Knowledge Structure Preserving-Based Evolutionary Many-Task Optimization

Yi Jiang, *Student Member, IEEE*, Zhi-Hui Zhan, *Fellow, IEEE*, Kay Chen Tan, *Fellow, IEEE*, Sam Kwong, *Fellow, IEEE*, and Jun Zhang, *Fellow, IEEE*

***Abstract*—As a challenging research topic in evolutionary multitask optimization (EMTO), evolutionary many-task optimization (EMaTO) aims at solving more than three tasks simultaneously. The design of the EMaTO algorithm generally needs to consider two major open issues, which are how to obtain useful knowledge from similar source tasks and how to effectively transfer knowledge to the target task. In this article, we discover that knowledge structure plays a significant role in dealing with these two issues and propose a novel knowledge structure preserving-based evolutionary algorithm (KSP-EA) to efficiently solve many-task optimization problems. KSP-EA aims to achieve two goals, which are first to obtain useful structure-preserved knowledge from similar source tasks and second to effectively transfer both direct and indirect knowledge to the target task. To achieve the first goal, we propose a local-structure-preserved knowledge acquisition strategy that projects the knowledge of similar source tasks into a unified subspace without loss of the knowledge structure, thus enhancing the quality of the obtained knowledge. To achieve the second goal, we propose a tree-based knowledge propagation strategy that constructs a knowledge propagating tree to connect all the tasks and propagates knowledge along the edges of this tree. This way, the target task can obtain both direct and indirect knowledge, improving the effectiveness of knowledge transfer. We conduct extensive experiments on CEC19 and WCCI22 many-task optimization test suites and a real-world application scenario to evaluate the performance of KSP-EA. The experimental results show that our KSP-EA generally outperforms state-of-the-art algorithms.**

***Index Terms*—Evolutionary computation (EC), evolutionary many-task optimization (EMaTO), evolutionary multitask optimization (EMTO), knowledge transfer, structure-preserved knowledge, tree-based knowledge propagation (TKP).**

I. INTRODUCTION

EVOLUTIONARY multitask optimization (EMTO) is an emerging research topic in evolutionary computation (EC) [1], [2]. The EMTO paradigm is inspired by the brain's ability to solve multiple tasks simultaneously, to achieve effective optimization of several tasks in a single execution of the EC algorithm by sharing common knowledge among them [3], [4]. Due to its potential for fast convergence and better-solution accuracy [5], [6], [7], EMTO has drawn a lot of interest and has been successfully used to solve a variety of real-world multitask optimization problems [8], [9], [10].

Evolutionary many-task optimization (EMaTO) is a more challenging research paradigm in EMTO, which aims to solve more than three optimization tasks simultaneously [11], [12]. To build EMaTO algorithms that are more effective at solving the many-task optimization problems (MaTOPs), two major open issues should be taken into account. The first issue is "*knowledge acquisition*," i.e., how to obtain useful knowledge from similar source tasks. To be specific, an ideal EMaTO algorithm is needed to choose some tasks that are similar to the target task as the source tasks, since the knowledge transfer among similar tasks is more likely to be beneficial [13], [14]. The second issue is "*knowledge transfer*," i.e., how to effectively transfer knowledge to the target task. To be specific, when building the knowledge transfer strategy, the algorithms should consider how to effectively transfer knowledge and how to efficiently use this knowledge to aid the target task [15], [16], [17].

Many approaches have been proposed so far to deal with these two issues. For the first issue, i.e., how to obtain useful knowledge from similar source tasks, many studies have designed various knowledge selection strategies that aim to select the most effective knowledge from similar source tasks. The representative knowledge selection strategies include anomaly detection [17], source task selection probabilities adapting strategy [18], and intertask similarity-based knowledge selection strategies [19], [20]. For the second issue, i.e., how to effectively transfer knowledge to the target task, some different kinds of knowledge transfer strategies that aim

Manuscript received 15 August 2023; revised 27 November 2023; accepted 11 January 2024. Date of publication 18 January 2024; date of current version 2 April 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2022ZD0120001; in part by the National Natural Science Foundations of China (NSFC) under Grant 62176094 and Grant U23B2039; in part by the Tianjin Top Scientist Studio Project under Grant 24JRRCRC00030; and in part by the National Research Foundation of Korea (NRF) under Grant NRF-2022H1D3A2A01093478. This article was approved by Associate Editor H.-L. Liu. *(Corresponding authors: Zhi-Hui Zhan; Jun Zhang.)*

Yi Jiang is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China.

Zhi-Hui Zhan is with the College of Artificial Intelligence, Nankai University, Tianjin 300350, China, and also with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: zhanapollo@163.com).

Kay Chen Tan is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, SAR (e-mail: kctan@polyu.edu.hk).

Sam Kwong is with the Department of Computing and Decision Science, Lingnan University, Hong Kong, SAR (e-mail: samkwong@ln.edu.hk).

Jun Zhang is with the College of Artificial Intelligence, Nankai University, Tianjin 300350, China, and also with Hanyang University, ERICA, Ansan 15588, South Korea (e-mail: junzhang@ieee.org).

Digital Object Identifier 10.1109/TEVC.2024.3355781

to effectively transfer and utilize the knowledge were designed in existing studies. The representative categories of knowledge transfer strategies include multifactorial optimization-based knowledge transfer strategies [21], [22], [23], explicit knowledge transfer strategy [24], [25], [26], orthogonal knowledge transfer strategy [27], block level knowledge transfer strategy [28], and meta-knowledge transfer strategy [29].

Although the two aforementioned issues have been tentatively studied in the existing works and encouraging results are gained while solving MaTOPs, the existing EMaTO algorithms still have deficiencies in preserving knowledge structure and transferring indirect knowledge. First, in knowledge acquisition, most of the existing EMaTO algorithms disregard a critical factor that can affect the quality of knowledge: the knowledge structure might be damaged during the process of transferring knowledge from the space of the source task to the space of the target task. If the knowledge structure is preserved, the quality of the obtained knowledge can be enhanced. Second, in knowledge transfer, existing EMaTO algorithms just transfer knowledge from the source tasks that are directly similar to the target task (i.e., direct knowledge). However, positive indirect knowledge from other source tasks can also be helpful to the target task. To be more specific, suppose there are three different tasks T_A, T_B, and T_C, where T_A and T_B are mutually similar. If the knowledge from T_C can effectively help solve T_B, this knowledge is very likely to be helpful for T_A due to the similarity between T_A and T_B. For convenience, we call the knowledge in task T_C as the indirect knowledge for task T_A, because these two tasks are not directly similar but the knowledge of task T_C may be helpful for T_A. If both direct and indirect knowledge can be transferred, the EMaTO algorithm can achieve better performance by gaining diverse knowledge from more knowledge sources.

Aiming to preserve knowledge structure and transfer both direct and indirect knowledge to more effectively solve MaTOPs, this article proposes a novel and effective knowledge structure preserving-based evolutionary algorithm (KSP-EA). The KSP-EA has two goals. The first goal is to obtain useful knowledge from similar source tasks while preserving knowledge structure. To this goal, KSP-EA proposes a local-structure-preserved knowledge acquisition (LKA) strategy. The LKA strategy first projects the individuals of the source tasks into a unified subspace via a locality preserving projection (LPP) algorithm [30] to obtain the structure-preserved knowledge. The second goal is to effectively transfer both direct and indirect knowledge to the target task. To this goal, KSP-EA proposes a tree-based knowledge propagation (TKP) strategy. The TKP strategy first constructs a minimal spanning tree to connect all the tasks and then propagates both direct and indirect knowledge along the edges. This way, the propagated direct and indirect knowledge is effectively transferred among these tasks and can be well-utilized in the target task to generate offspring.

The characteristics and main contributions of this article are summarized as follows.

1) To obtain useful knowledge while preserving knowledge structure from similar source tasks, this article proposes the LKA strategy. The LKA strategy can obtain structure-preserved knowledge located in a unified subspace by projecting knowledge of the source tasks to the subspace. This enables KSP-EA to obtain high-quality knowledge from different source tasks.
2) To effectively transfer both direct and indirect knowledge to the target task, this article proposes the TKP strategy. The TKP strategy propagates both direct and indirect knowledge from source tasks to all the tasks along the edges of the knowledge propagating tree. This enables KSP-EA to effectively transfer both direct and indirect knowledge among multiple tasks and thus improve the effectiveness of solving MaTOPs.
3) The novel and effective KSP-EA is proposed by combining the LKA strategy and the TKP strategy. To show the effectiveness and efficiency of the proposed KSP-EA, we conduct experimental verification on the CEC19 [31] and WCCI22 [32] test suites, as well as MaTOPs in real-world applications [33]. The comparisons with several state-of-the-art EMaTO algorithms indicate that KSP-EA is more effective and more efficient.

The remainder of this article is organized as follows: the description and related works of EMTO and EMaTO are given in Section II. The proposed KSP-EA is described in Section III. The experimental studies of KSP-EA are provided in Section IV. The conclusion of this article is drawn in Section V.

II. Preliminaries

A. EMTO and EMaTO

Taking inspiration from the parallel processing capacity of human brains, the principle of EMTO is to tackle several optimization tasks concurrently in a single run of the EC algorithm. Due to the correlation between certain tasks, common knowledge can be obtained during the process of solving multiple tasks. This common knowledge can guide solving these tasks. Therefore, transferring and employing the beneficial knowledge throughout evolution can help to efficiently solve optimization tasks. Researchers have recently proposed numerous EMTO and EMaTO algorithms and displayed remarkable performance in various real-world application areas.

The corresponding optimization problem of EMTO is MTOP, which consists of multiple either related or unrelated tasks. Herein, each task is a single-objective minimization optimization problem and can be independently solved. The objective of solving an MTOP with K tasks is to find a set of K optimal solutions, each of which corresponds to a different task. The formulation of an MTOP with K tasks is shown as

$$x_k = \operatorname{argmin} f_k(X_k),\ \ x_k \in R_k,\ \ k = 1, 2, \ldots, K \tag{1}$$

where $f_k(.)$ indicates the objective function of the kth task, x_k denotes the obtained optimal solution of the kth task, and R_k denotes the search space of the kth task. Note that in MaTOP, the number K is larger than three.

The value ranges of the decision variables in different tasks are generally different, as different tasks have different search spaces. Considering this, to facilitate knowledge transfer, the value ranges for the different tasks should be normalized to $[0, 1]^D$. Without loss of generality, for the kth task, the formulation to encode the individual x_k in the original search space $[L_k, U_k]^D$ into the normalized individual y_k in the normalized search space $[0, 1]^D$ is shown as

$$y_k = \frac{x_k - L_k}{U_k - L_k} \tag{2}$$

where L_k and U_k are the lower bound and upper bound of the kth task, respectively. In EMTO and EMaTO algorithms, each individual x_k is normalized into y_k of the normalized search space $[0, 1]^D$. The knowledge transfer is carried out among the individuals in the normalized search space rather than those in the original search space.

In fitness evaluation, the normalized individuals are decoded to the original search space via

$$x_k = y_k \cdot (U_k - L_k) + L_k. \tag{3}$$

B. Related Works

As an emerging research topic in the EC community, EMTO has attracted much attention in recent years. Many EMTO and EMaTO algorithms have shown encouraging performance in solving MTOPs and MaTOPs. Existing EMTO and EMaTO algorithms generally concern two important open issues, which are how to obtain useful knowledge from similar source tasks (i.e., knowledge acquisition) and how to effectively transfer knowledge to the target task (i.e., knowledge transfer).

