Mathematical Biosciences and Engineering

MBE, 21(2): 2432–2457.
DOI: 10.3934/mbe.2024107
Received: 31 October 2023
Revised: 14 December 2023
Accepted: 05 January 2024
Published: 16 January 2024

http://www.aimspress.com/journal/MBE

Research article

An adaptive multitasking optimization algorithm based on population distribution

Xiaoyu Li[1,2], Lei Wang[1,3,*], Qiaoyong Jiang[1] and Qingzheng Xu[4]

[1] School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China
[2] School of Electronic and Information Engineering, Ankang University, Ankang 725000, China
[3] The Key Laboratory of Industrial Automation of Shaanxi Province, Shaanxi University of Technology, Hanzhong 723001, China
[4] College of Information and Communication, National University of Defense Technology, Wuhan 430019, China

* **Correspondence:** E-mail: leiwang_lw@126.com.

Abstract: Evolutionary multitasking optimization (EMTO) handles multiple tasks simultaneously by transferring and sharing valuable knowledge from other relevant tasks. How to effectively identify transferred knowledge and reduce negative knowledge transfer are two key issues in EMTO. Many existing EMTO algorithms treat the elite solutions in tasks as transferred knowledge between tasks. However, these algorithms may not be effective enough when the global optimums of the tasks are far apart. In this paper, we study an adaptive evolutionary multitasking optimization algorithm based on population distribution information to find valuable transferred knowledge and weaken the negative transfer between tasks. In this paper, we first divide each task population into K sub-populations based on the fitness values of the individuals, and then the maximum mean discrepancy (MMD) is utilized to calculate the distribution difference between each sub-population in the source task and the sub-population where the best solution of the target task is located. Among the sub-populations of the source task, the sub-population with the smallest MMD value is selected, and the individuals in it are used as transferred individuals. In this way, the solution chosen for the transfer may be an elite solution or some other solution. In addition, an improved randomized interaction probability is also included in the proposed algorithm to adjust the intensity of inter-task interactions. The experimental results on two multitasking test suites demonstrate that the proposed algorithm achieves high solution accuracy and fast convergence for most problems, especially for problems with low relevance.

Keywords: evolutionary multitasking optimization; differential evolution; population distribution information; maximum mean difference; similarity

1. Introduction

Evolutionary multitasking optimization is inspired by the extraordinary talent of people to perform multiple tasks simultaneously [1]. In 2016, Gupta et al. [2] introduced the multifactor genetic mechanism into evolutionary algorithms and proposed the multifactor evolutionary algorithm (MFEA) for the first time, which is considered to be the pioneering work in multitasking evolutionary algorithms. Compared with traditional single-task evolutionary algorithms, MFEA shows superiority, but MFEA suffers from slow convergence, easy to trap in local optimum local optimality, and negative migration. Subsequently, researchers have conducted extensive studies on EMTO and have made some progress. Liang et al. [3] introduced hyper-rectangular search and gene mapping strategies into MFEA to construct similar gene representation space while expanding the search space to improve transfer efficiency and accelerate population convergence. Bali et al. [4] introduced a linear-domain adaptive strategy of search space to improve MFEA (LDA-MFEA), which reduces inter-task differences and facilitates positive migration between tasks. Ong et al. [5] introduced online parameter learning in MFEA to adaptively adjust the strength of transfer knowledge and reduce negative migration. Yang et al. [6] used K-means clustering to cluster individuals in a population and selected different random interaction probabilities for different categories of variables, improving the efficiency of knowledge transfer between tasks. Liaw et al. [7] considered the information exchange between all tasks as a symbiotic relationship and proposed a generalized framework, SBO, that is applicable to multitasking problems. Feng et al. [8] applied a two-population framework to perform mapping from source domain to target domain via a single-layer denoising autoencoder for explicit knowledge transfer among tasks. Li et al. [9] developed the multi-population multitasking evolutionary optimization framework (MPEF), which introduces a multi-population evolutionary framework into multitasking optimization, allowing different optimization tasks to be handled by different evolutionary algorithms. Cai et al. [10] adopted a hybrid knowledge transfer strategy to select the corresponding transfer strategy according to the correlation between tasks.

In EMTO, the identification of valuable knowledge between tasks, that is, the selection of transfer content, is a factor affecting the performance of the EMTO algorithm. Existing research has shown that transferring elite solutions between two relevant tasks may accelerate the convergence of algorithms, but if these elites are local optima, they will lead the population to the local optima [11]. Therefore, determining the valuable content to transfer between tasks is an important issue. Recently, in order to improve the performance of multi-task optimization algorithms, many methods have been proposed to identify valuable knowledge between tasks. Gao et al. [12] proposed an EMTO algorithm based on semi-supervised learning that identifies individuals containing useful knowledge and selects them for transfer between tasks. In [13], the individuals in the source task that are superior to the optimal individuals in the target task are chosen to constitute the transferred population. Then, the excellent individuals in the transfer population are combined with the target population to construct a Gaussian mixture distribution model and generate the offspring according to the mixture model to realize the knowledge sharing between tasks. Lin et al. [14] used the neighbors of positive transferred individuals obtained during the evolutionary process as the next generation of transfer solutions,

achieving automatic identification of valuable transfer solutions during the evolutionary process. Sun et al. [15] used solutions in the source task that were similar to non-dominated solutions in the target task as transfer solutions. In [16], the authors utilized an incremental naïve Bayes classifiers to determine the individuals to transfer. In [17], the authors used anomaly detection to determine the individuals to transfer, and identified individuals carrying valuable knowledge, transferring them to the target task. Chen et al. [18] introduced the concept of transfer rank to quantify the priority of individuals and select transferred content based on the quantified priority.

Although these algorithms have shown the potential to solve the EMTO problem, their ability to find valuable transferred solutions needs to be further improved. The main purpose of EMTO is to transfer useful knowledge between tasks to facilitate the convergence of the target task, assuming that the co-optimized tasks are related. However, in reality many optimization problems are black-box optimizations without prior knowledge of task similarity. Therefore, how to identify and select knowledge to transfer between tasks becomes a key factor affecting the performance of EMTO. In addition, since evolutionary algorithms are based on populations, which contain rich information about data distribution, extracting and utilizing the feedback information of populations in the evolutionary process for the design of the EMTO algorithm helps to improve the performance of the algorithm. The elite solutions in the population are generally located near the global optimal solution; thus, finding solutions with similar distributions to the elites in the target task as the migration content in the resource task can accelerate the convergence of the target task.

In order to achieve more effective and robust knowledge transfer between different optimization tasks, this article proposes a single-objective EMTO algorithm based on population distribution. The proposed algorithm uses differential algorithm as a task solver, namely, AMTDE-PD. In addition, an improved randomized mating probability is also included in the proposed algorithm to adjust the intensity of the knowledge transfer. The primary contributions of this article are twofold.

1) The transfer content selection strategy based on population distribution information is proposed. This strategy can adaptively select the knowledge to be transferred between tasks, and the transfer solutions may be not only elite solutions but also other solutions.

2) Design an adaptive interaction probability to adjust the interaction intensity between tasks. The method adjusts the interaction probability between tasks based on their evolutionary trends to mitigate negative transfer between tasks.

The rest of this article is organized as follows. Section 2 gives the preliminaries about EMTO, and DE algorithm. The implementation details of AMTDE-PD are presented in Section 3. Section 4 provides and analyzes the experimental results. Section 5 gives the conclusion of this article.

2. Preliminaries

2.1. EMTO

EMTO algorithms use evolutionary algorithms as task solvers to exploit potential synergies or complementarities between tasks to simultaneously solve multiple tasks with different search spaces [2]. That is to say, the evolutionary multitasking optimization algorithm returns the optimal solutions for multiple tasks through one search. Assuming that there are Θ minimization single-objective problems to be optimized simultaneously, T_θ denotes the θth task, X_θ is the search space corresponding to the task T_θ, and its objective function is $F_\theta: X_\theta \to R$, R is the real set, the mathematical

representation of the multitasking optimization problem can be described as below:

$$\{x_1^*, x_2^*, \cdots x_\Theta^*\} = argmin\{F_1(x_1), \cdots, F_\Theta(x_\Theta)\}$$

$$s.t.\ \ x_\theta \in X_\theta \tag{1}$$

where $\mathbf{x}_1^*$ is the best solution of T_θ. $\mathbf{x}_\theta$ is a feasible solution in the search space X_θ.

During multitasking optimization process, since each optimization task has its own search space, the evolutionary multitasking algorithm has to map these tasks corresponding to different search spaces to a unified search space before solving these tasks and optimizing them using the corresponding evolutionary algorithm. Suppose the dimension of the search space for the θth task is denoted as D_θ, the dimension of the unified search space U is given as: $D_U = max\{D_1, D_2, \cdots, D_\Theta\}$, where U $\subseteq$ $[0,1]^{D_U}$. The solution x_θ in task T_θ are encoded into the uniform search space by

$$u_\theta^d = \frac{(x_\theta^d - L_\theta^d)}{(H_\theta^d - L_\theta^d)} \tag{2}$$

where u_θ^d is the dth dimension of the individual x_θ in U. H_θ^d and L_θ^d denote the upper and lower bound of dth dimension of X_θ. Conversely, decoding u_θ into the original search space via

$$x_\theta^d = u_\theta^d \times \left(H_\theta^d - L_\theta^d\right) + L_\theta^d \tag{3}$$

2.2. Differential evolution algorithm

The differential evolution (DE) algorithm is one of the most efficient evolutionary algorithms for solving continuous optimization problems [19]. In this paper, we use DE as a task solver. DE evolves a population by performing a random initialization in the search space and then performing mutation, crossover, and selection operations. A mutation strategy is used in the DE algorithm to generate a mutation vector for each $\mathrm{x_i}$ in the population [20]. An advanced mutation strategy in DE is as follows:

$$\text{DE/current-to-pbest/1:}\ \ \mathbf{v}_i = \mathbf{x}_i + F.\left(\mathbf{x}_{pbest} - \mathbf{x}_i\right) + F.\left(\mathbf{x}_{r1} - \tilde{\mathbf{x}}_{r2}\right) \tag{4}$$

where $\mathrm{v_i}$ is a mutant vector, $\mathrm{x_{pbest}}$ is randomly chosen from the top p% individuals in the population, $\mathrm{p} \in (0,1]$. $\mathrm{x_{r1}}$ is a random solution in the population. $\tilde{\mathrm{x}}_{\mathrm{r2}}$ is randomly chosen from the union of the population and its external archive Arc. Arc is used to preserve the solutions that were eliminated during the evolutionary process. F is the scaling factor.

After performing the mutation operation, DE uses the following binomial crossover to generate a new trial vector $\mathbf{u}_i$ [21]:

$$\mathbf{u}_{i,d} = \begin{cases} \mathbf{v}_{i,d}, & \text{if rand} < CR_i \ \ or\ d == jrand \\ \mathbf{x}_{i,d}, & \text{otherwise} \end{cases} \tag{5}$$

where CR is the crossover factor applied to determine the number of offspring individuals obtaining variables from the mutant vector $\mathbf{v}_i$, CR $\in$ [0,1], updated with reference to the literature [10]. $rand$ is a random number between 0 and 1. $jrand \subseteq$ [1, D], and D is the dimension of the search space.

After the crossover operation is executed, the following selection operation is used for the minimization problem to generate the next generation of new population individuals [22]:

$$\mathbf{x}_i = \begin{cases} \mathbf{u}_i, & if\ f(\mathbf{u}_i) < f(\mathbf{x}_i) \\ \mathbf{x}_i, & otherwise \end{cases} \tag{6}$$

where $f(\mathbf{u}_i)$ and $f(\mathbf{x}_i)$ denote the objective function value of the trial vector $\mathbf{u}_i$ and individual $\mathbf{x}_i$, respectively. This operation ensures that the newly generated population is at least not worse than the previous generation's population.

Due to its simplicity and effectiveness, and the fact that it has been widely used for multitasking solving [23–25], AMTDE-PD uses DE as a task solver.

