfer parameter is dynamically

modified to determine the transfer extent. Additionally, in the second component, another one fuzzy inference system is used to estimate the applicability of each transfer method for the current transfer scenario. After that, the most promising one is adaptively selected from multiple candidates for knowledge transfer. In comparison with existing deterministic methods, the proposed method has the mechanism for the fuzzy or inaccurate information processing, which effectively enhances the reliability in making decisions. The experimental results have validated the effectiveness of the proposed method.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis aims to study and design adaptive knowledge transfer methods to intelligently address one or more of three critical issues concerning knowledge transfer in ETO.

Firstly, to adaptively decide what to transfer in ESTO, this thesis has proposed a fuzzy classifier-assisted method to select the most useful source solution for knowledge transfer for accelerating the optimization of the target task, as presented in Chapter 3. By constructing the training data, the fuzzy classifier is built to estimate the solution usefulness of all available source tasks by returning a class label and its membership degree to that class. The label indicates whether the source task is useful for the target task, while its membership degree to that class further quantifies the de-

gree of its usefulness if it is useful. Thus, the proposed method can accurately select the most useful solution for knowledge transfer when compared to existing methods based on distance metrics or traditional machine learning models. The experimental results on a series of test problems have validated the effectiveness of the proposed method.

Secondly, to adaptively decide how to transfer in EMT, this thesis has proposed an ensemble method of domain adaptation to select the promising domain adaptation method for conducting knowledge transfer, as presented in Chapter 4. Considering multiple domain adaptation methods, the differences in the mapping construction mechanisms enable their mapping behaviours to have a unique bias in representing the connection from the source task to the target task. One specific bias can show the superiority in its preferred multitasking transfer scenario, while it has poor performance in other scenarios. By striking the balance between efficacy and diversity when determining which one domain adaptation method for use, the proposed method can take full advantages of the strengths of each domain adaptation method to further enhance knowledge transferability across tasks. The experimental results on a series of test problems have validated the effectiveness of the proposed method.

Finally, to adaptively decide when to transfer and how to transfer in EMT, this thesis has proposed a fuzzy logic-based adaptive knowledge transfer method, as presented in Chapter 5. The proposed method includes two fuzzy logic-based components, such as fuzzy logic-based transfer parameter adaption and fuzzy logic-based transfer method selection. The first component is designed to effectively adapt the transfer extent along the multitasking search process by dynamically adjusting the value of the transfer parameter, thereby alleviating the risk of negative transfer. Meanwhile, the second component is employed to adaptively select the optimal transfer method from multi-

ple candidates for conducting knowledge transfer, thereby enhancing knowledge transferability across tasks. The experimental results on a series of test problems have validated the effectiveness of the proposed method.

However, these proposed methods also introduce extra hyperparameters, which may significantly impact the optimization efficiency and performance. Various hyperparameters turning strategies have been proposed, including grid search, random search, Beyesian optimization, and genetic algorithms [141]. The choice of hyperparameter selection strategy often depends on the specific problem, the available computational resources, and the desired balance between exploration and exploitation. Using hyperparameters turning strategies can result in the proposed methods that generalized well to new problems, achieving better optimization efficiency and performance in practical applications.

6.2 Future Work

Exploring more intelligent and effective ETO methods to address the challenges of what, how, and when to transfer remains much room. Some potential research directions are listed as follows.

1) Advanced ETO Approaches for Large-Scale Optimization

In real-world applications, optimization problems containing a large number of decision variables are called large-scale optimization problems (LSOPs) [142]. As the number of decision variables increases, the search space will grow exponentially. Traditional EAs tend to suffer from slow convergence in solving LSOPs due to their

huge search spaces, which will result in a larger computational cost. A number of methods have been developed to assist EA solvers in solving LSOPs, such as decision variable grouping [143], decision space reduction [144], and novel search strategies [145]. As an emerging search paradigm incorporating transfer learning into EAs, ETO shows significant potential in further enhancing the search capability of EAs in handling LSOPs. Possible research directions include the following:

- Designing advanced ETO approaches to accelerate the optimization of LSOPs by conducting effective knowledge transfer among different groups of decision variables.
- Designing advanced ETO approaches to accelerate the optimization of LSOPs by conducting effective knowledge transfer in the reduced decision space.
- Designing advanced ETO approaches to accelerate the optimization of LSOPs by transferring knowledge from small-scale problem space to large-scale problem space.

2) Advanced ETO Approaches for Expensive Optimization

Expensive optimization problems (EOPs) refer to those problems in which the objective functions are expensive to evaluate [146]. The limited numbers of real function evaluations bring difficulties to traditional EAs when solving EOPs. Over the years, surrogate-based EAs have attracted increasing attention, aiming to replace real function evaluations by employing cheap surrogate models, including various classification models [147] and regression models [148]. However, several issues, including lacking enough training data and retraining surrogate model for each EOP, which se-

verely limit the optimization efficiency and performance of existing surrogate-based EAs. As an emerging search paradigm incorporating transfer learning into EAs, ETO shows huge potential in addressing EOPs [149], [150], [151]. Potential research topics include the following:

- Designing advanced ETO approaches to solving EOPs by transferring the training data from other related problems.
- Designing advanced ETO approaches to solving EOPs by transferring the trained surrogate model from related problems.
- Designing advanced ETO approaches to solving EOPs by transferring the training data and the trained surrogate model from other related problems concurrently.