To deal with the first open issue, i.e., how to obtain useful knowledge from similar source tasks, many algorithms have been proposed. For example, to obtain useful knowledge in solving MaTOPs, Chen et al. [14] proposed an adaptive archive-based evolutionary framework, where the Kullback–Leibler divergence was adopted to measure the intertask similarity to obtain knowledge from a source task with a similar distribution to the target task. Liang et al. [15] proposed an EMaTO framework based on multisource knowledge transfer (EMaTO-MKT), which designed a maximum mean discrepancy-based task selection strategy to select the most similar task to the target task as the source task to obtain helpful knowledge from this source task. Wang et al. [17] designed an EMTO algorithm based on anomaly detection, which incorporated an anomaly detection model to select the most suitable knowledge. Xu et al. [18] proposed an adaptive EMTO (AEMTO) framework, where an adaptation of the source task selection probabilities mechanism was designed to choose the most related knowledge sources and determine the knowledge quantity. An efficient bi-objective knowledge transfer framework was proposed by Jiang et al. [19]. This framework designed two separate indicators of intertask similarity to precisely identify similar tasks, allowing the algorithm to learn from related source tasks. Tang et al. [34] proposed a multifactorial differential evolution (DE) with an aligned subspace continuity transfer strategy (ASCMFDE). Aiming to select helpful individuals from all tasks and obtain useful knowledge, Liaw and Ting [35] proposed a framework of EMaTO based on biocoenosis through symbiosis (EBS). Wu and Tan [36] proposed a multitasking genetic algorithm, where the knowledge was obtained via rearranging the genes of the chromosome in the source task and shifting the rearranged chromosomes to the search region of the target task.

To deal with the second open issue, i.e., how to effectively transfer knowledge to the target task, many algorithms have been proposed. For example, Gupta et al. [21] proposed a multifactorial optimization framework, which was the first study in the EMTO research field. In the multifactorial optimization framework, the individuals of different tasks are assigned different skill factors and are placed in the same population, and the knowledge is transferred among the individuals with different skill factors. Feng et al. [24] proposed an effective explicit knowledge transfer strategy, where the knowledge of the source task was mapped to the space of the target task via an explicit autoencoder, and the mapped knowledge was transferred to the target task to assist evolution. Also, this explicit knowledge transfer strategy was extended by Feng et al. [25] and Shang et al. [26], resulting in two autoencoder-based explicit EMTO algorithms that achieved encouraging performance in solving capacitated vehicle routing problems and MaTOPs, respectively. Also, Zhou et al. [37] extended the linear autoencoder into the nonlinear autoencoder to propose a kernelized autoencoding mechanism, where the knowledge from a source task was nonlinearly transferred to the search space of the target task. An island-based EMTO framework was developed by Huang et al. [38] in which knowledge was transferred among the islands. Wu et al. [27] proposed an orthogonal knowledge transfer-based EMTO algorithm, which consisted of a cross-task mapping strategy to map the knowledge from the source task to the target task. A novel idea for knowledge transfer at the block level was proposed by Jiang et al. [28], which separated the individuals of various tasks into several small blocks and then transferred knowledge based on these blocks. Experimental results indicated that the block-level knowledge transfer is efficient and effective. Li et al. [29] designed a mechanism for transferring meta-knowledge from the source task to the target task, where meta-knowledge was said to be "knowledge of knowledge" that can assist the evolution of various tasks.

Numerous existing approaches focus on how to obtain useful knowledge from similar source tasks and how to effectively transfer knowledge to the target tasks. However, the challenges of preserving knowledge structure in the knowledge acquisition stage and the challenges of transferring both direct and indirect knowledge in the knowledge transfer stage are infrequently considered by the majority of existing algorithms. First, these algorithms may break the knowledge structure during the process of obtaining knowledge, which could result in a reduction of the knowledge quality. Second, not only the direct knowledge from similar source tasks but also the indirect knowledge from indirectly similar source tasks can be helpful for the evolution of the target task. However, many current algorithms only use direct knowledge in knowledge transfer. Therefore, it is necessary to develop an EMaTO algorithm that not only can obtain useful knowledge with preserved knowledge structure from similar source tasks, but also can

Algorithm 1 KSP-EA

Input: $T_1, T_2, \ldots, T_K$ –K tasks in the MaTOP;
NP –population size of each task;
Output: $S_1, S_2, \ldots, S_K$ –optimal solutions of tasks $T_1, T_2, \ldots, T_K$;
Begin
1. Randomly initialize NP individuals for each task;
2. Obtain knowledge of each task via LKA strategy; // Algorithm 2
3. **While** not termination
4. Construct knowledge propagating tree and transfer knowledge by using the TKP strategy; //Algorithm 3
5. Obtain knowledge by using the LKA strategy; // Algorithm 2
6. Update knowledge pool of each task T_k;
7. **End**

End

effectively transfer both direct and indirect knowledge to the target task.

III. KSP-EA

A. General Framework of KSP-EA

The general framework of KSP-EA is introduced in Algorithm 1. For solving a MaTOP with K tasks, the KSP-EA first randomly initializes K populations, each of which contains NP individuals (line 1). Herein, each population corresponds to solving a single task. In the main loop of the evolutionary process, KSP-EA consists of the LKA strategy to obtain structure-preserved knowledge and the TKP strategy to effectively transfer both direct and indirect knowledge among tasks. First, the LKA strategy is executed to obtain structure-preserved knowledge by projecting the individuals of each task to a unified subspace (line 2). Each task maintains a knowledge pool to contain the obtained knowledge. Then, the TKP strategy constructs a knowledge propagating tree and transfers knowledge among all the tasks based on the knowledge propagating tree (line 4). In the TKP strategy, the knowledge propagating tree is first constructed to connect all the tasks based on the intertask similarity in the unified subspace. After that, both the direct and indirect knowledge is transferred along the edges of the knowledge propagating tree via TKP strategy, and the knowledge is utilized to generate offspring. In the TKP strategy, each target task can get knowledge from the knowledge pool of its parent task (i.e., the task corresponding to the parent node in the TKP strategy), and the knowledge pool of its parent task can contain both direct and indirect knowledge. Then, since new offspring are generated and the population of T_t is updated, the LKA strategy is carried out again to obtain the updated knowledge (line 5). Finally, the knowledge pool of the child task is updated by inheriting partial knowledge from the parent task (line 6). In other words, the knowledge pool of the child task not only contains knowledge of itself but also inherits partial knowledge from its parent task. This way, the knowledge can be propagated over all the tasks via the TKP strategy, and each task can obtain knowledge from both direct or indirect source tasks.

Algorithm 2 LKA Strategy

Input: P_k –population of task T_k;
N_n –number of neighbors of LPP;
D_s –dimensionality of the unified subspace;
Output: KP_k –knowledge pool of task T_k;
M_k –projection matrix that projects from the space of task T_k to the unified subspace;
MI_k –inversed projection matrix that projects from the unified subspace to the space of task T_k;
Begin
1. Get centralized population X of P_k via (4);
2. Construct adjacent graph of P_k;
3. **For** each individual x in P_k
4. Set N_n nearest neighbors of x as adjacent individuals;
5. Set other individuals as nonadjacent individuals;
6. **End For**
7. Calculate adjacency matrix based on (5);
8. Calculate eigenmatrix A and eigenvalues based on (7);
9. Select the columns of A corresponding to D_s smallest eigenvalues to get M_k;
10. Select the rows of inv(A) corresponding to D_s smallest eigenvalues to get MI_k;
11. $KP_k = X \times M_k$;

End

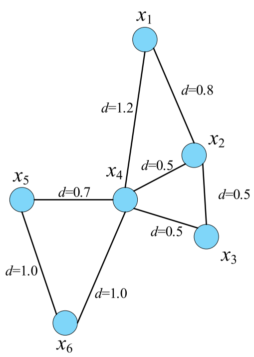
B. Obtain Knowledge via LKA Strategy

LKA strategy aims to obtain useful knowledge in a unified subspace while preserving knowledge structure. Herein, the knowledge structure refers to the adjacent relationship among these knowledge-containing individuals in the original search space. Suppose there are two individuals x_1 and x_2 in the original space S_o, and x_1 is near to x_2 measured by the Euclidean distance. The two individuals x_1 and x_2 are projected to the subspace S_p to obtain the projected individuals k_1 and k_2. Preserving knowledge structure refers to making each two adjacent individuals in the original space remain adjacent in the subspace during the projection from the original space to the subspace, as far as possible. In this sense, preserving knowledge structure requires finding an optimal projecting matrix M that can project x_1 and x_2 to the subspace S_p and ensure that the projected individuals k_1 and k_2 are still nearby.

The pseudo-code of the LKA strategy for obtaining knowledge from task T_k is shown in Algorithm 2. In this LKA strategy, the centralization operation is first performed (line 1), whose effect is to shift the mean of individuals of each task T_k to located at $[0, \ldots, 0]^D$, where D indicates the dimension of task T_k. Specifically, the formulation of the centralization operation on each individual x_i of task T_k is shown as

$$\widetilde{x_i} = x_i - \frac{1}{NP} \sum_{x \in P_k} x \tag{4}$$

where P_k stands for the population of task T_k and NP stands for the population size. After the centralization operation, a



(a)

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	0.53	0	0.24	0	0
x_2	0.53	0	0.78	0.78	0	0
x_3	0	0.78	0	0.78	0	0
x_4	0.24	0.78	0.78	0	0.62	0.37
x_5	0	0	0	0.62	0	0.37
x_6	0	0	0	0.37	0.37	0

(b)

Fig. 1. Example of constructing the adjacency graph and adjacency matrix in LPP. (a) Adjacency graph. (b) Adjacency matrix.

centralized population X composed of NP centralized individuals is obtained.

After the centralization operation, the LPP algorithm (lines 2–9) is carried out to project the individuals in the original search space into a unified subspace and obtain structure-preserved knowledge in this unified subspace. Herein, the dimensionality of the unified subspace is denoted as D_s. The LPP algorithm was proposed by He and Niyogi [30], which was designed to project high-dimensional data into a low-dimensional space and preserve the local structure of the data simultaneously. The process of the LPP algorithm is detailed in the following.

LPP first constructs an adjacency graph to connect all the individuals (line 2). In the adjacency graph, for each individual x, its N_n nearest neighbors are set to be adjacent to x, and the other individuals are set to be nonadjacent to x. Then, an adjacency matrix is calculated based on the adjacency graph (line 7). If two individuals x_i and x_j are adjacent, the corresponding weight $w_{i,j}$ of the edge connecting x_i and x_j is calculated via the heat kernel function in (5). Otherwise, if two individuals x_i and x_j are nonadjacent, the weight $w_{i,j}$ is set as zero. Specifically, let $\|x_i - x_j\|$ indicates the Euclidean distance between x_i and x_j, the calculation of each weight $w_{i,j}$ is shown as

$$w_{i,j} = \begin{cases} \exp\left(-\|x_i - x_j\|^2\right), & \text{if } x_i \text{ and } x_j \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases}. \quad (5)$$

To describe how to get the adjacency graph and the adjacency matrix, we illustrate an example in Fig. 1. In this example, the number of nearest neighbors N_n is set as 2. Fig. 1(a) illustrates the adjacency graph constructed according to the population with six individuals, where the individuals are illustrated in blue circles and d indicates the Euclidean distance between two individuals. The corresponding adjacent matrix is illustrated in Fig. 1(b).