3. Proposed EMTO algorithm

In the last few years, EMTO algorithms have been widely used for solving various problems [26], vehicle routing problems [27–30], traveling salesman problems [31], path planning problems [32], and hyperspectral image classification [33]. However, the common knowledge between different tasks has not been fully explored in traditional single-objective EMTO algorithms. In this article, we present a single-objective multitasking optimization algorithm based on population distribution to use the common knowledge between tasks. In this section, we first give the main framework of AMTDE-PD. Next, the transfer content selection strategy and inter-task knowledge transfer strategy are introduced. Then, an improved adaptive mating probability strategy is presented. Finally, the improved adaptive mating probability strategy is given.

3.1. Main framework

The flowchart of AMTDE-PD is illustrated in Figure 3. Algorithm 1 shows the main framework of AMTDE-PD. First, a randomly initialized population containing N individuals is assigned to each task, and all individuals in the population are evaluated on the corresponding task. At the beginning of each iteration, the subpopulation division operation is performed on target and resource tasks, respectively. Subsequently, the transferred population is constructed based on subpopulation (lines 6 and 7). Then, N offspring were generated for each task using two different methods (lines 8–17). At the end of each iteration, the total successful evolution rate SR_t of the target population and the center-position distance between the target population and the resource population are calculated (lines 18 and 19). Next, the random interaction probability between tasks is updated based on the change in the distance of the center location between populations. To adjust the interaction probability RMP_t of the current task T_t, $(\mathrm{t} \in \Theta)$, we first check whether the total evolutionary success rate SR_t (the number of offspring in the population that are better than their parents/the population size) of the population P_t is larger than a pre-specified threshold δ. If $SR_t > \delta$, it means that the population can achieve better evolution using the current RMP, and there is no need to update the RMP; otherwise, if $SR_t < \delta$, it means that the RMP between the populations needs to be adjusted. Finally, the optimal solutions for all tasks are output.

Algorithm 1 Main Framework of AMTDE-PD

Input: N : population size of each task;
Θ : number of tasks;
$T_1, \cdots T_\Theta$: all tasks;
K : number of clusters;
$MaxFES$: maximum function evaluation;
RMP_0 : initial value of interaction probability for each task;
δ: The threshold for interaction probability adjustment;
q: control parameters;

Output: the optimal solutions for all tasks

Initialize and evaluate each population $P_\theta(\theta = 1,2,\cdots,\Theta)$;
$FES = \Theta * N$; $g = 1$
while $FES \le MaxFES$ **do**
  Perform subpopulation division operations for target and resource tasks; (see Algorithm 2)
  The MMD values of the clusters where the optimal solution of the target task is located and all the clusters in the source task are calculated using Eq (10);
  Use Eqs (11) and (12) to construct the transferred population;
  for $t = 1:\Theta$
    for $i = 1:N$
      if $r\ and\ < RMP_t$
        Use Eq (13) to generate the mutant individual $\mathbf{v}_i^t$;
      else
        Use Eq (14) to generate the mutant individual $\mathbf{v}_i^t$;
      end if
      Perform the crossover operation on $\mathbf{x}_i^t$ and $\mathbf{v}_i^t$ using Eq (5) to produce a trial vector $\mathbf{u}_i$ and evaluate $\mathbf{u}_i$ on task T_t;
      Perform selection operation using Eq (6) to generate the individual of the next generation population;
    end for
    Calculate the total successful evolution rate SR_t of the population;
    Calculate the center-position distance $Dist_{\mathrm{t,s}}$ between target task T_t and source task T_l according to Eq (7);
    if $SR_t < \delta$
      $RMP_t(g+1) = Update\big(RMP_t(g), Dist_{\mathrm{t,s}}, q\big)$ (see Algorithm 3);
    end if
  end for
  $FES = FES + N$;
  $g = g + 1$;
end while

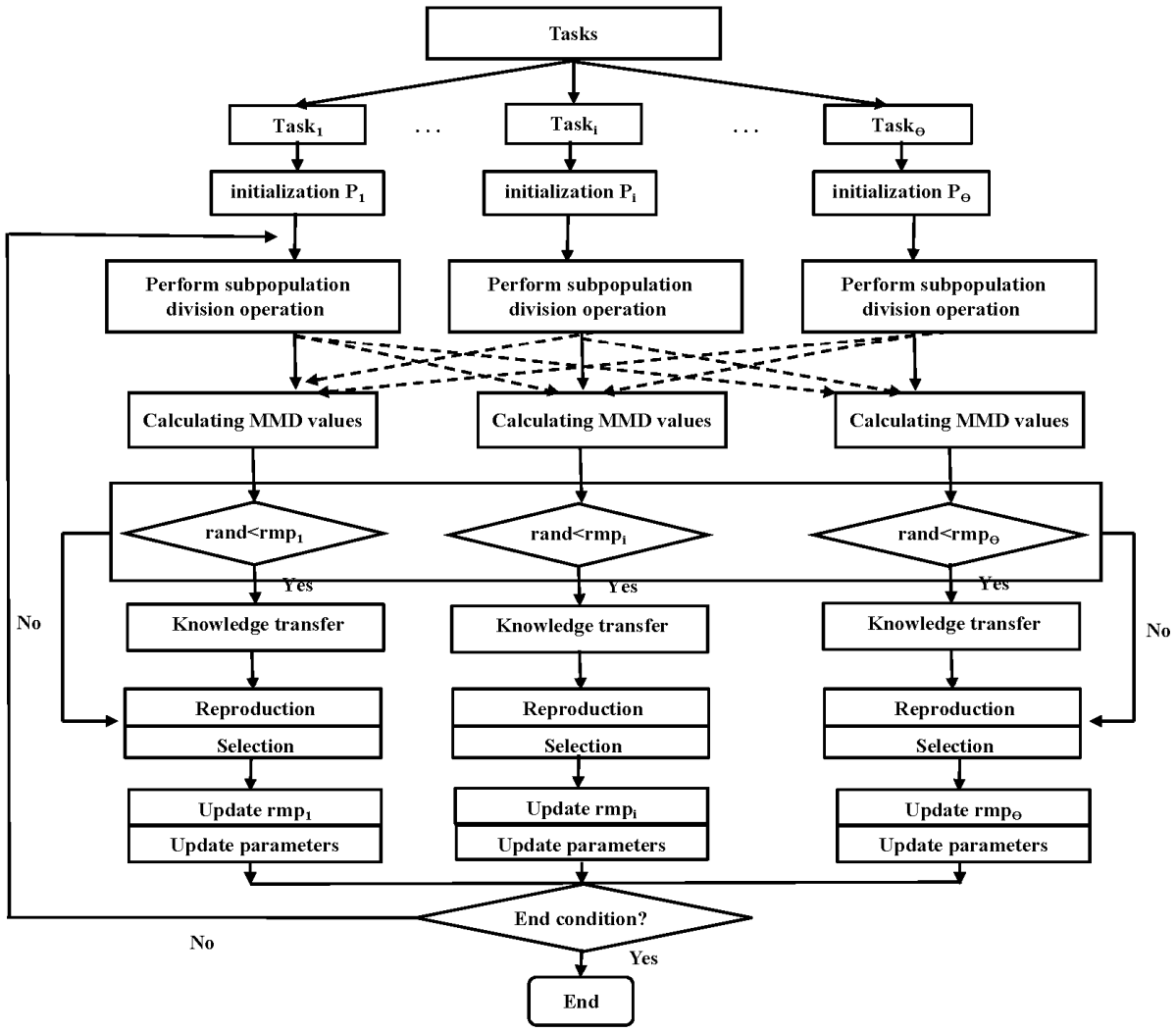


Figure 1. The flowchart of AMTDE-PD.

3.2. Transferred content selection strategy

Transferring the elite solution in a task as a transfer solution between tasks is effective if the co-optimized task pairs are correlated. However, there is no a priori knowledge of inter-task correlation. Therefore, this method of choosing what to transfer will result in a negative transfer if task pairs are irrelevant or of low relevance. An elite solution in one task may be inferior in another. Herein, a transfer content selection strategy based on population distribution information is proposed to select appropriate transferred content between tasks. First, each task population was divided into K sub-populations based on the fitness values of the individuals. Second, the Maximum Mean Discrepancy (MMD) is used to select the most similar clustering from the source task to the clustering of the optimal solution of the target task as the transferred knowledge between tasks. The steps are as follows:

1) Division of subpopulations

Individuals in each task population are first sorted in ascending order of fitness, and then individuals are categorized into advantaged, intermediate, and disadvantaged solutions a based on their fitness values. Then, the task population is categorized into three sub-populations: superior,

intermediate, and inferior. The pseudocode for the subpopulation division process is presented in Algorithm 2.

Algorithm 2 Subpopulation division process

Input: Task populations $P_1, \cdots, P_\Theta$; Number of subpopulation:K; Task population size:pop_t
Output: K subpopulation
1 Individuals in the population were sorted in ascending order of fitness values;
2 Calculate grouping factors $\lambda = pop_t/K$;
3 Calculate the size of each subpopulation based on the grouping factor;
4 Task population is divided into K subpopulations according to the size of the subpopulation;

2) Calculate the MMD value between each cluster in the source task and the cluster where the optimal solution of the target task lies

MMD is a method to determine whether two data distributions are similar by measuring the distance between the two distributions in the reproducing kernel Hilbert space (RKHS) [34]. In [35], MMD is used to select one of the source domains that is most similar to the target domain for inter-task knowledge transfer and has achieved significant success. Inspired by this, we utilize MMD to compute the distance of the clusters and get the similarity between clusters based on the MMD value. When calculating the MMD value, the two high-dimensional distributions are mapped to the RKHS, then the distribution difference is calculated. A smaller MMD value shows that the two distributions are less different, that is, the search preferences of the two clusters are more similar. Suppose the dataset $\mathbf{X} = \left(x_1, \ldots, x_\mu\right)$ and $\mathbf{Y} = \left(y_1, \ldots, y_v\right)$ obey the distributions P and Q, respectively. $\mathcal{F}$ is a class of functions $f: \chi \to \mathrm{R}$ that maps the original search space χ to the set R of real numbers. Let RKHS be denoted as $\mathcal{H}$, $\varphi: \chi \to \mathcal{H}$ denotes a mapping from χ to $\mathcal{H}$. The following is the inner product representation of $f(x)$.

$$f(x) = \langle f, \varphi(x) \rangle_{\mathcal{H}} \tag{8}$$

The MMD values of P and Q in RKHS are calculated as follows:

$$MMD(\mathcal{F}, P, Q) = \sup_{\|f\|_{\mathcal{H}} \le 1} E_P[f(x)] - E_Q[f(y)] = \left\| \frac{1}{\mu} \sum_{i=1}^{\mu} \varphi(x_i) - \frac{1}{v} \sum_{j=1}^{v} \varphi\left(y_j\right) \right\|_2 \tag{9}$$

At each generation, the MMD value between two cluster distributions is calculated as follows:

$$MMD(CK_e^s,\ CK_B^t) = \left\| \frac{1}{N} \sum_{i=1}^{N} \varphi(ck_i^s) - \frac{1}{M} \sum_{j=1}^{M} \varphi\left(ck_j^t\right) \right\|_2, e = 1,2,3 \tag{10}$$

where CK_e^s is a subpopulation in the source task population, CK_B^t is the subpopulation where the optimal solution in the target task lies $ck_i^s \in CK_e^s$, $ck_j^t \in CK_B^t$. The process of calculating the MMD value is shown in Figure 2.

As can be seen from Figure 2, the MMD values between CK_B^t and all subpopulations in the resource task are calculated, and the subpopulation in the resource task corresponding to the minimum MMD value is selected as the migration population.

