



ARTICLE

EMaTO-SGN: Evolutionary Many-Task Multi-Objective Optimization with Star-Topology Generative Networks

Xu Chen^{*} and Haogang Li

School of Electrical and Information Engineering, Jiangsu University, Zhenjiang, China

^{*}Corresponding Author: Xu Chen. Email: xuchen@ujs.edu.cn

Received: 23 June 2026; Accepted: 17 September 2026; Published: 22 October 2026

ABSTRACT: The evolutionary multitasking algorithm based on generative strategies (EMT-GS) has emerged as a promising approach to evolutionary multitasking. However, when extended to many-task optimization scenarios, EMT-GS may suffer from negative knowledge transfer. Moreover, the fully connected generative adversarial network (GAN) architecture employed in EMT-GS incurs quadratic time complexity with respect to the number of tasks. To address these limitations, we propose a novel algorithm, termed EMaTO-SGN (Evolutionary Many-Task Optimization with Star-Topology Generative Networks). EMaTO-SGN employs a star-topology architecture for centralized knowledge routing and scheduling, in which a central hub evaluates candidate source tasks and determines when knowledge transfer is needed. Each task is assigned a dedicated Wasserstein generative adversarial network with gradient penalty (WGAN-GP), reducing the computational complexity of generative model training and sample generation from quadratic to linear with respect to the number of tasks. To mitigate negative transfer, an inter-task scoring mechanism combines maximum mean discrepancy (MMD) with historical transfer success rates to identify eligible source tasks. In addition, an adaptive triggering strategy monitors evolutionary stagnation to balance cross-task exploration and task-specific exploitation. Experiments on the CEC2019 and WCCI2020 benchmark suites show that EMaTO-SGN achieves the best average rankings among the state-of-the-art algorithms considered. It also requires less computational time than EMT-GS under the tested task configurations. These results demonstrate the effectiveness and scalability of EMaTO-SGN for evolutionary many-task multi-objective optimization.

KEYWORDS: Evolutionary multitask optimization; many-task multi-objective optimization; generative adversarial networks; WGAN-GP; knowledge transfer

1 Introduction

Optimization problems are often characterized by high dimensionality, multiple objectives, and complex interactions among objectives. Traditional evolutionary algorithms typically solve these problems independently [1], limiting the reuse of prior optimization experience and potentially leading to redundant computational effort when handling multiple tasks. Evolutionary Multitask Optimization (EMTO) addresses this limitation by enabling knowledge reuse and joint search across tasks [2]. Its central idea is to solve multiple related optimization tasks simultaneously within a single population or a unified evolutionary framework. Through implicit or explicit cross-task knowledge transfer, EMTO allows tasks to benefit from one another, potentially accelerating convergence, helping the search escape local optima, and improving overall optimization performance and reliability [3].

As EMTO has developed, its scope has expanded to many-task multi-objective optimization, where dozens of tasks are optimized concurrently [4]. In such settings, tasks may differ substantially in their search spaces, objective distributions, Pareto-optimal sets, and Pareto fronts. Conventional transfer strategies based on direct crossover between individuals, polynomial mutation, or simple Gaussian sampling may struggle to capture the underlying structure of promising solutions across tasks. Consequently, knowledge transfer may introduce unsuitable search information, leading to negative transfer and slowing overall convergence [5].

Generative strategies have been creatively introduced into the field of EMTO to improve knowledge modeling and transfer. Evolutionary Multitasking Optimization Based on Generative Strategies (EMT-GS), proposed by Liang et al., represents a notable approach in this domain [6]. By training Generative Adversarial Networks (GANs), EMT-GS effectively learns the implicit mappings between source and target tasks. It generates high-quality transferred individuals that retain useful information from the source tasks while adapting to the characteristics of the target tasks.

Despite its strong performance in conventional multitask optimization, EMT-GS faces several limitations when extended to many-task settings. First, sparse population data during the early stages of evolution may make GAN training susceptible to mode collapse, while training on large amounts of unfiltered data can incur substantial computational overhead. Second, EMT-GS lacks a dedicated mechanism for screening tasks and training data to filter out potentially unhelpful information from heterogeneous source tasks. Finally, its pairwise network architecture requires separate mapping models for task pairs, causing the number of models to grow quadratically with the number of tasks. This may incur substantial time costs, with the complexity reaching the level of $\mathcal{O}(T^2)$, where T denotes the number of tasks.