LPP aims to preserve the local structure of the knowledge during the process of projecting the individuals from the original space to a subspace. In other words, the objective of the LPP is to ensure that if the two points are close and adjacent in the original space then these two points should be close in the subspace. Thus, finding an optimal projection matrix M^* can be achieved by minimizing the following objective function as:

$$M^* = \operatorname{argmin} \sum_i \sum_j \left(x_i M - x_j M\right)^2 \cdot w_{i,j}. \quad (6)$$

According to the justification in [30], solving the objective function in (6) is equivalent to finding the eigenvalues and eigenvectors of the following equation:

$$X^T M_L X\alpha = \lambda X^T M_D X\alpha. \quad (7)$$

where M_D is a diagonal matrix, where the ith element on the diagonal is the sum of the ith row in the adjacency matrix W. M_L is the Laplacian matrix, which is calculated by $M_L = M_D - W$. X is the matrix containing all individuals, where each row corresponds to an individual. The notations λ and α indicate an eigenvalue and an eigenvector of this generalized eigenvalue problem.

Let D and D_s stand for the dimensions of the original space and the subspace, respectively. By solving (7), we can get D eigenvalues and a $D \times D$ eigenmatrix A, where each column is an eigenvector (line 8). Then, the D_s columns in A (i.e., D_s eigenvectors) corresponding to the first D_s minimal eigenvalues are selected to compose a $D \times D_s$ projection matrix M_k for projecting the population P_k with dimension D to the subspace with dimension D_s (line 9). Also, by selecting D_s rows in the inverse matrix inv(A) corresponding to the first D_s minimal eigenvalues, we can obtain a $D_s \times D$ inversed projection matrix MI_k for projecting the knowledge in the subspace to the original search space of population P_k (line 10). Finally, the population of the kth task can be transformed into the local structure preserved knowledge in the subspace via $KP_k = X \times M_k$, where KP_k is the knowledge pool of the kth task. Herein, each row of the knowledge pool KP_k is called a knowledge entity. The knowledge pool KP_k is an $NP \times D_s$ matrix, indicating that there are NP knowledge entities in each knowledge pool and each knowledge entity is a D_s dimensional vector.

C. *Transfer Knowledge via TKP Strategy*

1) Construct Knowledge Propagating Tree: In KSP-EA, the TKP strategy aims to effectively transfer both direct and indirect knowledge from source tasks to the target task. Herein, indirect knowledge transfer refers to the propagation and utilization of knowledge between tasks to help optimize the current work by utilizing knowledge that is indirectly related to it. Through both direct and indirect knowledge transfer, tasks can acquire more diverse knowledge from different tasks to help better optimize it.

The pseudo-code of the TKP strategy is given in Algorithm 3, and the construction process of the knowledge propagating tree in the TKP strategy corresponds to lines 1–5 in Algorithm 3. To construct the knowledge propagating tree via the TKP strategy, the first step is to construct a minimal spanning tree to connect all tasks (lines 1–3). To get the minimal spanning tree, first, the distance between each two tasks is calculated. Herein, we use Euclidean distance between

Algorithm 3 TKP Strategy

Input: $KP_1, KP_2, \ldots, KP_K$–knowledge pools of tasks $T_1, T_2, \ldots, T_K$;
$P_1, P_2, \ldots, P_K$–populations of tasks $T_1, T_2, \ldots, T_K$;
σ –probability of inter-task knowledge transfer;
Output: $P_1, P_2, \ldots, P_K$–populations in the next generation;
Begin
1. Calculate distance between each two tasks via (8);
2. Construct a complete adjacent graph of the tasks;
3. Construct the minimal spanning tree based on the adjacent graph;
4. Randomly select a task as root node;
5. Obtain knowledge propagating tree *KPT*;
6. **For** $k = 1 : K$;
7. Target task T_t = the task of the kth node;
8. Source task T_s = the task of the parent node of kth node;
9. Calculate the center C_t of P_t via (9);
10. **For** each individual x_i in P_t
11. **If** rand() $\leq \sigma$
12. Randomly select a knowledge entity p_r in KP_s;
13. Project p_r to get y_r in target task space via (10);
14. SBX and polynomial mutation on x_i and y_r;
15. Selection on x_i and the offspring u_i;
16. **Else**
17. Randomly select three individuals x_1, x_2, and x_3 in P_t;
18. DE/rand/1 and binomial crossover on x_i, x_1, x_2, and x_3;
19. Selection on x_i and the offspring u_i;
20. **End If**
21. **End For**
22. Obtain knowledge of P_t via LKA strategy; // Algorithm 2
23. Update knowledge pool KP_t;
24. **End For**

End

the centers of the two knowledge pools to represent the distance between the two tasks. The equation $\text{Dis}(T_a, T_b)$ that calculates the distance between tasks T_a and T_b is shown as

$$\text{Dis}(T_a, T_b) = \left\| \frac{1}{NP} \sum_{p_i \in KP_a} p_i - \frac{1}{NP} \sum_{p_j \in KP_b} p_j \right\| \tag{8}$$

where KP_a and KP_b are the knowledge pools of tasks T_a and T_b, respectively. p_i and p_j stand for two knowledge entities in KP_a and KP_b, respectively. The size of each knowledge pool is the same as the population size NP. The distance $\text{Dis}(T_a, T_b)$ aims to evaluate the difference between tasks T_a and T_b in the unified subspace. A smaller value of distance $\text{Dis}(T_a, T_b)$ indicates the two tasks T_a and T_b are similar. Based on the distance between every two tasks, we can construct a complete adjacent graph (i.e., each node is connected to other $K - 1$ nodes) of all the tasks, and then the minimal spanning tree can be obtained according to the complete adjacent graph. To more clearly describe the process of constructing the minimal spanning tree and the knowledge propagating tree, we illustrate an example in Fig. 2. In this example, there are six tasks, and the corresponding complete adjacent graph and the minimal spanning tree are illustrated in Fig. 2(a) and (b), respectively.

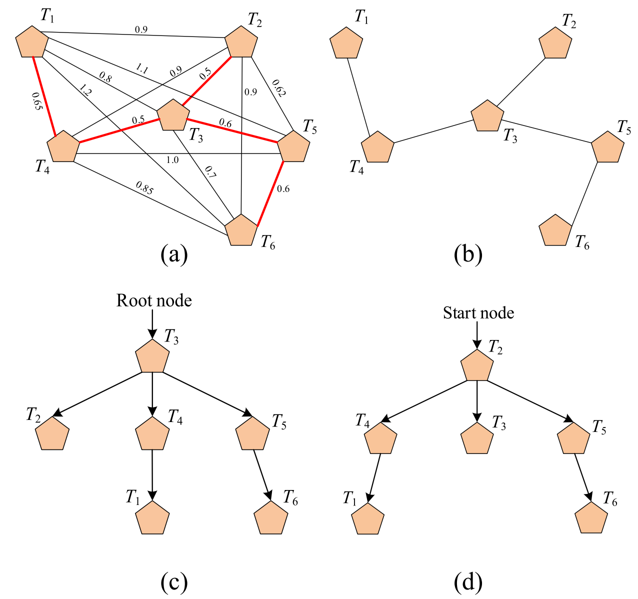


Fig. 2. Example of constructing the minimal spanning tree and knowledge propagating tree. (a) Complete adjacency graph. (b) Minimal spanning tree. (c) Knowledge propagating tree generated with T_3 as root node. (d) Knowledge propagating tree generated with T_2 as root node.

After getting the minimal spanning tree, we can construct the knowledge propagating tree (lines 4–5). First, a random node (i.e., task) of the minimal spanning tree is selected as the root node (i.e., start node). Selecting a random node as the root node can ensure that each task can receive knowledge from any other task to increase the diversity of the obtained knowledge. Starting from this root node, the knowledge can be propagated along the edges to traverse all the nodes in the minimal spanning tree. For clarity, we illustrated two examples of the knowledge propagating tree in Fig. 2(c) and (d), which corresponds to selecting T_3 as root node and selecting T_2 as root node, respectively. In the example of Fig. 2(c), the knowledge in task T_3 is first propagated to its three child nodes, i.e., T_2, T_4, and T_5. Herein, the node of task T_3 is said to be a parent node of tasks T_2, T_4, and T_5. Then, the knowledge in task T_4 is propagated to its child node T_1, and the knowledge in task T_5 is propagated to its child node T_6. For the root task T_3, the knowledge is transferred from T_3 to itself. That is, the parent node of task T_3 is itself. Note that, since the minimal spanning tree is reconstructed in each generation and the root node of the knowledge propagating tree is randomly selected, the knowledge propagating trees in different generations are also different. Therefore, knowledge can be propagated from different root tasks to other tasks.

2) Knowledge Transfer in TKP Strategy: In TKP strategy, after constructing the knowledge propagating tree, the knowledge is transferred from the parent task to the child task based on the knowledge propagating tree. The process of the knowledge transfer in the TKP strategy corresponds to lines 6–24 in Algorithm 3. As the knowledge is propagated along the edges in the knowledge propagating tree, first, the task in the kth node of the knowledge propagating tree is selected as the target task T_t, and its parent node is selected as the source task T_s (lines 7–8). Then, the center C_t of the population P_t of T_t is calculated, whose formulation is shown as

$$C_t = \frac{1}{NP} \sum_{x_i \in P_t} x_i. \tag{9}$$

For each individual, two strategies for generating offspring are randomly selected according to the parameter σ. Herein, σ indicates the probability of executing intertask knowledge transfer. If a number randomly generated in [0, 1] is smaller than or equal to σ, the intertask knowledge transfer is carried out to generate offspring (lines 12–15 in Algorithm 3). First, a random knowledge entity p_r is selected in the knowledge pool KP_s of the source task. Since p_r is located in the unified subspace, whose dimensionality is different from that of target task T_t, p_r is then projected to the space of T_t. Let y_r denotes the projected knowledge entity in the space of T_t, the formulation of the projection is shown as

$$y_r = p_r \times MI_t + C_t \tag{10}$$

where MI_t indicates the inversed projection matrix that projects from the unified subspace to the space of target task T_t. After getting y_r, the simulated binary crossover (SBX) [39], [40] and polynomial mutation [41] are orderly executed on y_r and x_i to generate a new offspring.

Otherwise, if this random number in [0, 1] is larger than σ, the new offspring is generated based on three parent individuals selected from the population P_t of the target task (lines 17–19 in Algorithm 3). Specifically, three mutually exclusive individuals x_1, x_2, and x_3 are randomly selected in P_t as parent individuals. Then, the DE/rand/1 and binomial crossover operations are conducted on these three parent individuals and x_i to generate a new offspring. The formulations of the DE/rand/1 and binomial crossover operations are, respectively, shown as

$$v_i = x_1 + F \cdot (x_2 - x_3) \tag{11}$$

$$u_i^d = \begin{cases} v_i^d, & \text{if } r^d \leq CR \text{ or } d = drand \\ x_i^d, & \text{otherwise} \end{cases} \tag{12}$$

where x_i^d indicates the dth dimension of individual x_i. u_i indicates the generated offspring of individual x_i. F and CR indicate the amplify factor and crossover rate, respectively. r^d indicates a random number in [0, 1], and $drand$ is a randomly selected dimension.