1) Calculate the index of the cluster CK_e^s corresponding to the smallest MMD value

$$idx = arg\left(min_{1 \le e \le 3}\left(MMD(CK_e^s,\ CK_B^t)\right)\right) \tag{11}$$

2) Transferred content selection

Individuals in the clusters indexed as idx in the source task are used as transferred content in the transfer population TP.

$$TP \leftarrow CK_{idx}^{s} \tag{12}$$

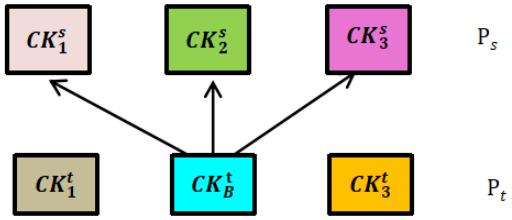


Figure 2. The process of calculating the MMD value between CK_e^s and CK_B^t.

3.3. Inter-task knowledge transfer strategy

Suppose there is a multitasking optimization problem containing Θ tasks. During the evolutionary process, if the Random Mating Probability (RMP) between tasks satisfies the condition $rand < \mathrm{RMP}_\theta$ $(\theta = 1,2,\ldots,\Theta)$, then inter-task evolution is performed. Otherwise, intra-task self-evolution is executed. AMTDE-PD uses the DE algorithm as the task solver and designs the transfer strategy based on the mutation strategy in the DE algorithm. In this paper, we propose a mutation strategy *DE / rand – to – pbest* that considers both exploration and exploitation. In AMTDE-PD, the transfer strategy is designed according to *DE / rand – to – pbest*. Let T_t be the target task, the inter-task transfer strategy is designed based on the following:

$$\mathbf{v}_{\mathrm{i}}^{\mathrm{t}} = \mathbf{x}_{r1}^{s} + F.\left(\mathbf{x}_{pbest}^{t} - \mathbf{x}_{r1}^{s}\right) + F.\left(\mathbf{x}_{r2}^{s} - \mathbf{x}_{r3}^{s}\right) \tag{13}$$

where $\mathbf{v}_i^t$ is the mutant individual corresponding to the current individual $\mathbf{x}_i^t$ in the target task; $\mathbf{x}_{r1}^s$, $\mathbf{x}_{r2}^s$, and $\mathbf{x}_{r3}^s$ are three mutually exclusive individuals randomly selected from the transfer population TP. $\mathbf{x}_{pbest}^t$ is the individual with the top $100 * p$, $p \in (0,1]$ fitness values in the target task. F is the scale factor and is updated with reference to [20].

When the condition $rand < \mathrm{RMP}_\theta$ is not satisfied, AMTDE-PD performs intra-task self-evolution using the mutation strategy *DE / current – to – pbest / 1* and binomial crossover, specifically as follows:

$$\mathbf{v}_{\mathrm{i}}^{\mathrm{t}} = \mathbf{x}_{\mathrm{i}}^{\mathrm{t}} + F \cdot \left(\mathbf{x}_{\mathrm{pbest}}^{\mathrm{t}} - \mathbf{x}_{\mathrm{i}}^{\mathrm{t}}\right) + F \cdot \left(\mathbf{x}_{\mathrm{r1}}^{\mathrm{t}} - \tilde{\mathbf{x}}_{\mathrm{r2}}^{\mathrm{t}}\right) \tag{14}$$

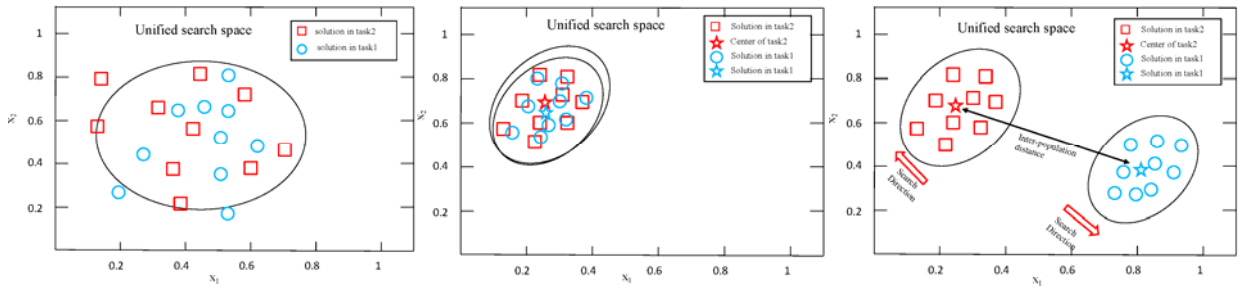
where $\mathbf{x}_{r1}^t$ is randomly selected from the target task. $\tilde{\mathbf{x}}_{\mathrm{r2}}^{\mathrm{t}}$ is randomly selected from the union of the target population and its external archive.

At each generation, AMTDE-PD first judges the condition $rand < RMP$ is satisfied and then decides whether to perform inter-task evolution or intra-task self-evolution.

3.4. Adaptive mating probability strategy

In the EMTO algorithm, multiple tasks corresponding to different search spaces are mapped to a

unified search space. In the early stages of evolution, solutions from different tasks are randomly distributed in a unified search space. As evolution progresses, related tasks move closer together while unrelated tasks move further apart. Figure 3 illustrates this situation.



(a) (b) (c)

Figure 3. (a) Population distribution of tasks in the initial phase (b) Evolutionary trends of similar tasks and (c) Evolutionary trends of dissimilar tasks.

In the evolutionary multitasking optimization algorithm, RMP is used to control the interaction intensity between tasks. In existing studies, RMP is usually adjusted adaptively based on the improvement rate of fitness values during the evolution process. This method can promote the interaction between tasks with similar fitness landscapes and mitigate the impact of negative transfer between tasks with dissimilar fitness landscapes. However, when jointly optimizing two unrelated tasks, this approach can lead to negative transfer. As evolution progresses, the search domains of two unrelated tasks gradually overlap less and eventually become completely non-overlapping, as depicted in Figure 3(c). In this case, knowledge transfer between tasks becomes ineffective and consumes a large number of computational resources.

In view of this, we adjust the RMP between tasks based on the evolutionary trend of populations during the evolution process. If two task populations evolve in the same direction, the intensity of knowledge transfer between tasks is enhanced to accelerate convergence. Otherwise, the intensity of knowledge transfer between tasks is weakened to avoid negative transfer. The direction of change in the population center position can reflect the evolutionary trend of the population, so the evolutionary trend between tasks is estimated by comparing distances between tasks' population centers in current and previous generations. Let the center positions of the target task and source task be denoted as $\overline{X^{\mathrm{t}}}$ and $\overline{X^{s}}$, respectively. In generation g, the distance of the center positions between the target and source tasks is calculated as:

$$Dist_{\mathrm{t,s}}(g) = \left\|\overline{X^{\mathrm{t}}} - \overline{X^{s}}\right\|_2 \tag{15}$$

The calculation method for the center position of the population is as follows:

$$\bar{X} = \frac{1}{N}\sum_{\mathrm{i}=1}^{N} x_i \tag{16}$$

where N is the size of the population and x_i is the individuals in the population.

If $Dist_{\mathrm{t,s}}(g) > Dist_{\mathrm{t,s}}(g-1)$, it means that the populations of the two tasks are moving away

from each other, the evolutionary tendency between the tasks is opposite, and decreasing RMP reduces the interaction strength between the tasks. If $Dist_{\mathrm{t,s}}(g) < Dist_{\mathrm{t,s}}(g-1)$, it means that the populations of the two tasks are approaching each other, and the evolutionary tendency between the tasks is the same, so increase the RMP to improve the interaction strength between the tasks. However, when RMP is greater than or equal to 1, the evolution between tasks will occupy all computational resources, and intra-task self-evolution will not be able to proceed, which is detrimental to the evolution of the population. Therefore, when [RMP]_t is greater than or equal to 1, it should be set back to 0.5 to ensure a balance between inter-task evolution and intra-task self-evolution. The pseudo-code of the update of RMP is shown in Algorithm 3.

Algorithm 3 Update of RMP

Input: Inter-task mating probability for generation $RMP_t(g)$; Distance between population center positions $Dist_{\mathrm{t,s}}$; Control parameters q.

Output: $RMP_t(g+1)$

if $Dist_{\mathrm{t,s}}(g) < Dist_{\mathrm{t,s}}(g-1)$
  $RMP_t(g+1) = RMP_t(g)/q;$
  if $RMP_t(g+1) \geq 1$
    $RMP_t(g+1) = 0.5;$
  end if
  else
    $RMP_t(g+1) = RMP_t(g) * q;$
end if

4. Experiment and analysis

4.1. Test problems and compared algorithms

To evaluate the search efficiency of AMTDE-PD, we compare it with other advanced EMTO algorithms on two single-object multitasking test suites. The first test suite CEC2019-SOMTP [36] is the single-objective multitasking standard test set, which comprises nine single-objective MTO problems, each consisting of two minimization tasks. According to the intersection of the global optimum of the optimization task these problems can be classified into three categories: Complete Intersection (CI), Partial Intersection (PI), and No intersection (NI). In addition, Spearman's correlation coefficient (SRCC) was used to assess the similarity between the fitness landscapes of two tasks. According to the results of SRCC, these problems can be further characterized into three categories: High Similarity (HS), Medium Similarity (MS), and Low Similarity (LS). The second test suite is the single-objective multi-task complex test set WCCI2020-SOCTP[1] contains ten complex single-objective MFO problems. Different from the CEC2019-SOMTP, the task functions composing the complex test set are obtained by rotating and shifting four hybrid functions and seven multi-mode functions in the CEC2014 test suite to varying degrees, which makes them more difficult to solve.

To assess the effectiveness of the algorithm, we compare AMTDE-PD with nine advanced EMTO algorithms on the CEC2019-SOMTP, including four classical algorithms using EA as solver: MFEA [2], GMFEA [37], MTGA [38], SBO [7], and LDA-MFEA [4]; three high-performance algorithms using

[1]http://www.bdsc.site/websites/MTO/index.html

DE as solvers: MFDE [39], MFMP [9], and MPEFMTO [40]; and one algorithm that uses both DE and EA as task solvers: EMT-EAE [8]. Another reason for choosing these algorithms is that these algorithms use elite-based solutions, and randomized solutions, respectively, as transferred content. To further validate the superiority of the AMTDE-PD algorithm in solving complex optimization problems, the AMTDE-PD algorithm is compared with MFEA, MFDE, MFMP, and MPEFMTO on the CEC2019-SOMTP test suite.

4.2. Parameter settings and performance metrics

To ensure an unbiased comparison, the maximum function evaluation of each algorithm was set to 100,000, and each algorithm was run independently 20 times on each test problem. All other parameters in the comparison algorithms obey their original paper. The parameters of AMTDE-PD on the two test suites are set as follows: the population size of each task N = 100, the number of clusters K = 3, the RMP update threshold $\delta = 0.5$, the control parameter $q = 0.9$, the initial values of both F and CR are set to 0.5.

Synergy performance metrics proposed in the literature [38] are used in this experiment to verify the comprehensive performance of an algorithm on multiple tasks in a multitasking environment. Let the test case contain Θ minimization tasks. The optimal solution obtained by the *l*th execution of an algorithm on the task $T_\theta, \theta \in (1,2,\cdots,\Theta)$ is denoted as $f_{r,\theta}^i$. Let each algorithm be executed *R* times, then the performance metric of the *i*th algorithm is defined as follows:

$$Score_i = \sum_{\theta=1}^{\Theta} \sum_{r=1}^{R} \frac{f_{\theta,r}^i - \mu_\theta}{\sigma_\theta} \tag{17}$$

where μ_θ and σ_θ are the mean and variance of $f_{r,\theta}^i$ obtained by all algorithms running *R* times on task T_θ, respectively.

For the minimization problem, the smaller *Score* value of the algorithm indicates the better overall performance of the corresponding EMTO algorithm on the multitasking problem.