3) Advanced ETO Approaches for Multiform Optimization

In contrast to the existing two conceptual realizations for ETO, such as ESTO and EMT, multiform optimization (MFO) focuses on solving a single target optimization problem by employing the search experiences from some alternative formulation(s) of the target problem rather than that of other optimization problems [19]. Generally, compared to the original target problem, some alternative formulations usually have simpler search spaces, which may be related to the original target problem. Therefore, transferring useful knowledge from simpler alternative formulations can avoid inefficient evolutionary searches, thereby enhancing the optimization efficiency and performance on the original target problem. In the literature, some MFO algorithms have been developed to construct problem formulations and conduct knowledge transfer across different formulations [32], [152], [153], [154]. However, as the complexity of

problems increases, there are various new challenges for designing effective MFO approaches [155]. Possible research directions include the following:

- Designing advanced formulation construction approaches to generate useful alternate formulations for a given target optimization problem with various complex characteristics.
- Designing advanced knowledge transfer approaches to transfer useful knowledge across different problem formulations.
- Designing advanced resource allocation approaches to allocate appropriate computational resources for solving different problem formulations.

4) Theoretical Study in ETO

The research of ETO is still in the initial stage and lacks theoretical studies. In the literature, there are several theoretical studies and analyses, such as the proof of faster convergence of EMT compared to its single task counterpart [119], [156], and the exploitation of similarity between distinct tasks for positive knowledge transfer [49], [50]. To further promote the development of theoretical studies and analysis in the context of ETO, some potential research topics include the following:

- Theoretical study on the proof of faster convergence of MFO compared to its single formulation counterpart.
- Theoretical study on revealing the phenomenon of negative transfer when employ-

ing EMT to solve LSOPs.

- Theoretical study on the correlation between knowledge transfer and surrogate model for facilitating the improvement of optimization performance and efficiency in solving EOPs.

5) Integrating ETO with large language models

In recent years, large language models (LLMs) have attracted increasing attention as they have shown remarking capabilities in understanding and generating human-like text [157], [158]. Several studies have been made by integrating LLMs into the field of evolutionary computation [159], [160], [161]. However, there is little studies of integrating LLMs into ETO. Some potential research topics include the following:

- Using ETO methods to develop more effective prompts that maximize the utility of LLMs in handling various tasks.

- Using ETO methods to optimize the architectures and hyperparameters of LLMs to enhance their performance on specific tasks.

- Using LLMs to automatically design advanced ETO methods to solve various complex real-world problems.

References

[1] G. Morse and K. O. Stanley, "Simple Evolutionary Optimization Can Rival Stochastic Gradient Descent in Neural Networks," in Proceedings of the Genetic and Evolutionary Computation Conference (GECOO), 2016, pp. 477-484.

[2] G. Agrim, S. Silvio, G. Surya, and F. Li, "Embodied Intelligence via Learning and Evolution," Nature Communications, vol. 12, no. 1, 2021.

[3] Y. Zhou, T. Wang, and X. Peng, "MFEA-IG: A Multi-task Algorithm for Mobile Agents Path Planning," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2020, pp. 1-7.

[4] W. Dai, Z. Wang, and K. Xue, "System-in-package Design Using Multi-task Memetic Learning and Optimization," Memetic Computing, vol. 14, no. 1, pp. 45-59, 2021.

[5] S. Jiang, C. Xu, A. Gupta, L. Feng, Y.-S. Ong, A. Zhang, and P. Tan, "Complex and Intelligent Systems in Manufacturing," IEEE Potentials, vol. 35, no. 4, pp. 23-28, 2016.

[6] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based Software Engineering: Trends, Techniques and Applications," ACM Computing Surveys (CSUR), vol. 45, no. 1, pp. 1-61, 2012.

[7] G. B. Dantzig and N. N. Thapa, Linear Programming 1: Introduction. Berlin: Springer-Verlag, 1997.

[8] Z. Dostl, Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities,1st ed. Springer-Verlag, 2009.

[9] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge University. Press, 2004.

[10] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms. New York, NY, USA: Wiley, 2001.

[11] R. Salomon, "Evolutionary Algorithms and Gradient Search: Similarities and Differences," IEEE Transactions on Evolutionary Computation, vol. 2, no. 2, pp. 45-55, Jul. 1998.

[12] X. Qiu, J.-X. Xu, Y. Xu, and K. C. Tan, "A New Differential Evolution Algorithm for Minimax Optimization in Robust Design," IEEE Transactions on Cybernetics, vol. 48, no. 5, pp. 1355-1368, May 2018.

[13] A. Slowik and H. Kwasnicka, "Evolutionary Algorithms and Their Applications to Engineering Problems," Neural Computing and Application, vol. 32, no. 16, pp. 12363-12379, Aug. 2020.

[14] C. M. Fonseca and P. J. Fleming, "An Overview of Evolutionary Algorithms in Multiobjective Optimization," Evolutionary Computation, vol. 3, no. 1, pp.

1-16, Mar. 1995.

[15] Z. Song, H. Wang, C. He, and Y. Jin, "A Kriging-assisted Two-archive Evolutionary Algorithm for Expensive Many-objective Optimization," IEEE Transactions on Evolutionary Computation, vol. 25, no. 6, pp. 1013-1027, Dec. 2021.