Through knowledge transfer, NP offspring of the child task are generated. Then the knowledge pool of the child task is regenerated via the LKA strategy (line 22 in Algorithm 3). As the TKP strategy is designed to transfer both direct and indirect knowledge from source tasks to target task, the child node should inherit the indirect knowledge of its parent node. This enables the knowledge of the parent node to be indirectly transferred to the child nodes of its child nodes. Therefore, the knowledge pool of the child task is updated by not only the direct knowledge of this child node but also the indirect knowledge from its parent node. Without loss of generality, as the node of source task T_s is a parent node of target task T_t, the knowledge pool KP_t is updated by both N_K knowledge entities randomly selected from the knowledge pool of its parent task T_s and NP - N_K knowledge entities randomly selected from the knowledge pool of child task T_t. N_K is an integer that equals to $\lceil N_s/2 \rceil$, where N_s indicates the number of successful offspring of T_t that are selected to enter the next generation. This way, it can be ensured that the child task inherits partial knowledge of its parent task and more than half of the knowledge entities in the knowledge pool of the child task.

IV. Experimental Studies

This section conducts extensive experiments on KSP-EA to study its performance in solving MaTOPs and to show its effectiveness and efficiency. First, we conduct a comparison between KSP-EA and several state-of-the-art EMaTO algorithms on the test suite of CEC19 competition on EMaTO [31] (denoted as CEC19). Second, we also conduct the comparison on the test suite of the WCCI22 competition on EMaTO (denoted as WCCI22) [32]. Third, we analyze the effects of the LKA strategy and the TKP strategy of KSP-EA. Fourth, we evaluate the performance of KSP-EA variants with different parameter settings to find out the optimal settings of parameters. Finally, we compare the KSP-EA with several state-of-the-art algorithms on the multitask planar kinematic arm control problems to show the effectiveness and efficiency of KSP-EA in real-world MaTOPs.

A. Experimental Settings

Two test suites are adopted as benchmarks, which are CEC19 and WCCI22. In CEC19, there are 6 MaTOP instances. Each MaTOP in CEC19 contains 50 tasks and each task is a single-objective minimization problem. In WCCI22, there are 10 MaTOP instances. Also, each MaTOP in WCCI22 contains 50 tasks and each task is a single-objective minimization problem. The 50 tasks in each MaTOP in CEC19 are generated based on the homogeneous basic functions, which means that the tasks belonging to the same MaTOP have high similarity and correlation, and the knowledge transfer between similar tasks is easier for the existing EMaTO algorithms. In contrast, the 50 tasks in each MaTOP in WCCI22 are generated based on the heterogeneous basic functions, which have a lower-intertask similarity. Therefore, the MaTOPs in WCCI22 are relatively more challenging than those in CEC19.

Five recently state-of-the-art EMaTO algorithms are adopted as the compared algorithms. These five algorithms are EBS-based genetic algorithm (EBS-GA) [35], ASCMFDE [34], MaTDE [14], AEMTO [18], and EMaTO-MKT [15]. The extensive experiments in existing studies indicated these five EMaTO algorithms can achieve relatively encouraging performance in both CEC19 and WCCI22 test suites [18], [19]. The parameters of the KSP-EA are set as follows: the population size is set as 100, the N_n and D_s in the LPP are set as 5 and 25, respectively, and the knowledge transfer probability σ is set as 0.3. The amplifier factor F and crossover rate CR in DE/rand/1 mutation and binomial

TABLE I
EXPERIMENTAL RESULTS OF KSP-EA AND THE COMPARED ALGORITHMS ON CEC19 AND WCCI22

Problem	KSP-EA	EBS-GA	ASCMFDE	MaTDE	AEMTO	EMaTO-MKT
CEC19-P1	**5.91E+01**	1.07E+03 (+)	1.35E+02 (+)	1.76E+02 (+)	9.03E+01 (+)	1.73E+02 (+)
CEC19-P2	**3.22E-06**	1.86E+01 (+)	2.02E+01 (+)	1.86E+01 (+)	1.84E+01 (+)	5.97E-02 (≈)
CEC19-P3	3.27E+02	3.25E+02 (≈)	4.27E+02 (+)	3.88E+02 (+)	4.10E+02 (+)	**1.04E+02** (−)
CEC19-P4	3.49E-04	1.80E-01 (+)	4.69E-02 (+)	5.49E-03 (+)	5.81E-03 (+)	**2.83E-04** (≈)
CEC19-P5	3.04E-03	1.42E+00 (+)	2.03E+00 (+)	1.17E-01 (+)	1.30E+00 (+)	**4.03E-04** (−)
CEC19-P6	1.20E+04	2.71E+03 (−)	1.07E+04 (+)	2.05E+00 (−)	**1.18E-01** (−)	2.93E+03 (−)
WCCI22-P1	**7.06E-11**	2.17E+00 (+)	1.48E-02 (+)	2.36E-07 (+)	5.65E-03 (+)	9.09E-09 (+)
WCCI22-P2	**6.99E+01**	2.19E+04 (+)	5.11E+02 (+)	1.22E+02 (+)	9.53E+01 (≈)	1.60E+02 (+)
WCCI22-P3	3.29E+02	3.24E+02 (−)	4.27E+02 (+)	3.43E+02 (+)	4.14E+02 (+)	**1.09E+02** (−)
WCCI22-P4	**2.63E+01**	1.07E+04 (+)	2.48E+02 (+)	1.29E+02 (+)	9.85E+01 (+)	1.47E+02 (+)
WCCI22-P5	**1.72E+01**	1.22E+02 (+)	1.44E+02 (+)	1.40E+02 (+)	1.42E+02 (+)	4.22E+01 (+)
WCCI22-P6	**1.53E+03**	1.02E+04 (+)	4.31E+03 (+)	4.03E+03 (+)	4.25E+03 (+)	2.44E+03 (+)
WCCI22-P7	**1.67E+01**	1.25E+02 (+)	1.44E+02 (+)	1.42E+02 (+)	1.43E+02 (+)	4.14E+01 (+)
WCCI22-P8	**2.96E+01**	4.17E+03 (+)	2.00E+02 (+)	2.44E+02 (+)	2.40E+02 (+)	1.48E+02 (+)
WCCI22-P9	**4.85E+02**	4.42E+03 (+)	2.05E+03 (+)	1.97E+03 (+)	2.11E+03 (+)	9.90E+02 (+)
WCCI22-P10	**5.94E+02**	1.11E+03 (+)	2.43E+03 (+)	2.39E+03 (+)	2.50E+03 (+)	1.16E+03 (+)
Number of +/≈/−		13 / 1 / 2	16 / 0 / 0	15 / 0 / 1	14 / 1 / 1	10 / 2 / 4

crossover are set as 0.5 and 0.7, respectively. In SBX, the parameter η_c is set as 2.0, and in polynomial mutation, the parameters η_m and p_m are set as 5.0 and $1/D$, respectively. The parameter settings of the compared algorithms are set as the recommended values in their original papers.

The maximum number of function evaluations is adopted as the termination condition of each algorithm execution, which is set as 5×10^6. The average fitness is adopted as the metric to evaluate the algorithms' performance, which is calculated as the average of the 50 fitness values corresponding to the 50 tasks [19]. To reduce the bias, each algorithm is executed 20 times and the mean results are adopted for performance evaluation. Besides, to further analyze the algorithms' performance in the statistic view, Wilcoxon's rank-sum test with a significance level of 5% is adopted as another metric [42]. The notations "+/≈/–" indicate KSP-EA is "superior/equivalent/inferior" to the compared algorithm according to the results of Wilcoxon's rank-sum test.

B. Comparison on CEC19 and WCCI22

The experimental results obtained by KSP-EA and the five compared algorithms on CEC19 and WCCI22 are listed in Table I. For clarity, the best result of these algorithms in each MaTOP instance is marked in **boldface**. From the overall experimental results, the proposed KSP-EA achieves significantly better performance than the compared algorithms in most MaTOPs. According to the results of Wilcoxon's rank-sum test, over the total 16 MaTOPs in CEC19 and WCCI22, KSP-EA is significantly superior to EBS-GA, ASCMFDE, MaTDE, AEMTO, and EMaTO-MKT on 13, 16, 15, 14, and 10 instances, respectively, while KSP-EA is inferior to these compared algorithms on only 2, 0, 1, 1, and 4 cases, respectively.

To discuss the performance of KSP-EA in solving multiple homogeneous and heterogeneous tasks, herein, the results at CEC19 and WCCI22 are discussed separately. For the experimental results on CEC19, where the tasks are homogeneous, the performance of KSP-EA is superior to that of EBS-GA, ASCMFDE, MaTDE, and AEMTO, while is relatively comparable to that of EMaTO-MKT. More specifically, EMaTO-MKT achieves relatively better performance than KSP-EA on P3, P4, and P5, while KSP-EA performs best on P1 and P2. This may be due to that the P3, P4, and P5 have relatively high-intertask similarity and therefore EMaTO-MKT can obtain good results by only efficiently transferring the direct knowledge. However, when comes to WCCI22, whose tasks are heterogeneous, the performance of KSP-EA is significantly superior to that of all the five compared algorithms. This evidence indicates that the KSP-EA achieves significantly better performance in solving more challenging WCCI22 and achieves comparable performance to the compared algorithms in solving the relatively less challenging CEC19. This is because the knowledge structures in homogeneous tasks can be similar and can achieve effective knowledge acquisition and knowledge transfer among these tasks, thus the KSP-EA and the compared algorithms have relatively equivalent performance. However, the knowledge acquisition and knowledge transfer strategies of most of the compared algorithms are less effective in heterogeneous tasks. Out proposed KSP-EA can preserve knowledge structure and transfer both direct and indirect knowledge. These two properties enable the KSP-EA to effectively acquire and transfer knowledge among both homogeneous tasks and heterogeneous tasks.

Besides, to further analyze the convergence behavior of KSP-EA and the compared algorithms, their convergence curves on six representative MaTOPs (including CEC19-P1, CEC19-P2, WCCI22-P1, WCCI22-P4, WCCI22-P7, and WCCI22-P10) are illustrated in Fig. 3. According to these figures, we can find the proposed KSP-EA obtains a relatively promising convergence speed than the other algorithms. In CEC19-P1 and CEC19-P2, the compared algorithms seem to be premature or trapped in the local optima after several generations, while the KSP-EA can continuously converge to approach the global optimum. Based on the above experimental analyses, we can conclude the proposed

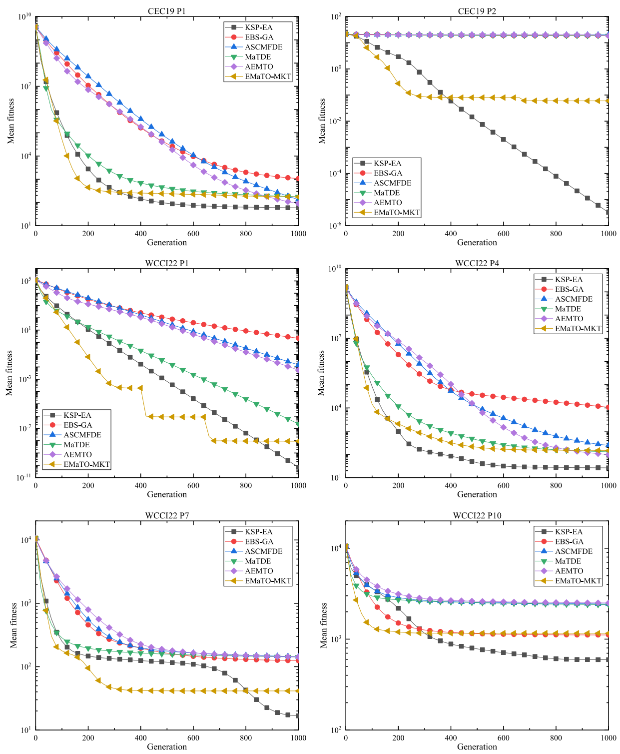


Fig. 3. Convergence curves of KSP-EA and the compared algorithms on six representative MaTOPs of CEC19 and WCCI22.