4.3. Results on CEC2019-SOMTP

The AMTDE-PD algorithm and the nine comparison algorithms are executed independently 20 times on the CEC2019-SOMTP, and the mean and standard deviation of the optimal function values achieved from the solution are shown in Table 1. The results of the Wilcoxon rank sum test at the significance level α = 0.05 are given in the last row of Table 1, and the symbols "+", "-" and "≈" indicate that the comparison algorithms' performance is better than, worse than, and approximate to the AMTDE-PD algorithm. To further minimize statistical comparison errors, Friedman's test is used to examine the difference between AMTDE-PD and the other comparison algorithms. The average rank is recorded as the comparison results, and a smaller one indicates better performance. The results of Friedman's test at the 0.05 significance level are given in Table 2, and the best results are shown in bold. Furthermore, this section uses the algorithm's performance metric *Score* as an assessment of the algorithm's performance, which reflects the algorithm's comprehensive performance on a problem. The values of the performance metric Score for the AMTDE-PD algorithm and the other compared algorithms are given in Table 3.

Table 1. Experimental results of AMTDE-PD algorithm and other nine comparison algorithms on CEC2019-SOMTP test suite.

Problem	Task	MFEA	MFDE	GMFEA	LDA-MFEA	SBO	MTGA	EMT-EAE	MPEF	MPEFMTO	AMTDE-PD
CI+HS	T_1	3.74E-01-	8.94E-04-	1.05E+00-	3.38E-01-	9.35E-01-	1.24E-02-	7.05E-01-	1.93E-08-	2.58E-09-	**4.7968e-12**
		(6.64E-02)	(2.70E-03)	(1.59E-02)	(7.60E-02)	(5.64E-02)	(8.83E-03)	(8.22E-02)	(1.55E-08)	(2.54E-09)	**(7.64E-12)**
	T_2	1.98E+02-	1.89E+00-	3.68E+02-	1.53E+02-	3.09E+02-	5.06E+01-	3.72E+02-	4.93E-05-	6.82E-06-	**7.00E-09**
		(5.16E+01)	(5.65E+00)	(2.57E+01)	(3.19E+01)	(2.94E+01)	(1.64E+01)	(6.23E+01)	(4.25E-05)	(6.54E-06)	**(1.03E-08)**
CI+MS	T_1	4.72E+00-	4.45E-02-	7.15E+00-	3.15E+00-	4.72E+00-	1.20E+00-	3.97E+00-	4.40E-06-	2.72E-07-	**8.55E-09**
		(5.49E-01)	(1.96E-01)	(6.92E-01)	(4.27E-01)	(2.90E-01)	(8.22E-01)	(2.64E-01)	(5.17E-06)	(1.33E-07)	**(1.03E-08)**
	T_2	2.12E+02-	1.50E-01-	4.69E+02-	1.69E+02-	3.24E+02-	5.18E+01-	3.48E+02-	3.12E-08-	5.75E-11-	**1.91E-14**
		(6.29E+01)	(6.67E-01)	(6.56E+01)	(3.10E+01)	(3.90E+01)	(1.84E+01)	(3.78E+01)	(6.43E-08)	(5.31E-11)	**(5.50E-14)**
CI+LS	T_1	**2.02E+01**+	2.12E+01-	**2.02E+01**+	2.10E+01+	2.11E+01≈	**2.02E+01**+	2.12E+01-	2.12E+01-	2.12E+01-	2.11E+01
		(6.46E-02)	(3.10E-02)	(4.03E-02)	(2.07E-01)	(2.43E-01)	(3.51E-01)	(3.79E-02)	(7.51E-02)	(7.26E-02)	(7.46E-02)
	T_2	**3.71E+03+**	1.11E+04-	6.62E+03-	4.28E+03+	4.56E+03+	3.41E+03+	4.49E+03+	5.96E+03-	5.82E+03-	5.60E+03
		(4.93E+02)	(1.62E+03)	(6.30E+02)	(5.11E+02)	(7.23E+02)	(5.99E+02)	(5.60E+02)	(4.11E+02)	(4.37E+02)	(4.27E+02)
PI+HS	T_1	5.81E+02-	8.27E+01-	9.09E+02-	3.17E+02-	4.26E+02-	**5.40E+01+**	3.93E+02-	2.65E+02+	2.60E+02+	2.66E+02
		(1.17E+02)	(1.67E+01)	(1.33E+02)	(4.88E+01)	(6.73E+01)	**(1.91E+01)**	(5.11E+01)	(1.91E+01)	(1.83E+01)	(2.12E+01)
	T_2	8.82E+00-	2.30E-05-	2.31E+02-	1.19E+01-	1.03E+02-	1.17E-08-	5.63E+01-	1.67E-09-	4.64E-10-	**1.90E-13**
		(2.06E+00)	(2.14E-05)	(5.15E+01)	(2.63E+00)	(2.32E+01)	(3.24E-08)	(1.48E+01)	(2.31E-09)	(2.98E-10)	**(1.34E-13)**
PI+MS	T_1	3.53E+00-	1.03E-01-	7.99E+00-	2.84E+00-	5.04E+00-	1.51E-01-	3.96E+00-	6.79E-05-	2.02E-05-	**1.36E-07**
		(5.04E-01)	(4.50E-01)	(8.12E-01)	(4.63E-01)	(3.06E-01)	(4.67E-01)	(4.23E-01)	(5.94E-05)	(1.67E-05)	**(2.19E-07)**
	T_2	6.38E+02-	7.22E+01-	1.15E+05-	5.31E+02-	1.42E+04-	2.34E+02-	4.57E+03-	8.17E+01-	7.70E+01-	**6.47E+01**
		(1.96E+02)	(2.26E+01)	(4.88E+04)	(1.81E+02)	(4.53E+03)	(4.00E+02)	(2.64E+03)	(9.07E+00)	(1.81E+01)	**(6.47E+01)**
PI+LS	T_1	2.00E+01-	7.80E-01-	1.84E+01-	3.18E+00-	5.43E+00	1.73E+00-	3.94E+00-	1.51E-05-	1.32E-06-	**3.82E-07**
		(1.15E-01)	(7.39E-01)	(4.78E+00)	(4.19E-01)	(9.38E-01)	(6.82E-01)	(4.98E-01)	(2.78E-05)	(3.66E-06)	**(5.23E-07)**
	T_2	2.11E+01-	1.12E-01-	2.01E+01-	3.15E+00-	5.75E+00-	2.64E+00-	7.39E+00-	7.32E-04-	2.58E-04-	**1.59E-04**
		(3.29E+00)	(2.73E-01)	(5.58E+00)	(8.29E-01)	(1.35E+00)	(2.37E+00)	(3.45E+00)	(7.20E-04)	(3.22E-04)	**(1.15E-04)**
NI+HS	T_1	7.49E+02-	9.53E+01-	5.15E+04-	8.04E+02-	1.40E+04-	1.12E+02-	6.18E+03-	4.34E+01-	4.28E+01-	**4.22E+01**
		(2.68E+02)	(6.81E+01)	(1.99E+04)	(3.93E+02)	(5.39E+03)	(1.09E+02)	(3.61E+03)	(1.09E+00)	(5.16E-01)	**(8.87E-01)**
	T_2	2.60E+02-	3.03E+01-	5.00E+02-	2.08E+02-	3.66E+02-	5.96E+01-	3.70E+02-	5.62E-04-	1.51E-06-	**5.31E-07**
		(4.39E+01)	(1.34E+01)	(6.73E+01)	(6.78E+01)	(3.09E+01)	(1.94E+01)	(6.80E+01)	(1.16E-03)	(4.24E-06)	**(1.04E-06)**
NI+MS	T_1	4.09E-01-	2.65E-03-	1.05E+00-	3.92E-01-	9.24E-01-	6.27E-03-	8.06E-01-	3.14E-07-	3.17E-08-	**5.25E-09**
		(6.63E-02)	(4.18E-03)	(1.81E-02)	(6.61E-02)	(5.20E-02)	(9.74E-03)	(8.76E-02)	(1.73E-07)	(5.41E-08)	**(5.25E-09)**
	T_2	2.58E+01-	3.21E+00-	2.88E+00-	1.41E+01-	2.33E+01-	8.41E+00-	2.34E+01-	1.33E+00-	**9.63E-01**+	1.12E+00
		(3.05E+00)	(1.65E+00)	(2.56E+00)	(2.14E+00)	(5.19E+00)	(6.83E+00)	(5.02E+00)	(7.63E-01)	(6.23E-01)	(5.42E-01)
NI+LS	T_1	6.06E+02-	**1.01E+02+**	8.71E+02-	3.20E+02+	4.27E+02-	1.69E+02+	4.38E+02-	2.75E+02-	2.65E+02-	2.59E+02
		(9.99E+01)	**(2.38E+01)**	(1.81E+02)	(4.73E+01)	(5.47E+01)	(6.90E+01)	(8.68E+01)	(1.99E+01)	(1.65E+01)	(1.96E+01)
	T_2	3.62E+03-	4.07E+03-	6.65E+03-	4.14E+03-	4.31E+03-	2.89E+03-	4.52E+03-	5.98E+03-	6.07E+03-	**1.99E+03**
		(4.60E+02)	(6.99E+02)	(6.71E+02)	(5.23E+02)	(5.90E+02)	(6.43E+02)	(5.55E+02)	(6.43E+02)	(3.37E+02)	**(5.38E+02)**
+/≈/-		2/0/16	1/0/17	1/0/17	3/0/15	1/1/16	4/0/14	1/0/17	1/0/17	2/0/16	

In Table 1, AMTDE-PD significantly outperforms the other nine compared algorithms in terms of average objective values on 13 out of the 18 test tasks of the CEC2019-SOMTP. According to the numbers of "+/≈/-" in Table1, we can see that AMTDE-PD better than MFEA, MFDE, GMFEA, LDA-MFEA, SBO, MTGA, EMT-EAE, MPEF, and MPEFMTO on the CEC2019-SOMTP test suite is 16, 17, 17, 15, 16, 14, 17, 17, and 16, respectively. It can be concluded that the AMTDE-PD algorithm outperforms other algorithms on the CEC2019-SOMTP benchmark based on the accuracy of the solution and the stability of the algorithm.

From Table 2, it can be seen that AMTDE-PD obtains the best ranks on both Task T1 and Task T2 in all test problems compared to MFEA, MFDE, GMFEA, LDA-MFEA, SBO, MTGA, EMT-EAE, MFMP, and MPEFMTO. That is to say, AMTDE-PD has the best overall performance on Task T1 and Task T2 for all the test problems. Moreover, the P-value values of AMTDE-PD on Task T1 and Task T2 are both 0, which indicates that the performance of the AMTDE-PD algorithm is significantly different from the other compared algorithms, and the AMTDE-PD algorithm outperforms the other compared algorithms in a statistically significant way.

Table 2. Friedman test results of AMTDE-PD and nine comparison algorithms on CEC2019-SOMTP.