[16] Z. Wang, Z. Zhan, Y. Lin, W. Yu, H. Wang, S. Kwong, and J, Zhang, "Automatic Niching Differential Evolution with Contour Prediction Approach for Multimodal Optimization Problems," IEEE Transactions on Evolutionary Computation, vol. 24, no. 1, pp. 114-128, Feb. 2020.

[17] J. Liang, X. Ban, K. Yu, B. Qu, K. Qiao, C. Yue, K. Chen, and K. C Tan, "A Survey on Evolutionary Constrained Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 27, no. 2, pp. 201-221, Apr. 2023.

[18] X. Yang, J. Zou, S. Yang, J. Zheng, and Y. Liu, "A Fuzzy Decision Variables Framework for Large-Scale Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 27, no. 3, pp. 445-459, Jun. 2023.

[19] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on Transfer Optimization: Because Experience is the Best Teacher," IEEE Transactions on Emerging Topic in Computational Intelligence, vol. 2, no. 1, pp. 51-64, Feb. 2018.

[20] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary Transfer Optimization - A New Frontier in Evolutionary Computation Research," IEEE Computational Intelligence Magazine, vol. 16, no. 1, pp. 22-33, Feb. 2021.

[21] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Transactions

on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010.

[22] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, "A Survey of Transfer Learning," Journal of Big Data, vol. 3, no. 9, 2016.

[23] L. Shao, F. Zhu, and X. Li, "Transfer Learning for Visual Categorization: A Survey," IEEE Transactions on Neural Network and Learning System, vol. 26, no. 5, pp. 1019-1034, 2015.

[24] S. Marsland, Machine Learning: An Algorithmic Perspective, 2nd ed. Chapman & Hall, 2014.

[25] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. The MIT Press, 2016.

[26] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, "Transfer Learning-Based Dynamic Multiobjective Optimization Algorithms," IEEE Transactions on Evolutionary Computation, vol. 22, no. 4, pp. 501-514, Aug. 2018.

[27] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial Evolution: Toward Evolutionary Multitasking," IEEE Transactions on Evolutionary Computation, vol. 20, no. 3, pp. 343-357, Jun. 2016.

[28] Y. Huang, L. Feng, M. Li, Y. Wang, Z. Zhu, and K. C. Tan, "Fast Vehicle Routing via Knowledge Transfer in a Reproducing Kernel Hilbert Space," IEEE Transactions on Systems, Man, Cybernetics: Systems, vol. 53, no. 9, pp. 5404-5416, Sep. 2023.

[29] K. Qiao, K. Yu, B. Qu, J. Liang, H. Song, and C. Yue, "An Evolutionary Multitasking Optimization Framework for Constrained Multiobjective Optimization

Problems," IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 263-277, Apr. 2022.

[30] S. Liu, Q. Lin, L. Feng, K. -C. Wong and K. C. Tan, "Evolutionary Multitasking for Large-Scale Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 27, no. 4, pp. 863-877, Aug. 2023.

[31] G. Li, Z. Wang, W. Gao, and L. Wang, "Decoupling Constraint: Task Clone-Based Multi-Tasking Optimization for Constrained Multi-Objective Optimization," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TE VC.2024.3358854.

[32] Y. Feng, L. Feng, S. Kwong, and K. C. Tan, "A Multi-Form Evolutionary Search Paradigm for Bi-level Multi-Objective Optimization," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2023.3332676.

[33] Y. Huang, W. Zhou, Y. Wang, M. Li, L. Feng, and K. C. Tan, "Evolutionary Multitasking With Centralized Learning for Large-Scale Combinatorial Multi-Objective Optimization," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2023.3323877.

[34] Y. Feng, L. Feng, S. Liu, S. Kwong, and K. C. Tan, "Towards Multi-Objective High-Dimensional Feature Selection via Evolutionary Multitasking", arXiv:2401.01563, https://doi.org/10.48550/arXiv.2401.01563.

[35] X. Zhou, Z. Wang, L. Feng, S. Liu, K. -C. Wong, and K. C. Tan, "Towards Evolutionary Multi-Task Convolutional Neural Architecture Search," IEEE Transactions on Evolutionary Computation, vol. 28, no. 3, pp. 682-695, Jun. 2024.

[36] Q. Xu, N. Wang, L. Wang, W. Li, and Q. Sun, "Multi-Task Optimization and Multi-Task Evolutionary Computation in the Past Five Years: A Brief Review," Mathematics, vol. 9, no. 8, Apr. 2021.

[37] X. Xue, C. Yang, L. Feng, K. Zhang, L. Song, and K. C. Tan, "Solution Transfer in Evolutionary Optimization: An Empirical Study on Sequential Transfer," IEEE Transactions on Evolutionary Computation, 2023. doi: 10.1109/TEVC.2023.3339506.

[38] L. Kotthoff, "Algorithm Selection for Combinatorial Search Problems: A Survey," arXiv, Oct. 2012. https://doi.org/10.48550/arXiv.1210.7959.

[39] M. A. Muñoz, Y. Sun, M. Kirley, and S. K. Halgamuge, "Algorithm Selection for Black-Box Continuous Optimization Problems: A Survey on Methods and Challenges," Information Sciences, vol. 317, pp. 224-245, Oct. 2015.

[40] K. Tang, F. Peng, G. Chen, and X. Yao, "Population-Based Algorithm Portfolios with Automated Constituent Algorithms Selection," Information Sciences, vol. 279, pp. 94-104, Sep. 2014.

[41] F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stuetzle, "ParamILS: An Automatic Algorithm Configuration Framework," Journal of Artificial Intelligence Research, vol. 36, pp. 267-306, Oct. 2009.