KSP-EA is generally superior to the state-of-the-art EMaTO algorithms.

C. Effect of Preserving Knowledge Structure

In the LKA strategy, the knowledge of the source task is projected to the unified subspace via the LPP algorithm to obtain lower-dimensional knowledge while preserving the knowledge structure. To analyze whether preserving the knowledge structure is effective, first, we design a KSP-EA variant that uses the dimension alignment strategy in multifactorial optimization [21] to replace the LPP (denoted as KSP-EA-MF). In this dimension alignment strategy, the individuals of the tasks with lower dimensionalities are padded with several additional decision variables to make all the individuals of different tasks have the same dimensionality. Second, we conduct a comparison between KSP-EA and a KSP-EA variant that adopts principle component analysis (PCA) [43] to replace the LPP (denoted as KSP-EA-PCA). Similar to LPP, PCA can also project knowledge from the space of source tasks to the unified subspace, and it has been adopted in some existing EMaTO algorithms [34], [44]. However, the PCA can not preserve the knowledge structure, and comparing the performance of KSP-EA, KSP-EA-PCA, and KSP-EA-MF can reflect the effect of preserving the knowledge structure. The experimental results for comparison between KSP-EA, KSP-EA-MF, and KSP-EA-PCA on 10 MaTOPs of WCCI22 are shown in Table II.

1) Compare With Multifactorial Strategy: In the table, the performance of KSP-EA is generally better than that of KSP-EA-MF on all the MaTOPs of WCCI22. Specifically,

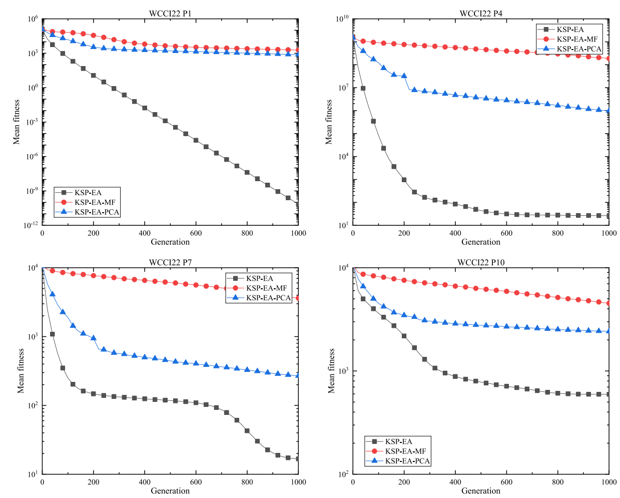


Fig. 4. Convergence curves of KSP-EA, KSP-EA-MF, and KSP-EA-PCA on four representative MaTOPs of WCCI22.

TABLE II
EXPERIMENTAL RESULTS OF KSP-EA, KSP-EA-MF, AND KSP-EA-PCA ON WCCI22

Problem	KSP-EA	KSP-EA-MF	KSP-EA-PCA
WCCI22-P1	**7.06E-11**	1.88E+03 (+)	7.43E+02 (+)
WCCI22-P2	**6.99E+01**	2.40E+08 (+)	4.51E+05 (+)
WCCI22-P3	**3.29E+02**	1.14E+04 (+)	5.77E+02 (+)
WCCI22-P4	**2.63E+01**	1.87E+08 (+)	9.83E+05 (+)
WCCI22-P5	**1.72E+01**	2.04E+03 (+)	2.48E+02 (+)
WCCI22-P6	**1.53E+03**	9.76E+07 (+)	7.73E+05 (+)
WCCI22-P7	**1.67E+01**	3.61E+03 (+)	2.69E+02 (+)
WCCI22-P8	**2.96E+01**	1.95E+08 (+)	3.67E+06 (+)
WCCI22-P9	**4.85E+02**	1.78E+08 (+)	1.67E+06 (+)
WCCI22-P10	**5.94E+02**	4.53E+03 (+)	2.42E+03 (+)
Number of +/≈/−		10 / 0 / 0	10 / 0 / 0

according to Wilcoxon's rank-sum test, KSP-EA achieves significantly superior performance to both KSP-EA-MF on all the ten MaTOPs. Besides, to better analyze the convergence behavior, the convergence curves on P1, P4, P7, and P10 of WCCI22 are illustrated in Fig. 4, and the KSP-EA achieves a generally faster convergence speed than KSP-EA-MF. Therefore, we can conclude that obtaining the structure-preserved knowledge via the LKA strategy is more effective than the multifactorial strategy.

2) Compare With PCA: Comparing the experimental results obtained by KSP-EA and KSP-EA-PCA in Table II, we can find that the performance of KSP-EA is generally better than that of KSP-EA-PCA. Specifically, according to Wilcoxon's rank-sum test, KSP-EA achieves significantly superior performance to KSP-EA-PCA on all the 10 MaTOPs. This is mainly because obtaining knowledge via PCA can generally damage the knowledge structure, which can decrease the effectiveness of knowledge transfer. However, obtaining knowledge via LPP can better preserve the knowledge structure, which enables the KSP-EA to get generally promising performance. Also, the convergence of KSP-EA and KSP-EA-PCA are illustrated in Fig. 4. According to these figures, the observation is that the convergence speed of KSP-EA is faster than that of KSP-EA-PCA during the whole evolutionary process on all these four MaTOPs.

TABLE III
EXPERIMENTAL RESULTS OF KSP-EA AND KSP-EA-W/O-TKP ON WCCI22

Problem	KSP-EA	KSP-EA-w/o-TKP
WCCI22-P1	**7.06E-11**	1.82E-06 (+)
WCCI22-P2	**6.99E+01**	1.21E+02 (+)
WCCI22-P3	**3.29E+02**	3.76E+02 (+)
WCCI22-P4	**2.63E+01**	8.86E+01 (+)
WCCI22-P5	**1.72E+01**	1.29E+02 (+)
WCCI22-P6	**1.53E+03**	3.30E+03 (+)
WCCI22-P7	**1.67E+01**	1.29E+02 (+)
WCCI22-P8	**2.96E+01**	1.06E+02 (+)
WCCI22-P9	**4.85E+02**	1.45E+03 (+)
WCCI22-P10	**5.94E+02**	1.40E+03 (+)
Number of +/≈/−		10 / 0 / 0

Concerning the mean fitness and the convergence speed, the performance of KSP-EA dominates the performance of KSP-EA-MF and KSP-EA-PCA, and the LKA strategy is effective for KSP-EA. Therefore, we can conclude that preserving knowledge structure is effective for solving MaTOPs.

D. Effect of Transfer Direct and Indirect Knowledge

1) Effect of TKP Strategy: In the proposed KSP-EA, the TKP strategy is designed to effectively transfer both direct and indirect knowledge to the target task. Thus, it is necessary to analyze whether transferring both direct and indirect knowledge via the TKP strategy is effective. We first design a KSP-EA variant named KSP-EA-w/o-TKP, where the TKP strategy is eliminated and the knowledge is directly transferred from a random source task to the target task. By comparing the performance between KSP-EA and KSP-EA-w/o-TKP, we can analyze the effects of transferring both direct and indirect knowledge.

The experimental results obtained by KSP-EA and KSP-EA-w/o-TKP on WCCI22 are provided in Table III. Based on these results, we can conclude that the KSP-EA generally outperforms KSP-EA-w/o-TKP on WCCI22. According to Wilcoxon's rank-sum test, on all the 10 MTOPs, the results obtained by KSP-EA are significantly superior to that obtained by KSP-EA-w/o-TKP. Considering the results for fitness, KSP-EA also outperforms KSP-EA-w/o-TKP. This is because transferring both direct and indirect knowledge to the target task can enhance the quality of the knowledge and thus enhance the effectiveness of KSP-EA. The observation of the comparison indicates the TKP strategy can effectively transfer knowledge and is necessary for KSP-EA.

We illustrate the convergence curves of KSP-EA and KSP-EA-w/o-TKP in Fig. 5 to help analyze their convergence behavior. According to the curves, we can find the convergence speed of KSP-EA is generally faster than that of KSP-EA-w/o-TKP. Specifically, in WCCI22-P1, WCCI22-P4, and WCCI22-P7, KSP-EA converges faster throughout the evolutionary process. In WCCI22-P10, KSP-EA has a slower convergence speed in the first 200 generations, while the convergence speed after 200 generations is better than that of KSP-EA-w/o-TKP. Therefore, based on the above analyses, we can conclude that transferring both direct and indirect knowledge among tasks can enhance effectiveness.

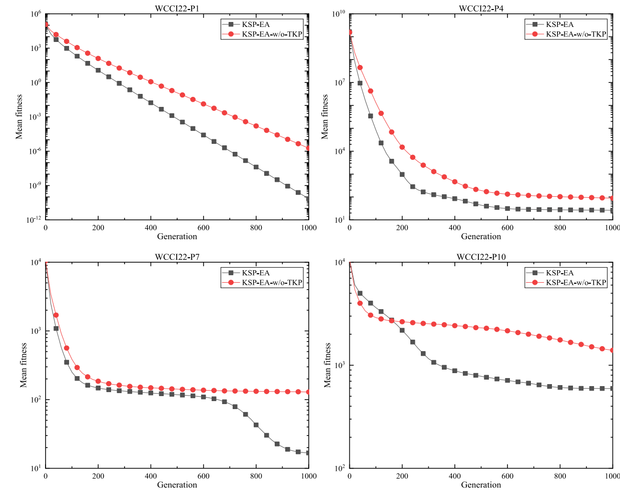


Fig. 5. Convergence curves of KSP-EA and KSP-EA-w/o-TKP on four representative MaTOPs of WCCI22.