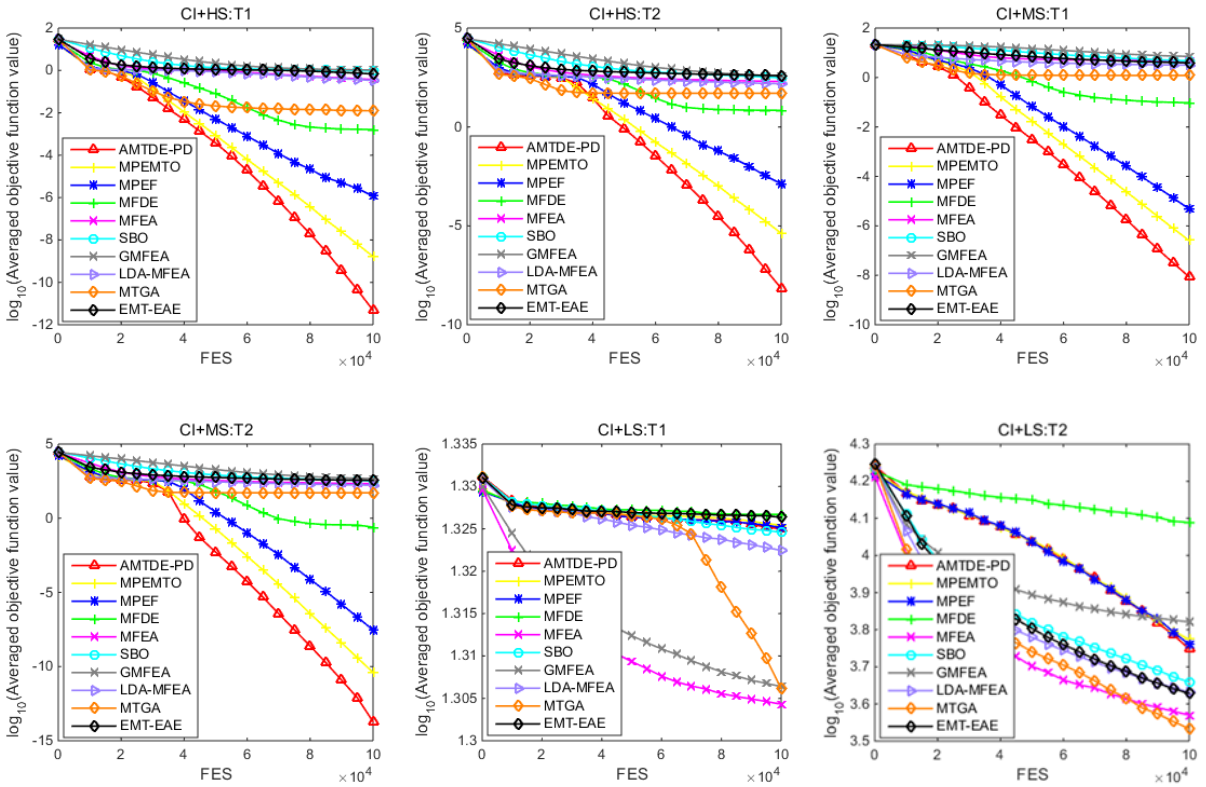
	Task1		Task2	
	Algorithm	Average rank	Algorithm	Average rank
1	AMTDE-PD	2.17	AMTDE-PD	1.67
2	MPEFMTO	3.06	MPEFMTO	3.33
3	MTGA	3.89	MTGA	4.22
4	MPEF	3.94	MPEF	4.22
5	MFDE	3.94	MFDE	4.67
6	LDA-MFEA	5.89	LDA-MFEA	5.78
7	MFEA	7.28	MFEA	6.56
8	EMT-EAE	7.72	EMT-EAE	8.00
9	SBO	8.11	SBO	7.56
10	GMFEA	9.00	GMFEA	9.00
statistic	51.451		46.406	
P-value	0.000		0.000	

Table 3 shows that the AMTDE-PD achieved the best Score values on six of the nine test problems. For the complete intersection (CI) problem AMTDE-PD took the best performance scores on medium to high similarity problems and performed worse than algorithms (MFEA, GMFEA, LDA-MFEA, SBO, MTGA, EMT-EAE) using chromosome crossover for gene transfer on low similarity problems, but better than algorithms (MFDE, MFMP, MPEFMTO) using the mutation mechanism of DE for inter-task knowledge transfer. For the PI problem, AMTDE-PD obtains better Score values than the other compared algorithms on the low and medium similarity problems (PI+MS, PI+LS), and slightly worse than MTGA and MPEFMTO on the high similarity problem (PI+HS) problem. For the NI problem, AMTDE-PD obtains the best Score values on both the high similarity and the low similarity problems (NI+HS, NI+LS) achieves the best Score values and only slightly underperforms the MPEFMTO algorithm on the medium similarity problem (NI+MS). In conclusion, the proposed

algorithm obtains good Score values on nine benchmark problems, and the overall performance outperforms other algorithms. This is because when the global optimum of a co-optimization task varies greatly, using either the elite solution or a randomly selected solution as the migrated knowledge between tasks may be less effective. On the contrary, AMTDE-PD uses a transfer content selection strategy based on population information, which is useful when the global optimums of the tasks differ greatly. Therefore, AMTDE-PD achieves better results on most PI and NI problems.

Table 3. Score values of AMTDE-PD and nine comparison algorithms on CEC2019-SOMTP.

Problem	AMTDE-PD	MFEA	MFDE	GMFEA	LDA-MFEA	SBO	MTGA	EMT-EAE	MFMP	MPEFMTO
CI+HS	**-17.242573**	4.096979	-17.101201	30.988655	0.392154	24.409735	-13.758544	22.699937	-17.242569	-17.242572
CI+MS	**-18.444439**	11.652982	-18.268405	35.157126	3.248721	17.936475	-10.789895	16.396293	-18.444422	-18.444438
CI+LS	5.920443	**-23.359814**	31.359196	-8.055326	-2.654138	0.696335	-22.288089	2.720678	8.073585	7.587131
PI+HS	-9.000528	4.797912	-16.276617	46.481302	-5.309010	11.321329	**-17.440352**	3.606440	-8.975350	-9.205126
PI+MS	**-12.279223**	0.540779	-11.871226	48.059094	-1.920223	10.299737	-11.619883	3.340061	-12.274124	-12.274992
PI+LS	**-14.112506**	37.378323	-12.989220	33.089194	-6.151046	-0.104017	-8.738217	-0.148317	-14.111795	-14.112398
NI+HS	**-14.054337**	0.197694	-12.396925	43.392954	-2.450153	14.897147	-10.800141	9.320940	-14.053334	-14.053844
NI+MS	-18.423680	11.916704	-16.592061	29.504499	1.720064	21.714273	-12.197161	19.171619	-18.256331	**-18.557927**
NI+LS	**-20.483169**	5.271598	-14.224810	35.217437	-4.210461	1.822962	-18.571801	3.434995	5.706511	6.036738



Continued on next page

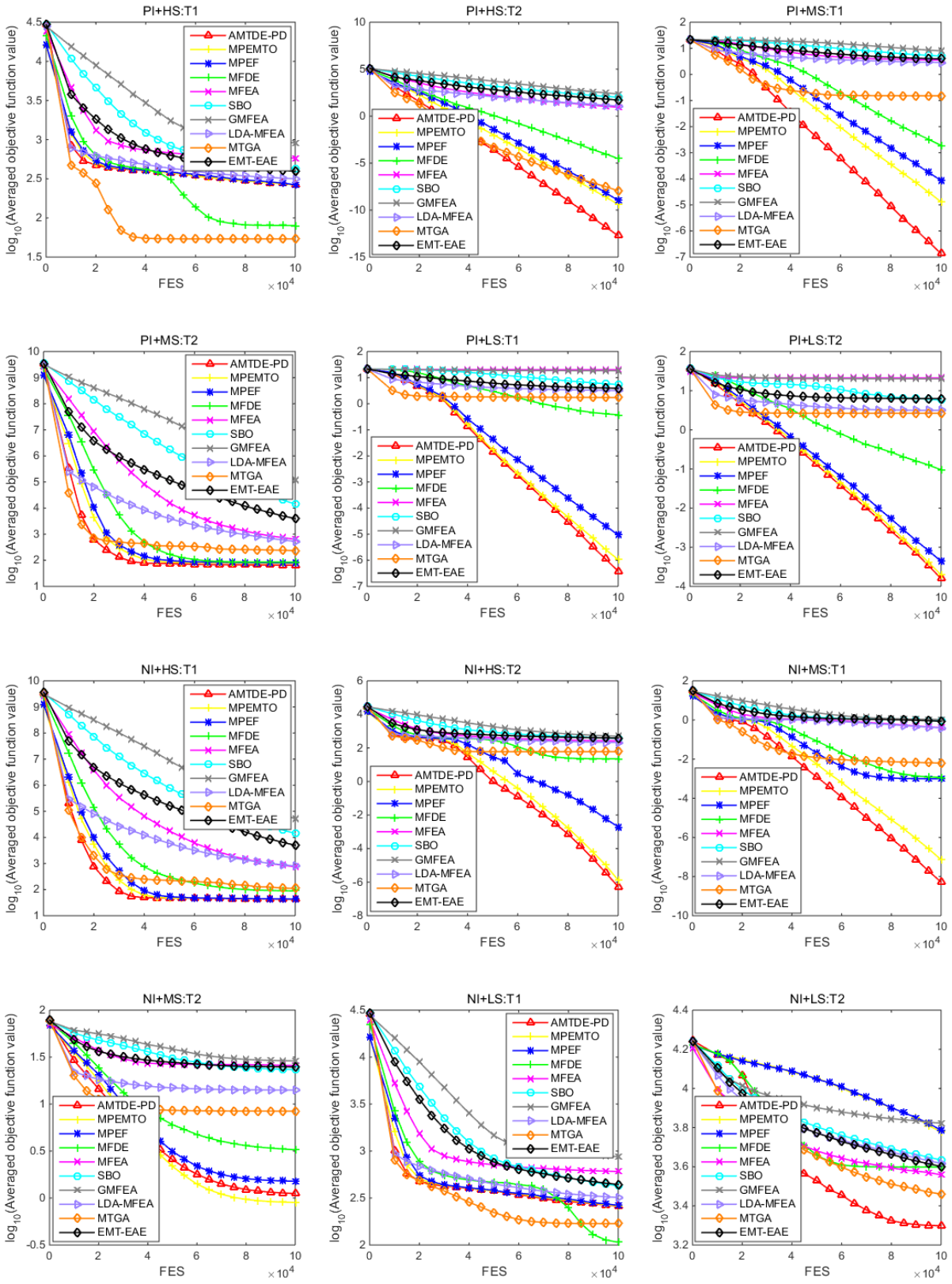
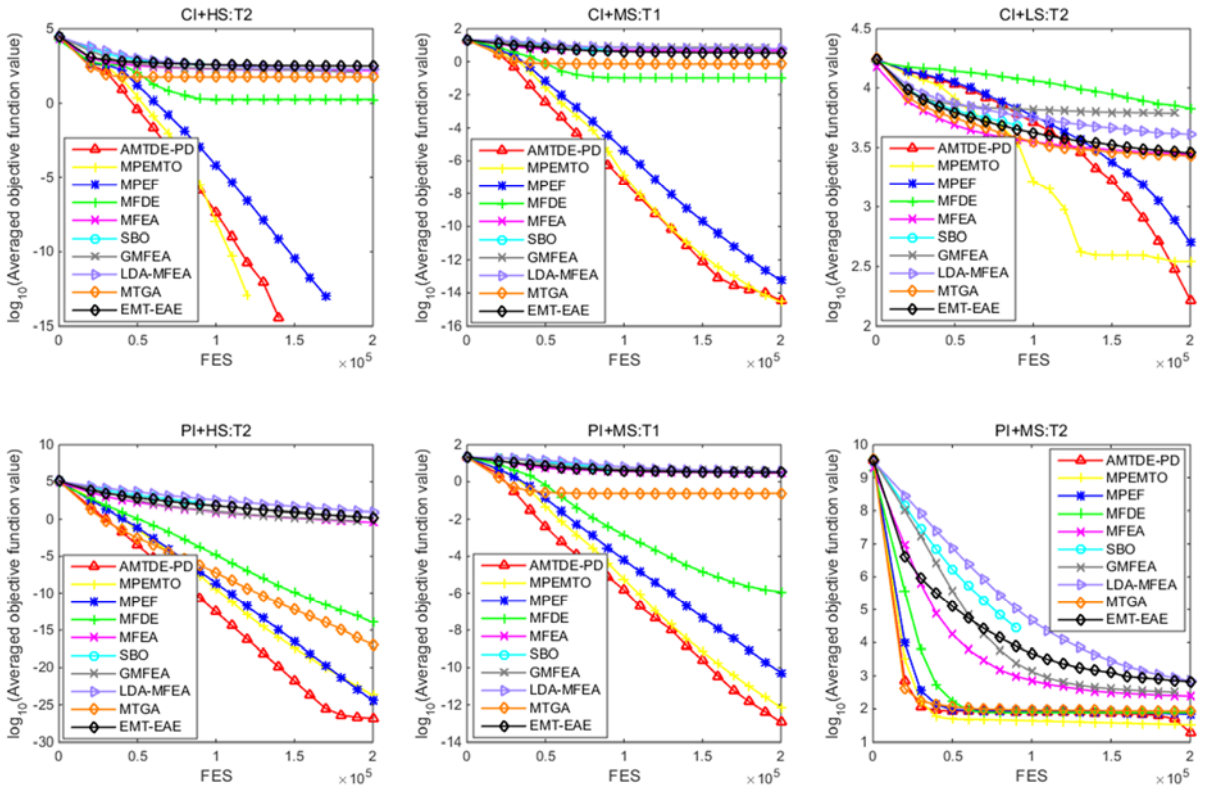


Figure 4. Average convergence curves obtained by AMTDE-PD and compared algorithms On CEC2019-SOMTP.