[42] H. H. Hoos, "Automated Algorithm Configuration and Parameter Tuning," Autonomous Search, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 37-71.

[43] C. Huang, Y. Li, and X. Yao, "A Survey of Automatic Parameter Tuning

Methods for Metaheuristics," IEEE Transactions on Evolutionary Computation, vol. 24, no. 2, pp. 201-216, Apr. 2020.

[44] P. Cunningham and B. Smyth, "Case-based Reasoning in Scheduling: Reusing Solution Components," International Journal of Production Research, vol. 35, no. 11, pp. 2947-2962, Nov. 1997.

[45] S. J. Louis and J. McDonnell, "Learning with Case-Injected Genetic Algorithms," IEEE Transactions on Evolutionary Computation, vol. 8, no. 4, pp. 316-328, Aug. 2004.

[46] L. Feng, Y.-S. Ong, S. Jiang, and A. Gupta, "Autoencoding Evolutionary Search with Learning across Heterogeneous Problems," IEEE Transactions on Evolutionary Computation, vol. 21, no. 5, pp. 760-772, Oct. 2017.

[47] X. Xue, C. Yang, Y. Hu, K. Zhang, Y. Cheung, L. Song, and K. C. Tan, "Evolutionary Sequential Transfer Optimization for Objective-Heterogeneous Problems," IEEE Transactions on Evolutionary Computation, vol. 26, no. 6, pp. 1424-1438, Dec. 2022.

[48] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y.-S. Ong, K. C. Tan, and A. K. Qin, "Evolutionary Multitasking via Explicit Autoencoding," IEEE Transactions on Cybernetics, vol. 49, no. 9, pp. 3457-3470, Sep. 2019.

[49] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial Evolutionary Algorithm With Online Transfer Parameter Estimation: MFEA-II," IEEE Transactions on Evolutionary Computation, vol. 24, no. 1, pp. 69-83, Feb. 2020.

[50] K. K. Bali, A. Gupta, Y. S. Ong, and P. S. Tan, "Cognizant Multitasking in

Multiobjective Multifactorial Evolution: MO-MFEA-II," IEEE Transactions on Cybernetics, vol. 51, no. 4, pp. 1784-1796, Apr. 2021.

[51] Wei, S. Wang, J. Zhong, D. Liu, and J. Zhang, "A Review on Evolutionary Multitask Optimization: Trends and Challenges," IEEE Transactions on Evolutionary Computation, vol. 26, no. 5, pp. 941-960, Oct. 2022.

[52] J. Lin, H.-L. Liu, K. C. Tan, and F. Gu, "An Effective Knowledge Transfer Approach for Multiobjective Multitasking Optimization," IEEE Transactions on Cybernetics, vol. 51, no. 6, pp. 3238-3248, Jun. 2021.

[53] J. Zhang, W. Zhou, X. Chen, W. Yao, and L. Cao, "Multi-Source Selective Transfer Framework in Multi-Objective Optimization Problems," IEEE Transactions on Evolutionary Computation, vol. 24, no. 3, pp. 424-438, Jun. 2020.

[54] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An Adaptive Archive-Based Evolutionary Framework for Many-Task Optimization," IEEE Transactions on Emerging Topic in Computational Intelligence, vol. 4, no. 3, pp. 369-384, Jun. 2020.

[55] Z. Liang, X. Xu, L. Liu, Y. Tu, and Z. Zhu, "Evolutionary Many-Task Optimization Based on Multisource Knowledge Transfer," IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 319-333, Apr. 2022.

[56] C. Wang, J. Liu, K. Wu, and Z. Wu, "Solving Multitask Optimization Problems with Adaptive Knowledge Transfer via Anomaly Detection," IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 304-318, Apr. 2022.

[57] J. Lin, H.-L. Liu, B. Xue, M. Zhang, and F. Gu, "Multiobjective Multitasking

Optimization Based on Incremental Learning," IEEE Transactions on Evolutionary Computation, vol. 24, no. 5, pp. 824-838, Oct. 2020.

[58] H. Chen, H.-L. Liu, F. Gu, and K. C. Tan, "A Multiobjective Multitask Optimization Algorithm Using Transfer Rank," IEEE Transactions on Evolutionary Computation, vol. 27, no. 2, pp. 237-250, Apr. 2023.

[59] L. Zhou et al., "Towards Effective Mutation for Knowledge Transfer in Multifactorial Differential Evolution," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2019, pp. 1541-1547.

[60] L. Zhou, L. Feng, K. C. Tan, J. Zhong, Z. Zhu, K. Liu, C. Chao, "Toward Adaptive Knowledge Transfer in Multifactorial Evolutionary Computation," IEEE Transactions on Cybernetics, vol. 51, no. 5, pp. 2563-2576, May 2021.

[61] H. Xiao, G. Yokoya, and T. Hatanaka, "Multifactorial PSO-FA Hybrid Algorithm for Multiple Car Design Benchmark," in Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC), 2019, pp. 1926-1931.

[62] Z. Tang and M. Gong, "Adaptive multifactorial particle swarm optimization," CAAI Transactions on Intelligence Technology, vol. 4, no. 1, pp. 37-46, 2019.

[63] H. Song, A. K. Qin, P. -W. Tsai, and J. J. Liang, "Multitasking Multi-Swarm Optimization," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2019, pp. 1937-1944.