TABLE IV
EXPERIMENTAL RESULTS OF KSP-EA, KSP-EA-KLD, AND KSP-EA-MMD ON WCCI22

Problem	KSP-EA	KSP-EA-KLD	KSP-EA-MMD
WCCI22-P1	**7.06E-11**	1.73E+01 (+)	3.30E-06 (+)
WCCI22-P2	**6.99E+01**	4.61E+04 (+)	1.49E+02 (+)
WCCI22-P3	**3.29E+02**	4.96E+02 (+)	3.78E+02 (+)
WCCI22-P4	**2.63E+01**	1.78E+04 (+)	8.07E+01 (+)
WCCI22-P5	**1.72E+01**	1.82E+02 (+)	1.29E+02 (+)
WCCI22-P6	**1.53E+03**	1.92E+04 (+)	3.21E+03 (+)
WCCI22-P7	**1.67E+01**	1.85E+02 (+)	1.29E+02 (+)
WCCI22-P8	**2.96E+01**	6.59E+03 (+)	1.30E+02 (+)
WCCI22-P9	**4.85E+02**	5.84E+03 (+)	1.26E+03 (+)
WCCI22-P10	**5.94E+02**	7.90E+02 (+)	1.48E+03 (+)
Number of +/≈/−		10 / 0 / 0	10 / 0 / 0

2) Compare With Existing Direct Knowledge Transfer Strategies: In this part, we compare the TKP strategy with several transfer strategies in state-of-the-art EMaTO algorithms that only transfer direct knowledge. In these direct knowledge transfer strategies, the similarity between each pair of tasks is calculated first and the knowledge is transferred between each pair of tasks that have the highest similarity. Two KSP-EA variants are designed by replacing the TKP strategy with two different knowledge transfer strategies. These two KSP-EA variants are KSP-EA with Kullback–Leibler divergence (KSP-EA-KLD) and KSP-EA with maximum mean discrepancy (KSP-EA-MMD), which adopt the Kullback–Leibler divergence and the maximum mean discrepancy as the similarity measurement and directly transfer knowledge between the most similar tasks, respectively. In the existing EMaTO algorithm, the Kullback–Leibler divergence and the maximum mean discrepancy were shown to be able to accurately calculate the intertask similarity [14], [15]. However, KSP-EA-KLD and KSP-EA-MMD can only achieve direct knowledge transfer. Comparing the performance of KSP-EA, KSP-EA-KLD and KSP-EA-MMD can better reflect the effects of indirect knowledge transfer.

The experimental results of KSP-EA, KSP-EA-KLD, and KSP-EA-MMD on WCCI22 are shown in Table IV. Besides,

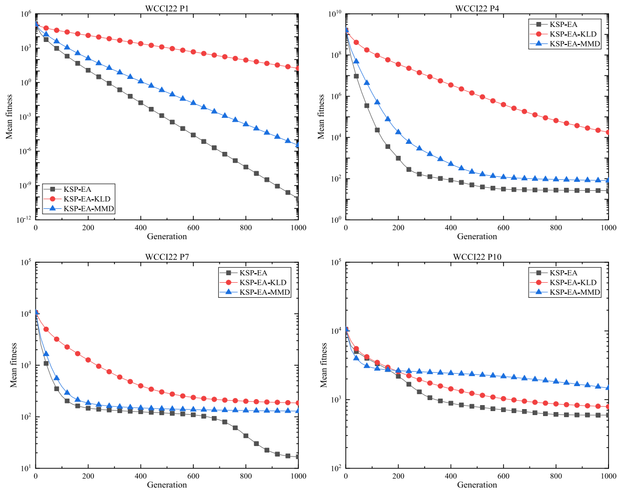


Fig. 6. Convergence curves of KSP-EA, KSP-EA-KLD, and KSP-EA-MMD on four representative MaTOPs of WCCI22.

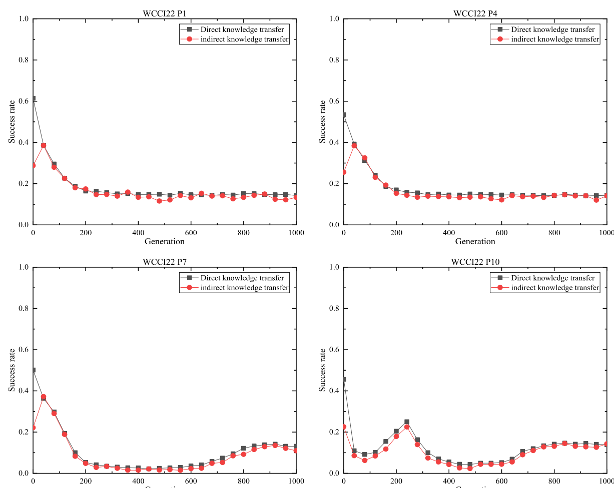


Fig. 7. Curves of the success rate of direct knowledge transfer and indirect knowledge transfer on four representative MaTOPs of WCCI22.

the convergence curves of KSP-EA, KSP-EA-KLD, and KSP-EA-MMD on P1, P4, P7, and P10 of WCCI22 are shown in Fig. 6. In the results, we can find that the KSP-EA achieves generally better performance than KSP-EA-KLD, and KSP-EA-MMD on most of the MaTOPs. Also, the convergence speed of KSP-EA is faster than that of KSP-EA-KLD and KSP-EA-MMD. Therefore, we can conclude that combining the advantages of both direct knowledge transfer and indirect knowledge transfer is more effective than only using direct knowledge transfer.

3) Success Rate of Direct- and Indirect Knowledge Transfer: Besides, to intuitively show the effects of direct knowledge transfer and indirect knowledge transfer, we illustrate the curves of the success rates of direct knowledge transfer and indirect knowledge transfer on several representative MaTOPs of WCCI22 in Fig. 7. The success rate of direct knowledge transfer (or indirect knowledge transfer) is defined as the proportion of the successful offspring over all the offspring generated by direct knowledge transfer (or indirect knowledge transfer). Herein, the successful offspring is the

TABLE V
COMPARISON AMONG KSP-EAS WITH DIFFERENT N_n ON WCCI22

Problem	$N_n = 5$	$N_n = 10$	$N_n = 20$	$N_n = 50$
WCCI22-P1	7.06E-11	1.16E-10	4.97E-11	**1.38E-11**
WCCI22-P2	6.99E+01	7.25E+01	7.46E+01	**6.77E+01**
WCCI22-P3	3.29E+02	3.17E+02	2.90E+02	**2.04E+02**
WCCI22-P4	2.63E+01	2.52E+01	**2.42E+01**	2.47E+01
WCCI22-P5	**1.72E+01**	1.74E+01	1.79E+01	1.97E+01
WCCI22-P6	**1.53E+03**	1.54E+03	1.55E+03	1.60E+03
WCCI22-P7	**1.67E+01**	1.69E+01	1.77E+01	1.97E+01
WCCI22-P8	2.96E+01	**2.59E+01**	3.06E+01	2.79E+01
WCCI22-P9	**4.85E+02**	4.93E+02	5.19E+02	5.46E+02
WCCI22-P10	**5.94E+02**	6.19E+02	6.40E+02	6.63E+02
Mean fitness	**3.09E+02**	3.13E+02	3.16E+02	3.17E+02
Mean rank	**2.10**	2.40	2.80	2.70

offspring selected to enter the next generation. Specifically, let NSO indicates the number of successful offspring and NTO indicates the number of total offspring generated by direct knowledge transfer (or indirect knowledge transfer), the success rate SR of direct knowledge transfer (or indirect knowledge transfer) is calculated via

$$SR = \frac{NSO}{NTO}. \tag{13}$$

In the curves of Fig. 7, we can find that the success rate of indirect knowledge transfer is similar to that of direct knowledge transfer. During the evolutionary process, the proportion of successful offspring generated by indirect knowledge transfer is similar to that generated by direct knowledge transfer. Therefore, indirect knowledge transfer is as effective as direct knowledge transfer, and they can both transfer positive knowledge to enhance the search capacity of the algorithm.

E. Parameter Analysis

In the proposed KSP-EA, there are three important parameters, which are N_n, D_s, and σ. Herein, N_n is an integer that controls the number of neighbor nodes in LPP, D_s is an integer that controls the dimensionality of the unified subspace in LPP, and σ is a number in [0, 1] that controls the probability of knowledge transfer. Since the settings of these three parameters can influence the performance of KSP-EA, these parameters should be carefully set. To find out their optimal settings, we conduct the following three experiments.

1) Analysis of N_n: N_n controls the number of neighbor nodes in LPP. To find out the optimal setting of N_n, we conduct a comparison among the KSP-EA variants with different settings of N_n, which are $N_n = 5$, $N_n = 10$, $N_n = 20$, and $N_n = 50$. The experimental results of this comparison on WCCI22 are provided in Table V. We calculate the mean fitness among all these 10 MaTOPs and the mean rank obtained by these KSP-EA variants in the last two rows of Table V. KSP-EA with $N_n = 5$ shows generally better performance than the other settings of N_n. Also, the KSP-EA with $N_n = 5$ achieves the best result on five MaTOPs of WCCI22, i.e., P5, P6, P7, P9, and P10, while $N_n = 5$, $N_n = 10$, and $N_n = 20$ obtain the best results on 3, 1, and 1 MaTOP, respectively. Therefore, the parameter N_n is recommended to be set as 5.

TABLE VI
COMPARISON AMONG KSP-EAS WITH DIFFERENT D_s ON WCCI22

Problem	$D_s = 0.1 \cdot D_{min}$	$D_s = 0.2 \cdot D_{min}$	$D_s = 0.5 \cdot D_{min}$	$D_s = 0.8 \cdot D_{min}$
WCCI22-P1	2.90E-10	**1.79E-11**	7.06E-11	2.53E-09
WCCI22-P2	1.01E+02	7.57E+01	6.99E+01	**6.88E+01**
WCCI22-P3	1.02E+02	**7.26E+01**	3.29E+02	3.39E+02
WCCI22-P4	7.11E+01	5.06E+01	2.63E+01	**2.40E+01**
WCCI22-P5	4.11E+01	2.84E+01	**1.72E+01**	2.80E+01
WCCI22-P6	1.84E+03	1.73E+03	1.53E+03	**1.48E+03**
WCCI22-P7	4.18E+01	2.85E+01	**1.67E+01**	3.83E+01
WCCI22-P8	7.37E+01	5.49E+01	**2.96E+01**	3.14E+01
WCCI22-P9	6.84E+02	6.29E+02	**4.85E+02**	5.01E+02
WCCI22-P10	8.26E+02	6.80E+02	5.94E+02	**5.81E+02**
Mean fitness	3.78E+02	3.35E+02	**3.09E+02**	**3.09E+02**
Mean rank	3.70	2.50	**1.70**	2.10

TABLE VII
COMPARISON AMONG KSP-EAS WITH DIFFERENT σ ON WCCI22

Problem	$\sigma = 0.1$	$\sigma = 0.3$	$\sigma = 0.5$	$\sigma = 0.9$
WCCI22-P1	6.84E-07	**7.06E-11**	2.31E-10	1.39E-03
WCCI22-P2	7.35E+01	**6.99E+01**	7.96E+01	2.76E+02
WCCI22-P3	3.73E+02	3.29E+02	**6.54E+01**	9.91E+01
WCCI22-P4	3.26E+01	**2.63E+01**	3.05E+01	7.11E+01
WCCI22-P5	1.19E+02	**1.72E+01**	2.24E+01	3.80E+01
WCCI22-P6	5.75E+05	**1.53E+03**	1.63E+03	1.86E+03
WCCI22-P7	1.20E+02	**1.67E+01**	2.24E+01	3.86E+01
WCCI22-P8	3.04E+02	2.96E+01	**2.82E+01**	1.16E+02
WCCI22-P9	1.16E+03	**4.85E+02**	5.57E+02	7.19E+02
WCCI22-P10	6.46E+02	**5.94E+02**	7.23E+02	8.82E+02
Mean fitness	5.78E+04	**3.09E+02**	3.16E+02	4.10E+02
Mean rank	3.40	**1.30**	2.00	3.30

2) Analysis of D_s: The integer parameter D_s controls the dimensionality of the unified subspace in LKA strategy. If the value of D_s is set as a relatively small value, the obtained knowledge in the unified subspace can be very different from the knowledge in the original space, which may cause the degradation of the quality of the obtained knowledge. If the value of D_s is set as a relatively large number, the generalization of the obtained knowledge in the unified subspace is hard to guarantee, which can diminish the effectiveness of knowledge transfer. Therefore, the value of D_s should be carefully set.