To analyze the convergence behavior of AMTDE-PD and the comparison algorithms more intuitively, their convergence curves are given in Figure 4. In Figure 4, the convergence speed of AMTDE-PD is faster than that of other algorithms on most problems in CEC2019-SOMTP. However, it is slower than the algorithm for gene transfer with chromosome crossover on the CI+LS problem and slightly slower than MFDE on the NI+LS task 1 (Rastrigin function). Rastrigin function has multiple local maxima and minima, which tends to make the algorithm fall into a local optimum. MFDE generates offspring by randomly combining parents, which can obtain widely distributed solutions and escape from the local optimum. Therefore, MFDE can get better performance on the Rastrigin function.

In summary, the AMTDE-PD algorithm has better solution accuracy and faster convergence speed than other algorithms, can solve the single-objective MFO problem effectively and demonstrates a more competitive and comprehensive performance than other EMTO algorithms.

In order to evaluate the convergence of the algorithms more fairly, the maximum number of function evaluations of all the algorithms was increased to 200,000 times for the experiments, and the convergence curves of AMTDE-PD and its comparison algorithms are shown in Figure 5. From Figure 5, we can see that with sufficient function evaluation times, AMDE has competitiveness in terms of convergence speed.



Continued on next page

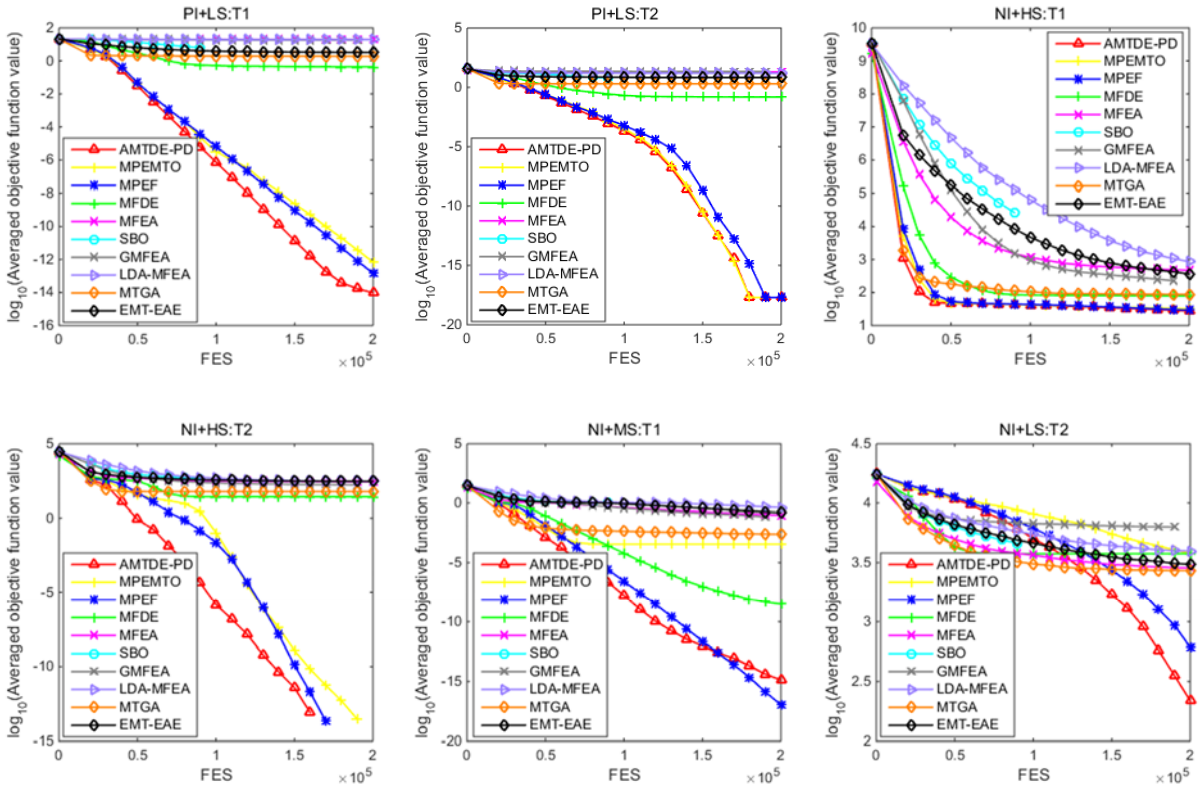


Figure 5. Average convergence curves obtained by AMTDE-PD and compared algorithms On CEC2019-SOMTP for maximum function evaluation as 200,000.

4.4. Results on complex single-objective multitasking optimization problems

Compare AMTDE-PD with two classical evolutionary multitasking optimization algorithms, MFEA and MFDE, and two recent high-performance evolutionary multitasking optimization algorithms, MPEF, and MPEFMTO, on WCCI2020-SOCTP. The reason for choosing these comparison algorithms is that these algorithms use different transfer content selection strategies. Among them, MFEA and MFDE use the randomized solutions in the task as the transfer content. MPEF uses the difference vector of the randomized solutions in the task and the elite solutions as the transfer content. The transfer content in MPEFMTO is the dominant solution, the inferior solution and the difference vector of the randomized solutions in the task. The average objective function values obtained from 20 independent runs of each algorithm on each task and the standard deviation were recorded as results. The results are shown in Table 4, and the results of the Wilcoxon rank sum test with a significance level $\alpha = 0.05$ are given in the last row. The synergy performance metric Score is also utilized to verify the comprehensive performance of each algorithm on each problem containing two complex tasks. The values of the performance metric Score for the AMTDE-PD, MFEA, MPDE, MFMP, and MPEFMTO are given in Table 5.

As can be seen in Table 4, AMTDE-PD outperforms MFEA, MFDE, MPEF, and MPEFMTO in the CCI2020-SOCTP test suite on 17, 18, and 12 tasks, respectively. However, MFEA, MFDE, MPEF, and MPEFMTO approximate AMTDE-PD on 0, 2, 5, and 6 tasks, respectively. In addition, the synergy performance metric scores of the algorithms in Table 5 show that the overall performance of AMTDE-PD is superior to that of MFEA, MFDE, MPEF and MPEFMTO on 8, 10, 6, and 6 problems,

respectively. The above statistical results show that AMTDE-PD can effectively deal with complex multitasking optimization problems.

Table 4. Experimental results of AMTDE-PD algorithm and other nine comparison algorithms on WCCI2020-SOCTP test suite.

Problem	Task	MFEA	MFDE	MPEF	MPEFMTO	AMTDE-PD
P1	$Task_1$	6.4841E+02	6.0408E+02	6.0084E+02	6.0076E+02	**6.0059E+02**
	$Task_2$	(4.8164e+00)-6.4832E+02	(2.5657e+00)-6.0442E+02	(5.6434E-01)-6.0101E+02	(4.4438E-01)-6.0094E+02	**(3.8116e-01)6.0067E+02**
P2	$Task_1$	(4.7256e+00)-7.0107E+02	(2.5177e+00)-**7.0000E+02**	(5.6434E-01)-**7.0000E+02**	(5.2927E-01)-**7.0000E+02**	**(3.1623e-01)7.0000E+02**
	$Task_2$	(2.5362e-02)-7.0107E+02	(7.0583e-05)≈**7.0000E+02**	(7.0972E-06)≈**7.0000E+02**	(1.6550E-03)≈**7.0000E+02**	(1.6538e-03)**7.0000E+02**
P3	$Task_1$	(1.4481e-02)-3.0247E+06	(3.8944e-03)≈6.3359E+06	(1.7538E-06)≈**8.9800E+03**	(2.3784E-03)≈1.3640E+04	(2.2084e-03)2.1596E+04
	$Task_2$	(1.6196e+06)-2.9191E+06	(2.0626e+06)-6.4765E+06	(4.3220E+03)+9.4501E+03	(1.6415E+04)+1.0963E+04	(3.9472e+04)**9.1345E+03**
P4	$Task_1$	(1.2221e+06)-1.3006E+03	(2.9946e+06)-1.3006E+03	(3.5152E+03)-**1.3004E+03**	(6.9444E+03)-**1.3004E+03**	**(2.8387e+03)1.3004E+03**
	$Task_2$	(8.7430e-02)-1.3005E+03	(8.0022e-02)-1.3005E+03	(4.4541E-02)≈**1.3003E+03**	(3.6728E-02)≈**1.3003E+03**	(4.8408e-02)**1.3003E+03**
P5	$Task_1$	(6.5949e-02)-1.5611E+03	(7.0733e-02)-1.5330E+03	(5.1003E-02)≈**1.5222E+03**	(4.2129E-02)≈1.5225E+03	(4.5993e-02)1.5226E+03
	$Task_2$	(1.3037e+01)-1.5523E+03	(2.0512e+00)-1.5339E+03	(1.8118E+00)+1.5237E+03	(1.8091E+00)+**1.5235E+03**	(1.8020E+00)1.5237E+03
P6	$Task_1$	(1.2078e+01)-1.8249E+06	(1.7193e+00)-2.5073E+06	(1.8002E+00)≈1.6597E+04	**(1.4926E+00)+1.2602E+04**	(1.6514E+00)3.1416E+04
	$Task_2$	(7.9999e+05)-1.8805E+06	(1.1612e+06)-1.9593E+06	(2.7983E+04)+1.0719E+04	**(7.2662E+03)+**9.6431E+03	(8.6112E+04)**8.6457E+03**
P7	$Task_1$	(8.9531e+05)-3.3196E+03	(7.6951e+05)-3.8522E+03	(8.4671E+03)-2.5376E+03	(4.8180E+03)-2.5312E+03	**(1.5279E+03)2.5238E+03**
	$Task_2$	(3.5229e+02)-3.3172E+03	(1.7922e+02)-3.9439E+03	(1.1344E+02)-2.6351E+03	(1.0284E+02)-**2.6156E+03**	(1.1907E+02)2.6235e+03
P8	$Task_1$	(3.1922e+02)-**5.2024E+02**	(1.9536e+02)-5.2120E+02	(1.1284E+02)-5.2119E+02	**(1.5219E+02)+**5.2118E+02	(1.1527e+02)5.2118E+02
	$Task_2$	(7.6601e-02)+**5.2029E+02**	(3.9190e-02)-5.2121E+02	(5.1409E-02)-5.2121E+02	(4.4255E-02)≈5.2119E+02	(6.1730E-02)5.2118E+02
P9	$Task_1$	(7.9562e-02)+**8.2322E+03**	(3.5954e-02)-1.4742E+04	(3.0681E-02)-1.1287E+04	(4.1841E-02)-1.1396E+04	(4.0389E-02)1.1261E+04
	$Task_2$	(1.2798e+03)+1.6217E+03	(4.0996e+02)-1.6227E+03	(4.1119E+02)-1.6215E+03	(5.0014E+02)-**1.6214E+03**	(4.6353e+02)**1.6214e+03**
P10	$Task_1$	(5.4480e-01)-3.2613E+04	(1.8758e-01)-2.7934E+04	(3.5049E-01)-2.4966E+03	(3.8833E-01)≈2.2924E+03	**(**4.2650e-01)**2.2409E+03**
	$Task_2$	(1.5889e+04)-3.0951E+06	(1.0828e+04)-2.3497E+06	(1.2714E+03)-5.0067E+04	(2.5205E+02)-**1.7257E+04**	**(4.0205E+01)**1.7769E+04
+/≈/-		(2.7513e+06)-3/0/17	(8.6756e+05)-0/2/18	(1.2795E+05)-3/5/12	(1.4961E+04)+6/6/8	(407316E+04)

Table 5. The Score values of performance metrics for AMTDE-PD, MFEA, MPDE, MFMP and MPEFMTO.