[64] G. Yokoya, H. Xiao, and T. Hatanaka, "Multifactorial Optimization Using Arti-

ficial Bee Colony and Its Application to Car Structure Design Optimization," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2019, pp. 3404-3409.

[65] Z. X, K. Zhang, X. X, and J. H, "A Fireworks Algorithm Based on Transfer Spark for Evolutionary Multitasking," Frontiers in Neurorobotics, vol. 13, 2020.

[66] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, "Linearized Domain Adaptation in Evolutionary Multitasking," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 1295-1302.

[67] X. Xue, K. Zhang, K. C. Tan, L. Feng, J. Wang, G. Chen, X. Zhao, L. Zhang, and J. Yao, "Affine Transformation-Enhanced Multifactorial Optimization for Heterogeneous Problems," IEEE Transactions on Cybernetics, vol. 52, no. 7, pp. 6217-6231, Jul. 2022.

[68] L. Zhou, L. Feng, A. Gupta, and Y.-S. Ong, "Learnable Evolutionary Search Across Heterogeneous Problems via Kernelized Autoencoding," IEEE Transactions on Evolutionary Computation, vol. 25, no. 3, pp. 567-581, Jun. 2021.

[69] R. Lim, L. Zhou, A. Gupta, Y. -S. Ong, and A. N. Zhang, "Solution Representation Learning in Multi-Objective Transfer Evolutionary Optimization," IEEE Access, vol. 9, pp. 41844-41860, Mar. 2021.

[70] Z. Tang, M. Gong, Y. Wu, A. K. Qin, and K. C. Tan, "A Multifactorial Optimization Framework Based on Adaptive Intertask Coordinate System," IEEE Transactions on Cybernetics, vol. 52, no. 7, pp. 6745-6758, Jul. 2022.

[71] Z. Liang, H. Dong, C. Liu, W. Liang, and Z. Zhu, "Evolutionary Multitasking

for Multiobjective Optimization With Subspace Alignment and Adaptive Differential Evolution," IEEE Transactions on Cybernetics, vol. 52, no. 4, pp. 2096-2109, Apr. 2022.

[72] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective Multifactorial Optimization in Evolutionary Multitasking," IEEE Transactions on Cybernetics, vol. 47, no. 7, pp. 1652-1665, Jul. 2017.

[73] Y. Feng, L. Feng, S. Kwong, and K. C. Tan, "A Multi-Variation Multifactorial Evolutionary Algorithm for Large-Scale Multi-Objective Optimization," IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 248-262, Apr. 2022.

[74] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized Multitasking for Evolutionary Optimization of Expensive Problems," IEEE Transactions on Evolutionary Computation, vol. 23, no. 1, pp. 44-58, Feb. 2019.

[75] J. Liang, K. Qiao, M. Yuan, K. Yu, B. Qu, S. Ge, Y. Li, and G. Chen, "Evolutionary Multi-task Optimization for Parameters Extraction of Photovoltaic Models," Energy Conversion and Management, vol. 207, 2020.

[76] Z. Wang, L. Cao, L. Feng, M. Jiang, and K. C. Tan, "Evolutionary Multitask Optimization With Lower Confidence Bound-Based Solution Selection Strategy," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2023.3349250.

[77] Y. W. Wen and C.-K. Ting, "Parting Ways and Reallocating Resources in Evolutionary Multitasking," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 2404-2411.

[78] H. T. T. Binh, N. Q. Tuan, and D. C. T. Long, "A Multi-objective Multfactorial Evolutionary Algorithm with Reference-point-based Approach," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2019, pp. 2824-2831.

[79] E. Osaba et al., "DMFEA-II: An Adaptive Multifactorial Evolutionary Algorithm for Permutation-based Discrete Optimization Problems," 2020, arXiv:2004.06559.

[80] R.-T. Liaw and C.-K. Ting, "Evolutionary Many-tasking Based on Biocoenosis through Symbiosis: A Framework and Benchmark Problems," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 2266-2273.

[81] Q. Shang et al., "A Preliminary Study of Adaptive Task Selection in Explicit Evolutionary Many-tasking," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2019, pp. 2153-2159.

[82] L. Zhou, L. Feng, J. Zhong, Z. Zhu, B. Da, and Z. Wu, "A Study of Similarity Measure between Tasks for Multifactorial Evolutionary Algorithm," in Proceedings of the Genetic and Evolutionary Computation Conference (GECOO), 2018, pp. 229-230.

[83] C. M. Bishop, Pattern Recognition and Machine Learning. New York: Springer, 2006.

[84] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, Perspectives, and Prospects," Science, vol. 349, no. 6245, pp. 255-260, Jul. 2015.

[85] G. J. Klir and B. Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications.

Upper Saddle River, N.J: Prentice Hall PTR, 1995.

[86] J. M. Keller, M. R. Gray, and J. A. Givens, "A Fuzzy K-nearest Neighbor Algorithm," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-15, no. 4, pp. 580-585, Jul. 1985.

[87] J. Derrac, S. García, and F. Herrera, "Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental Analysis and Prospects," Information Sciences, vol. 260, pp. 98-119, Mar. 2014.

[88] A. Zhou, J. Zhang, J. Sun, and G. Zhang, "Fuzzy-Classification Assisted Solution Preselection in Evolutionary Optimization," in Proceedings of AAAI Confer on Artificial Intelligence, vol. 33, no. 01, pp. 2403-2410, Jul. 2019.