To find out the optimal setting of D_s, we conduct a comparison among the KSP-EA variants with different settings of D_s. Let $D_{\min}$ indicates the minimal dimensionality of all the tasks, these settings are $D_s = 0.1 \cdot D_{\min}$, $D_s = 0.2 \cdot D_{\min}$, $D_s = 0.5 \cdot D_{\min}$, and $D_s = 0.8 \cdot D_{\min}$. The experimental results of this comparison on WCCI22 are provided in Table VI. The performance of $D_s = 0.5 \cdot D_{\min}$ is generally superior to that of the other settings of D_s. Besides, over all the 10 MaTOPs, the KSP-EA with $D_s = 0.5 \cdot D_{\min}$ achieves the best result in four cases, which is superior to the other KSP-EA variants. Therefore, $D_s = 0.5 \cdot D_{\min}$ is a relatively good setting for KSP-EA.

3) Analysis of σ: σ indicates the probability of executing the intertask knowledge transfer, whose value range should be [0, 1]. On the one hand, if the value of σ is set as a relatively small value, the algorithm will more focus on the intratask crossover, which can ignore the effectiveness of knowledge transfer. On the other hand, if the value of σ is set as a relatively large value, the algorithm will more focus on the intertask knowledge transfer. In this case, as the knowledge transfer is less effective between irrelated tasks, the performance of KSP-EA can be diminished if most of the tasks are irrelated or dissimilar. Therefore, we should analyze the performance of different σ and find out the optimal value of σ.

Herein, the experimental results obtained by four KSP-EA variants with different σ are given in Table VII. These four KSP-EA variants are denoted as $\sigma = 0.1$, $\sigma = 0.3$, $\sigma = 0.5$, and $\sigma = 0.9$. According to the results for mean fitness and mean rank, the $\sigma = 0.3$ gets the best-mean fitness and mean rank among all the four variants. Furthermore, over the 10 MaTOPs in WCCI22, the KSP-EA variant with $\sigma = 0.3$ obtains the best result in eight cases, which is significantly better than the other variants. Therefore, the value of σ should be 0.3.

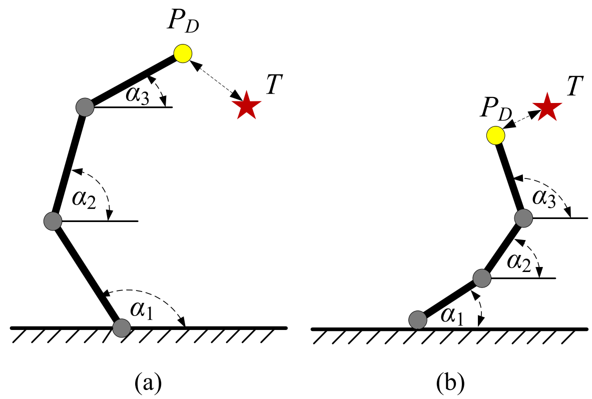


Fig. 8. Two-task planar kinematic arm control problem. (a) Task1. (b) Task2.

F. Comparison on Real-World MaTOP

This section verifies the effectiveness of the proposed KSP-EA in addressing real-world application problems by analyzing its performance in a multitask planar kinematic arm control application scenario [33]. The goal of the multitask planar kinematic arm control problem is to optimize the joint angles of multiple planar kinematic arms with different arm lengths and maximum motion angles so that the tips of these arms can be as close as possible to the target. For ease of understanding, we show a graphical example of this problem with two tasks in Fig. 8. The tip of each planar kinematic arm is denoted as P_D, the target is denoted as T. and the angles of all d joints are denoted as $[\alpha_1, \alpha_2, \ldots, \alpha_d]$. To be specific, the objective of the multitask planar kinematic arm control problem is to find a set of optimal angles $[\alpha_1, \alpha_2, \ldots, \alpha_d]$ to minimize the distance between P_D and T. The objective function $f_t(.)$ of the tth task is shown as

$$f_t(\alpha_1, \alpha_2, \ \ldots, \alpha_d) = \|P_D - T\|. \tag{14}$$

The multitask planar kinematic arm control problem contains multiple tasks, each of which corresponds to the control of a single arm. The total lengths L_t and upper bound of the angles αmax_t are different for different arms. Here, the parameter settings of the multitask planar kinematic arm control

TABLE VIII
RESULTS OF KSP-EA AND THE COMPARED ALGORITHMS ON PLANAR KINEMATIC ARM CONTROL PROBLEMS WITH FEW TASKS

Problem	BLKT-DE	MFEA-AKT	MFEA-II	AEMTO	ASCMFDE
	MDF	MDF	MDF	MDF	MDF
5 tasks	**7.76E-05**	1.19E-03 (+)	1.43E-04 (+)	1.30E-04 (≈)	1.04E-04 (+)
10 tasks	**7.33E-05**	5.08E-04 (+)	1.66E-04 (+)	1.14E-04 (+)	9.74E-05 (+)
Number of + / ≈ / −		2 / 0 / 0	2 / 0 / 0	1 / 1 / 0	2 / 0 / 0

TABLE IX
RESULTS OF KSP-EA AND THE COMPARED ALGORITHMS ON PLANAR KINEMATIC ARM CONTROL PROBLEMS WITH MANY TASKS

Problem	KSP-EA	AEMTO	EBS-DE	MaTDE	EMaTO-MKT
	MDF	MDF	MDF	MDF	MDF
50 tasks	**1.02E-04**	1.17E-04 (+)	2.18E-04 (+)	1.84E-04 (+)	2.76E-04 (+)
100 tasks	**9.22E-05**	1.13E-04 (+)	2.17E-04 (+)	1.59E-04 (+)	2.05E-04 (+)
Number of + / ≈ / −		2 / 0 / 0	2 / 0 / 0	2 / 0 / 0	2 / 0 / 0

problem are the same as the previous studies [19], [28]. We use the difference between the current fitness and the minimum fitness obtained by all algorithms as the experimental results in order to present the results more intuitively. The mean difference between the current fitness and the minimum fitness (denoted as *MDF*) is calculated via

$$MDF = \frac{1}{|F|}\sum_{f\in F}(f - f_{\min}) \tag{15}$$

where F represents all of the fitness values that the algorithms obtained on the current task, and $f_{\min}$ represents the minimum fitness value that all of the algorithms obtained on the current task.

To evaluate the performance of the KSP-EA on the planar kinematic arm control problem with different numbers of tasks, in this section, KSP-EA is experimented on planar kinematic arm control problem with few tasks (i.e., 5 and 10 tasks) and planar kinematic arm control problem with many tasks (i.e., 50 and 100 tasks), respectively. First, for the experiment with few tasks, the KSP-EA is compared with the state-of-the-art algorithms that have shown promising performance on MaTOPs with few tasks, including MFEA with adaptive knowledge transfer (MFEA-AKT) [23], MFEA-II [45], AEMTO [18], and ASCMFDE [34]. The experimental results are shown in Table VIII. The performance of KSP-EA is generally better than that of the compared algorithms. To be specific, according to Wilcoxon's rank-sum test, when the task number is 5, KSP-EA can get significantly superior performance to MFEA-AKT, MFEA-II, and ASCMFDE and competitive performance to AEMTO, while when the task number is 10, KSP-EA can get significantly superior performance to all compared algorithms.

Second, for the experiment on planar kinematic arm control problem with many tasks, the KSP-EA is compared with the state-of-the-art EMaTO algorithms that have shown promising performance on MaTOPs with many tasks, including AEMTO [18], EBS-DE [35], MaTDE [14], and EMaTO-MKT [15]. Table IX displays the experimental results of KSP-EA and those of the compared algorithms. KSP-EA performs generally better than AEMTO, EBS-DE, MaTDE, and EMaTO-MKT on these problems with many tasks, which is similar to the experimental results on the problem with few tasks. Wilcoxon's rank-sum test indicates that KSP-EA can outperform all other compared algorithms by a significant margin when the task number is 50 or 100. Thus, we can infer from the above analysis and results that, when applied to real-world MaTOPs, the KSP-EA has better performance than the compared state-of-the-art algorithms.

V. CONCLUSION

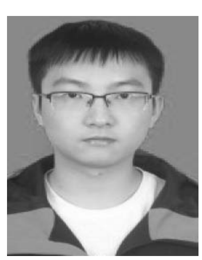
This article proposes the KSP-EA for solving MaTOPs. KSP-EA breaks through the limitations of existing algorithms, as it can obtain useful knowledge while preserving knowledge structure and effectively transferring both direct and indirect knowledge among the tasks. The proposed KSP-EA contains two main components, i.e., the LKA strategy and the TKP strategy. The LKA strategy projects the knowledge from the space of the source task to a unified subspace while preserving the knowledge structure. The TKP strategy builds a minimal spanning tree to connect all the tasks and propagates both direct and indirect knowledge among tasks. Concerning the experiments on CEC19 and WCCI22 benchmark test suites, the performance of KSP-EA is significantly superior to several state-of-the-art EMaTO algorithms.

In our future work, we will further develop the KSP-EA to enhance its performance for solving more difficult MaTOPs. Besides, we will also extend the idea of preserving knowledge structure in solving the real-world MaTOPs with more challenging properties, such as multiobjective [46], [47], [48], large-scale [49], [50], and multimodal optimization [51], [52], [53].

REFERENCES

[1] Z.-H. Zhan, L. Shi, K. C. Tan, and J. Zhang, "A survey on evolutionary computation for complex continuous optimization," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 59–110, Jan. 2022.
[2] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization-A new frontier in evolutionary computation research," *IEEE Comput. Intell. Mag.*, vol. 16, no. 1, pp. 22–33, Feb. 2021.
[3] T. Wei, S. Wang, J. Zhong, D. Liu, and J. Zhang, "A review on evolutionary multitask optimization: Trends and challenges," *IEEE Trans. Evol. Comput.*, vol. 26, no. 5, pp. 941–960, Oct. 2022.
[4] Y. Jiang, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Knowledge learning for evolutionary computation," *IEEE Trans. Evol. Comput.*, early access, May 19, 2023, doi: 10.1109/TEVC.2023.3278132.
[5] Z. Tang, M. Gong, Y. Wu, A. K. Qin, and K. C. Tan, "A multifactorial optimization framework based on adaptive intertask coordinate system," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 6745–6758, Jul. 2022.
[6] G. Li, Q. Zhang, and Z. Wang, "Evolutionary competitive multitasking optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 278–289, Apr. 2022.
[7] H. Han, X. Bai, H. Han, Y. Hou, and J. Qiao, "Self-adjusting multitask particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 1, pp. 145–158, Feb. 2022.
[8] J.-Y. Li, Z.-H. Zhan, Y. Li, and J. Zhang, "Multiple tasks for multiple objectives: A new multiobjective optimization method via multitask optimization," *IEEE Trans Evol. Comput.*, early access, Jul. 11, 2023, doi: 10.1109/TEVC.2023.3294307.
[9] H. Li, Y.-S. Ong, M. Gong, and Z. Wang, "Evolutionary multitasking sparse reconstruction: Framework and case study," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 733–747, Oct. 2019.
[10] Z.-G. Chen, Z.-H. Zhan, S. Kwong, and J. Zhang, "Evolutionary computation for intelligent transportation in smart cities: A survey," *IEEE Comput. Intell. Mag.*, vol. 17, no. 2, pp. 83–102, May 2022.
[11] B. H. T. Thanh, L. Van Cuong, T. B. Thang, and N. H. Long, "Ensemble multifactorial evolution with biased skill-factor inheritance for many-task optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 6, pp. 1735–1749, Dec. 2023.