Problem	AMTDE-PD	MFEA	MFDE	MPEF	MPEFMTO
P1	**-9.91003**	35.63654	-6.45095	-9.61851	-9.65704
P2	**-8.94667**	35.77683	-8.92343	-8.9653	-8.94144
P3	**-12.9939**	9.08291	29.5785	-12.95587	-12.93474
P4	-11.70563	16.20448	22.44774	-13.16699	**-13.7796**
P5	-11.99945	32.42007	4.20295	-12.21723	**-12.40634**
P6	-13.7214	17.81398	23.55646	-13.7802	**-13.86884**
P7	**-13.33472**	10.20006	29.38105	-13.03279	-13.2136
P8	8.31523	**-35.60554**	9.60732	9.2293	8.4537
P9	-4.06042	**-13.22761**	28.56233	-5.82066	-5.45363
P10	**-13.6617**	21.68881	19.39499193	-13.17343	-13.65862

4.5. Effects of the transfer strategy

To analyze the effectiveness of the inter-task transfer strategy used in AMTDE-PD, we define three variants of AMTDE-PD. One is No-AMTDE-PD with no inter-task knowledge transfer, another is Rand-AMTDE-PD with randomized solutions transfer between tasks, and another is Elist-AMTDE-PD with elite solutions transfer between tasks. AMTDE-PD is compared with these three variants on CEC2019-SOMTP, with each variant running 20 independent runs. The comparison results are shown in Table 6. In Table 6, it is clear that Rand-AMTDE-PD, Elist-AMTDE-PD, and AMTDE-PD outperform No-AMTDE-PD on most of the tasks, which indicates that inter-task knowledge transfer can facilitate co-optimization of tasks. Among them, Elist-AMTDE-PD performs better than No-AMTDE-PD on high and medium similarity problems and worse than No-AMTDE-PD on some low similarity problems. This is because if the tasks have high similarity, transferring the elite solutions between the tasks can effectively improve the co-optimization of the tasks. On the contrary, if the tasks are not similar or have low similarity, negative transfer will occur, thus affecting the performance of the algorithm. Rand-AMTDE-PD outperforms No-AMTDE-PD on 13 tasks and underperforms No-AMTDE-PD on 5. The above phenomenon is attributed to the random selection of what to transfer between tasks, which maintains population diversity to some extent due to the random nature of the transfer, but is not conducive to the full utilization of useful information between tasks. AMTDE-PD outperforms No-AMTDE-PD, Elist-AMTDE-PD, and Rand-AMTDE-PD on 15, 16, and 14 tasks out of the 18 tasks, respectively. The results show that the transfer strategy based on population distribution information proposed by us can effectively promote collaborative optimization but cannot completely avoid negative transfer between tasks, especially for problems with no intersection and low similarity.

Table 6. Comparison of solution accuracy of AMTDE-PD with Rand-AMTDE-PD, Elist-AMTDE-PD and No-AMTDE-PD.

Problem	Task	No-AMTDE-PD	Rand-AMTDE-PD	Elist-AMTDE-PD	AMTDE-PD
CI+HS	T_1	1.43E-09	6.55E-09	3.34E-10	**4.80E-12**
	T_2	2.50E+02	1.99E-05	5.73E-06	**7.00E-09**
CI+MS	T_1	3.27E-07	1.57E-08	5.08E-07	**8.55E-09**
	T_2	2.50E+02	1.41E-13	2.90E-10	**1.91E-14**
CI+LS	T_1	**2.11E+01**	2.12E+01	2.12E+01	**2.11E+01**
	T_2	**4.94E+03**	5.61E+03	6.31E+03	5.60E+03
PI+HS	T_1	2.68E+02	2.67E+02	**2.48E+02**	2.66E+02
	T_2	2.04E-09	1.92E-13	1.21E-13	**1.90E-13**
PI+MS	T_1	2.90E-07	**8.17E-08**	2.09E-01	1.36E-07
	T_2	4.87E+01	4.90E+01	**4.86E+01**	6.47E+01
PI+LS	T_1	5.36E-07	2.85E-07	1.30E-06	**3.82E-07**
	T_2	1.43E-03	2.42E-04	4.49E-04	**1.59E-04**
NI+HS	T_1	4.79E+01	4.23E+01	4.30E+01	**4.22E+01**
	T_2	2.53E+02	1.12E-03	5.85E-01	**5.31E-07**
NI+MS	T_1	3.09E-05	6.41E-07	6.18E-09	**5.25E-09**
	T_2	1.15E+00	**7.44E-01**	5.56E+00	1.12E+00
NI+LS	T_1	**2.39E+02**	2.61E+02	2.67E+02	2.59E+02
	T_2	5.23E+03	**1.77E+03**	8.29E+03	1.99E+03

4.6. Sensitivity analysis of the parameters

In AMTDE-PD, there are two main control parameters δ and q, which δ controls the update condition of the RMP and q are used to adjust the value of the RMP. To analyze the effect of parameter q on the AMTDE-PD algorithm, we compare the comprehensive performance of AMTDE-PD with different values of parameter q on all problems. In this experiment, set $q = 0.4$, $q = 0.5, q = 0.6, q = 0.7, q = 0.8, q = 0.9$. The results of Friedman's test with different parameter settings are shown in Table 7. It can be seen from Table 7 that the parameter q has no significant effect on the performance of AMTDE-PD when all tasks are considered. However, from the average rank, AMTDE-PD has the best rank on T2 and has the second rank on T1 for the case of q = 0.9. So, in this paper, we recommend q = 0.9.

To analyze the influence of parameter δ on the performance of AMTDE-PD, we compared the performance of AMTDE-PD with different values of δ on all the problems in the single-target classic test set. The Friedman test for the comparison results is shown in Table 8. As shown in Table 8, the P-values for tasks T_1 and T_2 on all test problems are 0.216 and 0.463, respectively, both of which are greater than the significance level of 0.05, indicating that the value of δ has no significant impact on the performance of the algorithm AMTDE-PD. However, from the average ranking, δ = 0.6 achieved the best ranking on task T_1, followed by δ = 0.5. The best ranking was obtained with δ = 0.5 on task T_2, followed by δ = 0.3, δ = 0.6, δ = 0.4, and δ = 0.2. Through the above analysis, δ = 0.5 is recommended.

Table 7. Friedman test results of AMTDE-PD with different parameter settings.

	Task1		Task2	
	Parameter	Rank	Parameter	Rank
	q = 0.9	3.11	q = 0.9	2.67
	q = 0.8	2.67	q = 0.8	3.67
	q = 0.7	3.78	q = 0.7	3.00
	q = 0.6	4.33	q = 0.6	2.78
	q = 0.5	3.89	q = 0.5	4.76
	q = 0.4	3.22	q = 0.4	4.22
statistic	5		statistic	5
P value	0.448		P value	0.122

Table 8. Friedman test results of AMTDE-PD with different δ values.

	T_1		T_2	
	Parameter	Rank	Parameter	Rank
	$\delta = 0.2$	3.78	$\delta = 0.2$	3.44
	$\delta = 0.3$	3.28	$\delta = 0.3$	2.83
	$\delta = 0.4$	2.94	$\delta = 0.4$	3.39
	$\delta = 0.5$	2.83	$\delta = 0.5$	2.22
	$\delta = 0.6$	2.17	$\delta = 0.6$	3.11
statistic	5.272		statistic	3.598
P value	0.216		P value	0.463

4.7. Sensitivity analysis of the mating probability

To validate the effectiveness of the population similarity-based RMP in AMTDE-PD, we compare the proposed similarity-based RMP (SIM-RMP) with the pre-improvement RMP on CI+HS, PI+MS, NI+LS, CI+LS, PI+LS, and NI+ML problems are compared. The comparison results are shown in Figure 6, from which we can see that SIM-RMP has larger values for the CI+HS problem and smaller values for the PIMS and NI+LS problems.

That is to say, the values of the proposed SIM-RMP on different problems change significantly as the evolutionary algebra advances, while the RMP values before the improvement change insignificantly but oscillate significantly. This is because the improvement rate of the solution is constantly changing during the evolutionary process, especially for problems with many local optimal solutions, the improvement rate of the solution changes more frequently during the evolutionary process. Thus, adjusting the RMP based only on the improvement rate of the solution during the evolutionary process makes it take values that oscillate sharply. In addition, it can also be seen from Figure 6 that the SIM-RMP proposed in this paper has a larger value in the CI+LS problem, while the RMP before improvement has a smaller value. Therefore, the proposed SIM-RMP can effectively capture the correlation between two tasks, which in turn can effectively control the intensity of inter-task interactions, increase the efficiency of inter-task knowledge transfer, and enhance the performance of the algorithm.

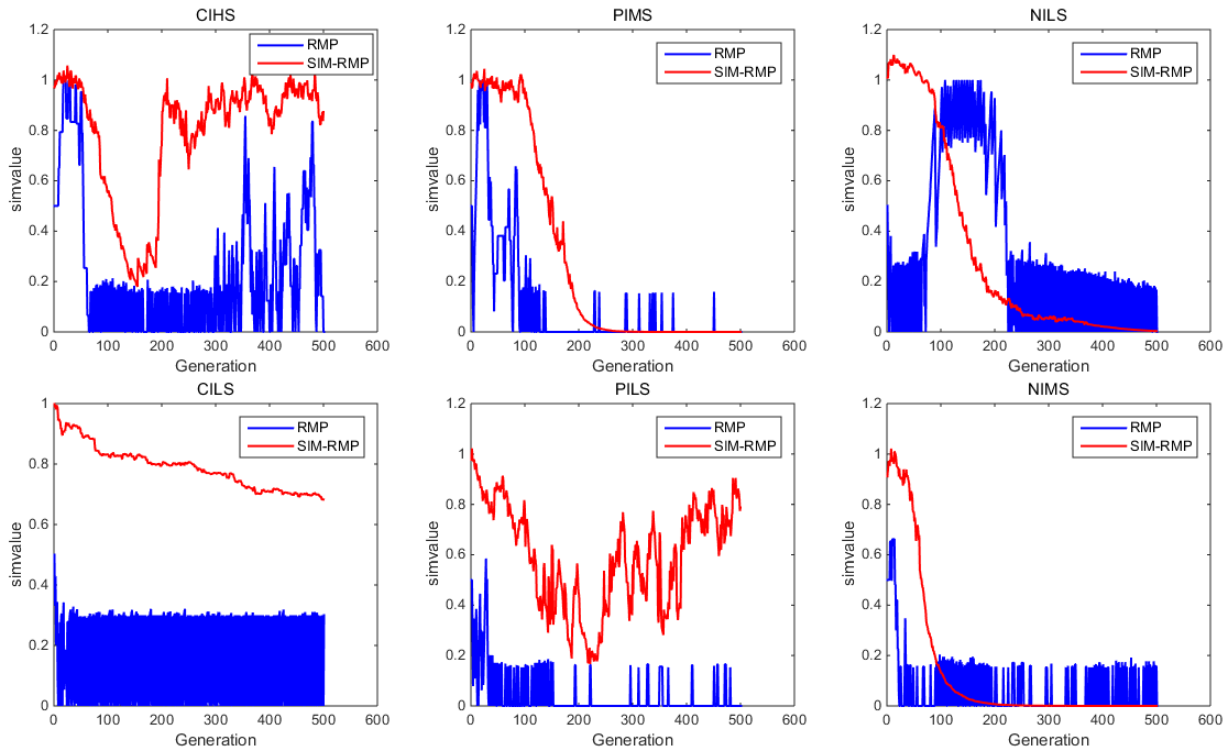


Figure 6. Comparison of random interaction probability before and after improvement.