[89] J. Zhang, J. Huang, and Q. Hu, "Boosting Evolutionary Optimization via Fuzzy-Classification-Assisted Selection," Information Sciences,vol. 519, pp. 423-438, 2020.

[90] J. Zhang et al., "Environmental Selection Using a Fuzzy Classifier for Multiobjective Evolutionary Algorithms," in Proceedings of the Genetic and Evolutionary Computation Conference (GECOO), 2021, pp: 485-492.

[91] J. Zhang and H. Ishibuchi, "Multiobjective Optimization with Fuzzy Classification-Assisted Environmental Selection," in Proceedings of Evolutionary Multi-Criterion Optimization (EMO), 2021, pp. 580-592.

[92] J. Zhang, L. He, and H. Ishibuchi, "Dual-Fuzzy-Classifier-Based Evolutionary Algorithm for Expensive Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 27, no. 6, pp. 1575-1589, Aug. 2022.

[93] J. Devore, "A Modern Introduction to Probability and Statistics: Understanding Why and How," Journal of the American Statistical Association, vol. 101, no. 473, pp. 393-394, Mar. 2006.

[94] K. Deb and R. B. Agrawal, "Simulated Binary Crossover for Continuous Search Space," Complex System, vol. 9, no. 2, pp. 115-148, 1995.

[95] K. Deb and D. Deb, "Analyzing Mutation Schemes for Real-Parameter Genetic Algorithms," International Journal of Artificial Intelligence and Soft Computing, vol. 4, no. 1, pp. 1-28, 2014.

[96] B. Da et al., "Evolutionary Multitasking for Single-objective Continuous Optimization: Benchmark Problems, Performance Metric, and Baseline Results," arXiv: 1706.03470, 2017.

[97] X. Xue, C. Yang, L. Feng, K. Zhang, L. Song, and K. C. Tan, "A Scalable Problem Generator for Sequential Transfer Optimization," arXiv preprint. Oct. 2023. https://doi.org/10.48550/arXiv.2304.08503.

[98] J. Mouret and G. Maguire, "Quality Diversity for Multi-Task Optimization," in Proceedings of the Genetic and Evolutionary Computation Conference (GECOO), 2020, pp: 121-129.

[99] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A Brief Review of Domain Adaptation," Advances in Data Science and Information Engineering, 2021, pp. 877-894.

[100] J. Yin, A. Zhu, Z. Zhu, Y. Yu, and X. Ma, "Multifactorial Evolutionary Algorithm Enhanced with Cross-task Search Direction," in Proceedings of IEEE

Congress on Evolutionary Computation (CEC), 2019, pp. 2244-2251.

[101] F. Peng, K. Tang, G. Chen, and X. Yao, "Population-Based Algorithm Portfolios for Numerical Optimization," IEEE Transactions on Evolutionary Computation, vol. 14, no. 5, pp. 782-800, Oct. 2010.

[102] V. A. Shim, K. C. Tan, and H. Tang, "Adaptive Memetic Computing for Evolutionary Multiobjective Optimization," IEEE Transactions on Cybernetics, vol. 45, no. 4, pp. 610-621, Apr. 2015.

[103] E. L. Yu and P. N. Suganthan, "Ensemble of Niching Algorithms," Information Sciences, vol. 180, no. 15, pp. 2815-2833, Aug. 2010.

[104] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An Evolutionary Many-Objective Optimization Algorithm Based on Dominance and Decomposition," IEEE Transactions on Evolutionary Computation, vol. 19, no. 5, pp. 694-716, Oct. 2015.

[105] W. Wang, S. Yang, Q. Lin, Q. Zhang, K.-C Wong, C. A. C. Coello, and J. Chen, "An Effective Ensemble Framework for Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 23, no. 4, pp. 645-659, Aug. 2019.

[106] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, "Differential Evolution with Multi-Population Based Ensemble of Mutation Strategies," Information Sciences, vol. 329, pp. 329-345, Feb. 2016.

[107] S. Sharma, J. Blank, K. Deb, and B. K. Panigrahi, "Ensembled Crossover based Evolutionary Algorithm for Single and Multi-objective Optimization,"

in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 1439-1446.

[108] J. Zhou, L. Gao, and X. Li, "Ensemble of Dynamic Resource Allocation Strategies for Decomposition-Based Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 25, no. 4, pp. 710-723, Aug. 2021.

[109] R. Mallipeddi and P. N. Suganthan, "Ensemble of Constraint Handling Techniques," IEEE Transactions on Evolutionary Computation, vol. 14, no. 4, pp. 561-579, Aug. 2010.

[110] S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Decomposition-Based Multiobjective Evolutionary Algorithm With an Ensemble of Neighborhood Sizes," IEEE Transactions on Evolutionary Computation, vol. 16, no. 3, pp. 442-446, Jun. 2012.

[111] R. Xu and D. Wunsch, "Survey of Clustering Algorithms," IEEE Transactions on Neural Network, vol. 16, no. 3, pp. 645-678, May 2005.

[112] M. Makrehchi, "Hierarchical Agglomerative Clustering Using Common Neighbours Similarity," in 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI), Oct. 2016, pp. 546-551.

[113] C. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.

[114] X. He and P. Niyogi, "Locality Preserving Projections," In Proceedings of Advances in Neural Information Processing Systems (NIPS), 2003.