[12] L. T. Thanh, L. Van Cuong, T. B. Thang, and H. T. T. Binh, "Multi-armed bandits for many-task evolutionary optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2021, pp. 1664–1671.
[13] S. Huang, J. Zhong, and W.-J. Yu, "Surrogate-assisted evolutionary framework with adaptive knowledge transfer for multi-task optimization," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 4, pp. 1930–1944, Oct.–Dec. 2021.
[14] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An adaptive archive-based evolutionary framework for many-task optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 3, pp. 369–384, Jun. 2020.
[15] Z. Liang, X. Xu, L. Liu, Y. Tu, and Z. Zhu, "Evolutionary many-task optimization based on multi-source knowledge transfer," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 319–333, Apr. 2022.
[16] Z. Liang, W. Liang, Z. Wang, X. Ma, L. Liu, and Z. Zhu, "Multiobjective evolutionary multitasking with two-stage adaptive knowledge transfer based on population distribution," *IEEE Trans. Syst. Man Cybern., Syst.*, vol. 52, no. 7, pp. 4457–4469, Jul. 2022.
[17] C. Wang, J. Liu, K. Wu, and Z. Wu, "Solving multitask optimization problems with adaptive knowledge transfer via anomaly detection," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 304–318, Apr. 2022.
[18] H. Xu, A. K. Qin, and S. Xia, "Evolutionary multitask optimization with adaptive knowledge transfer," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 290–303, Apr. 2022.
[19] Y. Jiang, Z.-H. Zhan, K. C. Tan, and J. Zhang, "A bi-objective knowledge transfer framework for evolutionary many-task optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 5, pp. 1514–1528, Oct. 2023.
[20] S.-H. Wu, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Transferable adaptive differential evolution for many-task optimization," *IEEE Trans. Cybern.*, vol. 53, no. 11, pp. 7295–7308, Nov. 2023.
[21] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.
[22] J. Yin, A. Zhu, Z. Zhu, Y. Yu, and X. Ma, "Multifactorial evolutionary algorithm enhanced with cross-task search direction," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 2244–2251.
[23] L. Zhou et al., "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2563–2576, May 2021.
[24] L. Feng et al., "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3457–3470, Sep. 2019.
[25] L. Feng et al., "Explicit evolutionary multitasking for combinatorial optimization: A case study on capacitated vehicle routing problem," *IEEE Trans. Cybern.*, vol. 51, no. 6, pp. 3143–3156, Jun. 2021.
[26] Q. Shang et al., "A preliminary study of adaptive task selection in explicit evolutionary many-tasking," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 2153–2159.
[27] S.-H. Wu, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Orthogonal transfer for multitask optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 1, pp. 185–200, Feb. 2023.
[28] Y. Jiang, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Block-level knowledge transfer for evolutionary multitask optimization," *IEEE Tran. Cybern.*, vol. 54, no. 1, pp. 558–571, Jan. 2024.
[29] J.-Y. Li, Z.-H. Zhan, K. C. Tan, and J. Zhang, "A meta-knowledge transfer-based differential evolution for multitask optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 4, pp. 719–734, Aug. 2022.
[30] X. He and P. Niyogi, "Locality preserving projections," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 153–160.
[31] L. Feng, K. Qin, A. Gupta, Y. Yuan, Y.-S. Ong, and X. Chi, "IEEE CEC 2019 competition on evolutionary multitask optimization," presented at *IEEE Congr. Evol. Comput.*, Wellington, New Zealand, 2019. [Online]. Available: http://www.bdsc.site/websites/MTO_competiton_2019/MTO_Competition_CEC_2019.html
[32] L. Feng et al., "IEEE CEC 2022 competition on evolutionary multitask optimization," presented at *IEEE World Congr. Comput. Intell.*, Padova, Italy, 2022. [Online]. Available: http://www.bdsc.site/websites/MTO_competition_2021/MTO_Competition_WCCI_2022.html
[33] J. B. Mouret and G. Maguire, "Quality diversity for multi-task optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2020, pp. 121–129.
[34] Z. Tang, M. Gong, Y. Wu, W. Liu, and Y. Xie, "Regularized evolutionary multitask optimization: Learning to intertask transfer in aligned subspace," *IEEE Trans. Evol. Comput.*, vol. 25, no. 2, pp. 262–276, Apr. 2021.
[35] R.-T. Liaw and C.-K. Ting, "Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems," in *Proc. IEEE Congr. Evol. Comput.*, 2017, pp. 2266–2273.
[36] D. Wu and X. Tan, "Multitasking genetic algorithm (MTGA) for fuzzy system optimization," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 6, pp. 1050–1061, Jun. 2020.
[37] L. Zhou, L. Feng, A. Gupta, and Y.-S. Ong, "Learnable evolutionary search across heterogeneous problems via kernelized autoencoding," *IEEE Trans. Evol. Comput.*, vol. 25, no. 3, pp. 567–581, Jun. 2021.
[38] Y. Huang, L. Feng, A. K. Qin, M. Chen, and K. C. Tan, "Toward large-scale evolutionary multitasking: A GPU-based paradigm," *IEEE Trans. Evol. Comput.*, vol. 26, no. 3, pp. 585–598, Jun. 2022.
[39] K. Deb, K. Sindhya, and T. Okabe, "Self-adaptive simulated binary crossover for real-parameter optimization," in *Proc. 9th Annu. Conf. Genet. Evol. Comput.*, 2007, pp. 1187–1194.
[40] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.
[41] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Informat.*, vol. 26, no. 4, pp. 30–45, 1996.
[42] J. Carrasco, S. Garcia, M. M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," *Swarm Evol. Comput.*, vol. 54, May 2020, Art. no. 100665.
[43] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[44] Z. Liang, H. Dong, C. Liu, W. Liang, and Z. Zhu, "Evolutionary multitasking for multiobjective optimization with subspace alignment and adaptive differential evolution," *IEEE Trans. Cybern.*, vol. 52, no. 4, pp. 2096–2109, Apr. 2022.
[45] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 69–83, Feb. 2020.
[46] M. Wu, K. Li, S. Kwong, Y. Zhou, and Q. Zhang, "Matching-based selection with incomplete lists for decomposition multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 554–568, Aug. 2017.
[47] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 909–923, Dec. 2014.
[48] K. Li, Á. Fialho, S. Kwong, and Q. Zhang, "Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 18, no. 1, pp. 114–130, Feb. 2014.
[49] Z.-H. Zhan, Z.-J. Wang, H. Jin, and J. Zhang, "Adaptive distributed differential evolution," *IEEE Trans. Cybern.*, vol. 50, no. 11, pp. 4633–4647, Nov. 2020.
[50] Z.-J. Wang, J.-R. Jian, Z.-H. Zhan, Y. Li, S. Kwong, and J. Zhang, "Gene targeting differential evolution: A simple and efficient method for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 4, pp. 964–979, Aug. 2023.
[51] Y. Jiang, Z.-H. Zhan, K. C. Tan, and J. Zhang, "Optimizing niche center for multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 53, no. 4, pp. 2544–2557, Apr. 2023.
[52] Y. Jiang, C.-H. Chen, Z.-H. Zhan, Y. Li, and J. Zhang, "Adversarial differential evolution for multimodal optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, 2022, pp. 1–8,
[53] Z.-G. Chen, Z.-H. Zhan, H. Wang, and J. Zhang, "Distributed individuals for multiple peaks: A novel differential evolution for multimodal optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 4, pp. 708–719, Aug. 2020.



Yi Jiang (Student Member, IEEE) received the B.S. degree in computer science and technology from the South China University of Technology, Guangzhou, China, in 2020, where he is currently pursuing the Ph.D. degree in computer science and technology with the School of Computer Science and Engineering.

His research interests mainly include computational intelligence, machine learning, and their applications in real-world problems.

Zhi-Hui Zhan (Fellow, IEEE) received the bachelor's and Ph.D. degrees in computer science from Sun Yat-Sen University, Guangzhou China, in 2007 and 2013, respectively.

He is currently a Changjiang Scholar Young Professor and a Gifted Professor with the College of Artificial Intelligence, Nankai University, Tianjin, China. His current research interests include evolutionary computation, swarm intelligence, and their applications in real-world problems.

Prof. Zhan was a recipient of the IEEE Computational Intelligence Society Outstanding Early Career Award in 2021, the Outstanding Youth Science Foundation from National Natural Science Foundations of China in 2018, and the Wu Wen-Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017. He is listed as the World's Top 2% Scientist for both Career-Long Impact and Year Impact in Artificial Intelligence, and is listed as the Elsevier Highly Cited Chinese Researcher in Computer Science. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS: SYSTEMS, and the IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE.

Kay Chen Tan (Fellow, IEEE) received the B.Eng. (First Class Hons.) and Ph.D. degrees from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is currently a Chair Professor (Computational Intelligence) with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. He has published over 300 refereed articles and seven books.

Prof. Tan is currently the Vice-President (Publications) of IEEE Computational Intelligence Society, USA. He served as the Editor-in-Chief for the IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE from 2010 to 2013 and the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2015 to 2020. He currently serves as the Chief Co-Editor for Book Series on *Machine Learning: Foundations, Methodologies, and Applications* (Springer), and as an Editorial Board Member for more than ten journals.

Sam Kwong (Fellow, IEEE) received the Ph.D. degree from the University of Hagen, Hagen, Germany, in 1996.

He is currently a Chair Professor of Computing and Decision Science with Lingnan University, Hong Kong. His research interests include pattern recognition, evolutionary computations, and video analytics.

Prof. Kwong was elevated to an IEEE Fellow for his contributions to optimization techniques for cybernetics and video coding in 2014. He is the Juniou Past President of the IEEE SYSTEMS, MAN, AND CYBERNETICS (SMC). He was also appointed as an IEEE Distinguished Lecturer of the IEEE SMC Society in March 2017. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION.

Jun Zhang (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the City University of Hong Kong, Hong Kong, in 2002.

His research interests include computational intelligence and power electronic circuits. His research contributions span over 500 peer-reviewed publications, of which more than 210 appear in IEEE TRANSACTIONS.

Prof. Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011, was appointed as a Cheung Kong Chair Professor in 2013 by the Ministry of Education, China, and the Brain Pool Fellow by the National Research Foundation of Korea. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE and the IEEE TRANSACTIONS ON CYBERNETICS.