5. Conclusions and future work

In order to make full use of the valuable information between tasks and reduce the negative transfer, we propose an adaptive multitasking differential optimization algorithm based on population distribution information (AMTDE-PD). Specifically, first, the differences in task population distribution are utilized to determine the valuable information transferred between tasks. Next, using population distribution information to evaluate the evolutionary trend between tasks and adjusting the interaction probability between tasks based on the evolutionary trend, the adaptive adjustment of interaction intensity is achieved to reduce negative transfer between tasks. Then, DE algorithm is used as the task solver to solve the related tasks. The AMTDE-PD algorithm proposed in this paper and other mainstream evolutionary multitasking optimization algorithms are experimented on CEC2019-SOMTP and WCCI2020-SOCTP, respectively, and the experimental results show that the proposed AMTDE-PD algorithm is able to solve different types of multitasking optimization problems effectively and has high robustness.

Use of AI tools declaration

We declare that we have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant (No.

62176146, No. 62272384), National Social Science Foundation of China under Grant (No. 21XTY012), National Education Science Foundation of China under Grant (No. BCA200083), Key Project of Shaanxi Provincial Natural Science Basic Research Program under Grant (No. 2023-JC-ZD-34), and Natural Science Basic Research Program of Shaanxi under Grant (No. 2022JM-050).

Conflict of interest

The authors declare that they have no conflicts of interest.

References

1. C. R. Cloninger, J. Rice, T. Reich, Multifactorial inheritance with cultural transmission and assortative mating. II. A general model of combined polygenic and cultural inheritance, *Am. J. Hum. Genet.*, **31** (1979), 176–198.
2. A. Gupta, Y. Ong, L. Feng, Multifactorial evolution: Toward evolutionary multitasking, *IEEE Trans. Evol. Comput.*, **20** (2016), 343–357. https://doi.org/10.1109/TEVC.2015.2458037
3. Z. Liang, J. Zhang, L. Feng, Z. Zhu, A hybrid of genetic transform and hyper-rectangle search strategies for evolutionary multi-tasking, *Expert Syst. Appl.*, **138** (2019), 112798. https://doi.org/10.1016/j.eswa.2019.07.015
4. K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, T. P. Siew, Linearized domain adaptation in evolutionary multitasking, in *2017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, (2017), 1295–1302. https://doi.org/10.1109/CEC.2017.7969454
5. K. K. Bali, Y. S. Ong, A. Gupta, P. S. Tan, Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II, *IEEE Trans. Evol. Comput.*, **24** (2019), 69–83. https://doi.org/10.1109/TEVC.2019.2906927
6. C. Yang, J. Ding, K. C. Tan, Y. Jin, Two-stage assortative mating for multi-objective multifactorial evolutionary optimization, in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, IEEE, (2017), 76–81. https://doi.org/10.1109/CDC.2017.8263646
7. R. T. Liaw, C. K. Ting, Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems, in 2*017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, (2017), 2266–2273. https://doi.org/10.1109/CEC.2017.7969579
8. L. Zhou, L. Feng, K. C. Tan, A. Gupta, Y. S. Ong, K. C. Tan, et al., Evolutionary multitasking via explicit autoencoding, *IEEE Trans. Cybern.*, **49** (2019), 3457–3470. https://doi.org/10.1109/tcyb.2018.2845361
9. G. Li, Q. Lin, W. Gao, Multifactorial optimization via explicit multipopulation evolutionary framework, *Inf. Sci.*, **512** (2020), 1555–1570. https://doi.org/10.1016/j.ins.2019.10.066
10. Y. Cai, D. Peng, P. Liu, J. M. Guo, Evolutionary multi-task optimization with hybrid knowledge transfer strategy, *Inf. Sci.*, **580** (2021), 874–896. https://doi.org/10.1016/j.ins.2021.09.021
11. Z. Liang, W. Liang, Z. Wang, X. Ma, L. Liu, Z. Zhu, Multiobjective evolutionary multitasking with two-stage adaptive knowledge transfer based on population distribution, *IEEE Trans. Syst. Man Cybern.: Syst.*, **52** (2021), 4457–4469. https://doi.org/10.1109/tsmc.2021.3096220
12. F. Gao, W. Gao, L. Huang, J. Xie, M. Gong, An effective knowledge transfer method based on semi-supervised learning for evolutionary optimization, *Inf. Sci.*, **612** (2022), 1127–1144. https://doi.org/10.1016/j.ins.2022.09.020

13. Y. Lai, H. Chen, F. Gu, A multitask optimization algorithm based on elite individual transfer, *Math. Biosci. Eng.*, **20** (2023), 8261–8278. https://doi.org/10.3934/mbe.2023360
14. J. Lin, H. L. Liu, K. C. Tan, F. Gu, An effective knowledge transfer approach for multi-objective multitasking optimization, *IEEE Trans. Cybern.*, **51** (2020), 3238–3248. https://doi.org/10.1109/TCYB.2020.2969025
15. H. Sun, P. Chen, Z. Hu, L. Wei, Multi-objective evolutionary multitasking algorithm based on cross-task transfer solution matching strategy, *ISA Trans.*, **138** (2023), 504–520. https://doi.org/10.1016/j.isatra.2023.03.015
16. J. Lin, H. L. Liu, B. Xue, M. Zhang, F. Gu, Multi-objective multitasking optimization based on incremental learning, *IEEE Trans. Evol. Comput.*, **24** (2019), 824–838. https://doi.org/10.1109/TEVC.2019.2962747
17. C. Wang, J. Liu, K. Wu, Z. Wu, Solving multitask optimization problems with adaptive knowledge transfer via anomaly detection, *IEEE Trans. Evol. Comput.*, **26** (2021), 304–318. https://doi.org/10.1109/TEVC.2021.3068157
18. H. Chen, H. L. Liu, F. Gu, K. C. Tan, A multiobjective multitask optimization algorithm using transfer rank, *IEEE Trans. Evol. Comput.*, **27** (2022), 237–250. https://doi.org/10.1109/TEVC.2022.3147568
19. R. Storn, K. Price, Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, **11** (1997), 341–359. https://doi.org/10.1023/A:1008202821328
20. J. Zhang, A. C. Sanderson, JADE: adaptive differential evolution with optional external archive, *IEEE Trans. Evol. Comput.*, **13** (2009), 945–958. https://doi.org/10.1109/TEVC.2009.2014613
21. Z. W. Li, L. J. Wang, Population distribution-based self-adaptive differential evolution algorithm, *Comput. Sci.*, **47** (2020), 180–185. https://doi.org/10.11896/jsjkx.181202356
22. H. Peng, Z. J. Wu, X. Y. Zhou, C. Deng, Dynamic differential evolution algorithm based on elite local learning, *Acta Electron. Sin.*, **42** (2014), 1522–1530. https://doi.org/10.3969/j.issn.0372-2112.2014.08.010
23. J. Y. Li, Z. H. Zhan, K. C. Tan, J. Zhang, A meta-knowledge transfer-based differential evolution for multitask optimization, *IEEE Trans. Evol. Comput.*, **26** (2021), 719–734. https://doi.org/10.1109/tevc.2021.3131236
24. L. Shi, Z. Hu, Q. Su, Y. Miao, A modified multifactorial differential evolution algorithm with optima-based transformation, *Appl. Intell.*, **53** (2023), 2989–3001. https://doi.org/10.1007/s10489-022-03537-w
25. Q. Dang, W. Gao, M. Gong, Dual transfer learning with generative filtering model for multi-objective multitasking optimization, *Memet. Comput.*, **15** (2023), 3–29. https://doi.org/10.1007/s12293-022-00374-9
26. A. Gupta, L. Zhou, Y. S. Ong, Z. Chen, Y. Hou, Half a dozen real-world applications of evolutionary multitasking, and more, *IEEE Comput. Intell. Mag.*, **17** (2022), 49–66. https://doi.org/10.1109/mci.2022.3155332
27. P. C. Pop, L. Fuksz, A. H. Marc, A variable neighborhood search approach for solving the generalized vehicle routing problem, in *International Conference on Hybrid Artificial Intelligence Systems*, Cham: Springer International Publishing, (2014), 13–24. https://doi.org/10.1007/978-3-319-07617-1_2

28. L. Zhou, L. Feng, J. Zhong, Y. S. Ong, Z. Zhu, E. Sha, Evolutionary multitasking in combinatorial search spaces: A case study in capacitated vehicle routing problem, in *2016 IEEE Symposium Series on Computational Intelligence* (*SSCI*), IEEE, (2016), 1–8. https://doi.org/10.1109/SSCI.2016.7850039
29. L. Feng, Y. Huang, L. Zhou, J. Zhong, A. Gupta, K. Tang, et al., Explicit evolutionary multitasking for combinatorial optimization: A case study on capacitated vehicle routing problem, *IEEE Trans. Cybern.*, **51** (2020), 3143–3156. https://doi.org/10.1109/tcyb.2019.2962865
30. Y. Huang, L. Feng, M. Li, Y. Wang, Z. Zhu, K. C. Tan, Fast vehicle routing via knowledge transfer in a reproducing Kernel Hilbert space, *IEEE Trans. Syst. Man Cybern.: Syst.*, **53** (2023), 5404–5416. https://doi.org/10.1109/TSMC.2023.3270308
31. J. Wu, H. Yang, Y. Zeng, Z. Wu, J. Liu, L. Feng, A twin learning framework for traveling salesman problem based on autoencoder, graph filter, and transfer learning, *IEEE Trans. Consum. Electron.*, **2023** (2023), 1–16. https://doi.org/10.1109/TCE.2023.3269071
32. J. Yi, J. Bai, H. He, W. Zhou, L. Yao, A multifactorial evolutionary algorithm for multitasking under interval uncertainties, *IEEE Trans. Evol. Comput.*, **24** (2020), 908–922. https://doi.org/10.1109/tevc.2020.2975381
33. J. Shi, T. Shao, X. Liu, X. Zhang, Z. Zhang, Y. Lei, Evolutionary multi-task ensemble learning model for hyperspectral image classification, *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, **14** (2020), 936–950. https://doi.org/10.1109/jstars.2020.3037353
34. Y. Jiang, Z. H. Zhan, K. C. Tan, J. Zhang, A bi-objective knowledge transfer framework for evolutionary many-task optimization, *IEEE Trans. Evol. Comput.*, **27** (2022), 1514–1528. https://doi.org/10.1109/TEVC.2022.3210783
35. Y. Zhang, K. Yang, G. W. Hao, D. Gong, Evolutionary optimization framework based on transfer learning of similar historical information, *Acta Autom. Sin.*, **47** (2021), 652–665. https://doi.org/10.16383/j.aas.c180515
36. B. Da, Y. S. Ong, L. Feng, A. K. Qin, A. Gupta, Z. Zhu, et al., Evolutionary multitasking for single-objective continuous optimization: benchmark problems, performance metric, and baseline results, preprint, arXiv:1706.03470.
37. J. Ding, C. Yang, Y. Jin, T. Chai, Generalized multitasking for evolutionary optimization of expensive problems, *IEEE Trans. Evol. Comput.*, **23** (2019), 44–58. https://doi.org/10.1109/tevc.2017.2785351
38. D. Wu, X. Tan, Multitasking genetic algorithm (MTGA) for fuzzy system optimization, *IEEE Trans. Fuzzy Syst.*, **28** (2020), 1050–1061. https://doi.org/10.1109/tfuzz.2020.2968863
39. L. Feng, W. Zhou, L. Zhou, S. W. Jiang, J. H. Zhong, B. S. Da, et al., An empirical study of multifactorial PSO and multifactorial DE, in *2017 IEEE Congress on Evolutionary Computation (CEC). Donostia, San Sebastián*, IEEE, (2017), 921–928. https://doi.org/10.1109/CEC.2017.7969407
40. X. Li, L. Wang, Q. Jiang, Multipopulation-based multi-tasking evolutionary algorithm, *Appl. Intell.*, **53** (2023), 4624–4647. https://doi.org/10.1007/s10489-022-03626-w