[115] Y. Hua, Y. Jin, and K. Hao, "A Clustering-Based Adaptive Evolutionary Algorithm for Multiobjective Optimization With Irregular Pareto Fronts," IEEE Transactions on Cybernetics, vol. 49, no. 7, pp. 2758-2770, Jul. 2019.

[116] Q. Lin, W. Lin, Z. Zhu, M. Gong, J. Li, and C. A. C. Coello, "Multimodal Multiobjective Evolutionary Optimization With Dual Clustering in Decision and Objective Spaces," IEEE Transactions on Evolutionary Computation, vol. 25, no. 1, pp. 130-144, Feb. 2021.

[117] Y. Yuan et al., "Evolutionary Multitasking for Multiobjective Continuous Optimization: Benchmark Problems, Performance Metrics and Baseline Results," 2017. [Online]. Available: arXiv:1706. 02766.

[118] P. A. Bosman and D. Thierens, "The Balance between Proximity and Diversity in Multiobjective Evolutionary Algorithms," IEEE Transactions on Evolutionary Computation, vol. 7, no. 2, pp. 174-188, Apr. 2003.

[119] L. Bai, W. Lin, A. Gupta, and Y.-S. Ong, "From Multitask Gradient Descent to Gradient-Free Evolutionary Multitasking: A Proof of Faster Convergence," IEEE Transactions on Cybernetics, vol. 52, no. 8, pp. 8561-8573, Aug. 2022.

[120] J.-Y. Li, Z.-H. Zhan, K. C. Tan, and J. Zhang, "A Meta-Knowledge Transfer-based Differential Evolution for Multitask Optimization," IEEE Transactions on Evolutionary Computation, vol. 26, no. 4, pp. 719-734, Aug. 2022.

[121] M. Friedman, "A Comparison of Alternative Tests of Significance for the Problem of m Rankings," Annals of Mathematical Statistics, vol. 11, no. 1, pp. 86-92, 1940.

[122] L. Feng et al., "An Empirical Study of Multifactorial PSO and Multifactorial DE," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 921-928.

[123] C. Yang, J. Ding, K. C. Tan, and Y. Jin, "Two-stage Assortative Mating for Multi-objective Multifactorial Evolutionary Optimization," in Proceedings of IEEE 56th Annual Conference on Decision and Control (CDC), 2017, pp. 76-81.

[124] J. Tang et al., "A Group-based Approach to Improve Multifactorial Evolutionary Algorithm," In Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2018.

[125] Z. Liu, G. Li, H. Zhang, Z. Liang, and Z. Zhu, "Multifactorial Evolutionary Algorithm Based on Diffusion Gradient Descent," IEEE Transactions on Cybernetics, 2023. doi: 10.1109/TCYB.2023.3270904.

[126] 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, vol. 28, no. 2, pp. 388-402, Apr. 2024.

[127] L. A. Zadeh, "Fuzzy Sets," Information and Control, vol. 8, no. 3, pp. 338-353, 1965.

[128] D. Dubois and H. Prade, "Fuzzy Sets, Probability and Measurement," European Journal of Operational Research, vol. 40, no. 2, pp. 135-154, Mar. 1989.

[129] A. L. Guiffrida and R. Nagi, "Fuzzy Set Theory Applications in Production Management Research: A Literature Survey," Journal of Intelligence Manu-

facturing, vol. 9, no. 1, pp. 39-56, 1998.

[130] M. C. M. Teixeira and S. H. Zak, "Stabilizing Controller Design for Uncertain Nonlinear Systems Using Fuzzy Models," IEEE Transactions on Fuzzy Systems, vol. 7, no. 2, pp. 133-142, Apr. 1999.

[131] C. Saha, S. Das, K. Pal, and S. Mukherjee, "A Fuzzy Rule-Based Penalty Function Approach for Constrained Evolutionary Optimization," IEEE Transactions on Cybernetics, vol. 46, no. 12, pp. 2953-2965, Dec. 2016.

[132] M. Orouskhani, D. Shi, and X. Cheng, "A Fuzzy Adaptive Dynamic NSGA-II With Fuzzy-Based Borda Ranking Method and its Application to Multimedia Data Analysis," IEEE Transactions on Fuzzy Systems, vol. 29, no. 1, pp. 118-128, Jan. 2021.

[133] P. Dziwiński and Ł. Bartczuk, "A New Hybrid Particle Swarm Optimization and Genetic Algorithm Method Controlled by Fuzzy Logic," IEEE Transactions on Fuzzy Systems, vol. 28, no. 6, pp. 1140-1154, Jun. 2020.

[134] X. Xia et al., "A Particle Swarm Optimization With Adaptive Learning Weights Tuned by a Multiple-Input Multiple-Output Fuzzy Logic Controller," IEEE Transactions on Fuzzy Systems, vol. 31, no. 7, pp. 2464-2478, Jul. 2023.

[135] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," International Journal of Man-Machine Studies, vol. 7, no. 1, pp. 1-13, 1975.

[136] N. Sabri et al., "Fuzzy Inference System: Short Review and Design," Inter-

national Review of Automatic Control, vol. 6, no, 4, pp. 441-449, Jul. 2013.

[137] Gu, Xiaowei et al., "Autonomous Learning for Fuzzy Systems: A Review," Artificial Intelligence Review, vol. 56, no. 8, pp. 7549-7595, 2023.

[138] P. Mamoria and D. Raj, "Comparison of Mamdani Fuzzy Inference System for Multiple Membership Functions," International Journal of Image, Graphics and Signal Processing, vol. 9, pp. 26-30, 2016.

[139] G. Selvachandran et al., "A New Design of Mamdani Complex Fuzzy Inference System for Multiattribute Decision Making Problems," IEEE Transactions on Fuzzy Systems, vol. 29, no. 4, pp. 716-730, Apr. 2021.

[140] E. V. Broekhoven and B. D. Baets, "Fast and Accurate Center of Gravity Defuzzification of Fuzzy System Outputs Defined on Trapezoidal Fuzzy Partitions," Fuzzy Sets and Systems, vol. 157, no. 7, pp. 904-918, Apr. 2006.

[141] T. Yu, and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," arXiv preprint arXiv: 2003.05689(2020).

[142] Y. Tian et al., "Evolutionary Large-scale Multi-objective Optimization: A Survey," ACM Computing Surveys (CSUR),vol 54, no. 8, pp. 1-34, 2021.

[143] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A Decision Variable Clustering-Based Evolutionary Algorithm for Large-Scale Many-Objective Optimization," IEEE Transactions on Evolutionary Computation, vol. 22, no. 1, pp. 97-112, Feb. 2018.

[144] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A Framework for Large-Scale Multiobjective Optimization Based on Problem Transformation,"

IEEE Transactions on Evolutionary Computation, vol. 22, no. 2, pp. 260-275, Apr. 2018.

[145] Y. Tian, X. Zheng, X. Zhang, and Y. Jin, "Efficient Large-Scale Multiobjective Optimization Based on a Competitive Swarm Optimizer," IEEE Transactions on Cybernetics, vol. 50, no. 8, pp. 3696-3708, Aug. 2020.

[146] J. Li, Z. Zhan, and J. Zhang, "Evolutionary Computation for Expensive Optimization: A Survey," Machine Intelligence Research, vol. 19, no. 1, pp. 3-23, 2022.

[147] Z. Song, H. Wang, C. He, and Y. Jin, "A Kriging-assisted Two-archive Evolutionary Algorithm for Expensive Many-objective Optimization," IEEE Transactions on Evolutionary Computation, vol. 25, no. 6, pp. 1013-1027, Dec. 2021.

[148] L. Pan, H. Cheng, Y. Tian, H. Wang, X. Zhang, and Y. Jin, "A Classification-based Surrogate-assisted Evolutionary Algorithm for Expensive Many-objective Optimization," IEEE Transactions on Evolutionary Computation, vol. 23, no. 1, pp. 74-88, Feb. 2019.

[149] X. Xue, Y. Hu, L. Feng, K. Zhang, L. Song, and K. C. Tan, "Surrogate-Assisted Search With Competitive Knowledge Transfer for Expensive Optimization," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2024.3478732.

[150] J. Luo, Y. Dong, Q. Liu, Z. Zhu, W. Cao, and K.C. Tan, "A New Multitask Joint Learning Framework for Expensive Multi-Objective Optimization Problems," IEEE Transactions on Emerging Topics in Computational Intelligence,

vol. 8, no. 2, pp. 1894-1909, Apr. 2024.

[151] S. Tan, Y. Wang, G. Sun, T. Pang, and K. Tang, "A Surrogate-Assisted Evolutionary Framework for Expensive Multitask Optimization Problems," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2024.3370937.

[152] R. Jiao, B. Xue, and M. Zhang, "A Multiform Optimization Framework for Constrained Multiobjective Optimization," IEEE Transactions on Cybernetics, vol. 53, no. 8, pp. 5165-5177, Aug. 2023.

[153] Y. Feng, L. Feng, S. Kwong, and K. C. Tan, "A Multivariation Multifactorial Evolutionary Algorithm for Large-Scale Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 248-262, Apr. 2022.

[154] C. Dai, X. Sun, H. Hu, W. Song, Y. Zhang, and D. Gong, "Multiform Differential Evolution With Elite-Guided Knowledge Transfer for Coal Mine Integrated Energy Systems Constrained Dispatch," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2024.3496852.

[155] Y. Feng, L. Feng, X. Xue, S. Kwong, and K. C. Tan, "A Review on Evolutionary Multiform Transfer Optimization," in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2024, pp. 1-8.

[156] J. Liu, A. Gupta, C. Ooi, and Y. -S. Ong, "ExTrEMO: Transfer Evolutionary Multiobjective Optimization With Proof of Faster Convergence," IEEE Transactions on Evolutionary Computation, 2024. doi: 10.1109/TEVC.2023.3349313.

[157] Q. Guo et al., “Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers,” arXiv preprint arXiv:2309.08532(2023).

[158] J. Cai et al., “Exploring the Improvement of Evolutionary Computation via Large Language Models,” in Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO), 2024, pp. 83-84.

[159] S. Liu, C. Chen, X. Qu, K. Tang, and Y. -S. Ong, “Large Language Models as Evolutionary Optimizers,” in Proceedings of IEEE Congress on Evolutionary Computation (CEC), 2024, pp. 1-8.

[160] Y. Huang, X. Lv, S. Wu, J. Wu, L. Feng, and K. C. Tan, “Advancing Automated Knowledge Transfer in Evolutionary Multitasking via Large Language Models,” arXiv preprint arXiv:2409.04270(2024).

[161] M. Clint, M. Jurado, and J. Zutty, “LLM Guided Evolution-The Automation of Models Advancing Models,” in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2024, pp. 377-384.