To address these limitations, we propose Evolutionary Many-Task Optimization with Star-Topology Generative Networks (EMaTO-SGN). The proposed algorithm incorporates two key components: a star-topology generative network architecture and an inter-task scoring and selection mechanism. Instead of maintaining separate GANs for task pairs, EMaTO-SGN assigns a dedicated GAN to each target task to map solutions from selected source tasks, reducing the runtime and decreasing the computational complexity of generative knowledge transfer from $\mathcal{O}(T^2)$ to $\mathcal{O}(T)$. This design improves the scalability of EMaTO-SGN in many-task multi-objective optimization. The scoring and selection mechanism combines cross-task similarity with the empirical effectiveness of knowledge transfer to identify beneficial source tasks and dynamically regulate the direction and intensity of transfer, thereby mitigating negative transfer from multiple sources. Extensive experiments demonstrate that EMaTO-SGN achieves performance comparable to or better than that of the state-of-the-art algorithms evaluated in many-task multi-objective optimization.

2 Preliminaries and Related Work

2.1 Preliminaries of Multi-Objective Multitask Optimization

A multi-objective optimization problem (MOP) [7] can be formulated as follows:

$$\min \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})), \quad \mathbf{x} \in \Omega \quad (1)$$

Here, $\mathbf{x}$ denotes the decision vector, Ω is the search space, and M is the number of objectives. The Pareto-optimal set (PS) comprises all solutions that are not dominated by any other feasible solutions, and its image in the objective space is the Pareto front (PF). A multi-objective multitask optimization problem consists of T MOPs and aims to find the PS for each task simultaneously. In Evolutionary Multitasking (EMT), a unified search space $\mathbf{U}$ is typically used to facilitate cross-task knowledge transfer. An individual $\mathbf{y}$ is encoded in $\mathbf{U}$ and can be decoded into the original search space of any task. Each individual is characterized by three core attributes: factorial cost, which is its objective value on the task indicated by its skill factor; factorial rank,

which describes its standing relative to other individuals on a given task, as determined by non-dominated sorting and crowding distance following NSGA-II [8]; and skill factor, which identifies the task on which the individual performs best [2].

2.2 GAN and WGAN-GP

Generative adversarial networks (GANs), introduced by Goodfellow et al. [9], learn data distributions through adversarial training between a generator and a discriminator. However, the original GAN formulation is susceptible to mode collapse and vanishing gradients, which can destabilize training. Arjovsky et al. proposed the Wasserstein GAN (WGAN) [10], which uses the Wasserstein distance as the loss function. The Wasserstein distance requires the discriminator function to satisfy the 1-Lipschitz continuity condition.

Although the WGAN uses weight clipping to enforce this constraint, this strategy can hinder training or cause vanishing gradients. To address these issues, Gulrajani et al. proposed the Wasserstein GAN with Gradient Penalty (WGAN-GP) [11]. WGAN-GP introduces a gradient penalty term into the loss function to encourage satisfaction of the Lipschitz constraint. The critic loss is defined as follows:

$$\mathcal{L}_C = \mathbb{E}_{\mathbf{x} \sim p_g} [C(\mathbf{x})] - \mathbb{E}_{\mathbf{x} \sim p_{data}} [C(\mathbf{x})] + \lambda_{GP} \mathbb{E}_{\hat{\mathbf{x}} \sim p_{\hat{\mathbf{x}}}} [(\|\nabla_{\hat{\mathbf{x}}} C(\hat{\mathbf{x}})\|_2 - 1)^2] \quad (2)$$

where $\hat{\mathbf{x}}$ represents the linear interpolation between real sample $\mathbf{x}$ and generated samples $\tilde{\mathbf{x}}$, defined as $\hat{\mathbf{x}} = \alpha \mathbf{x} + (1 - \alpha) \tilde{\mathbf{x}}$ with $\alpha \sim U[0, 1]$, and λ_{GP} is the gradient penalty coefficient. The penalty encourages the critic's gradient norm at the interpolated samples to remain close to 1, providing an approximate means of enforcing the 1-Lipschitz constraint. By improving training stability and mitigating mode collapse, WGAN-GP provides a suitable generative modeling foundation for EMaTO-SGN.

The generator G is trained to maximize the critic's score for generated samples by minimizing the following loss:

$$\mathcal{L}_G = -\mathbb{E}_{\tilde{\mathbf{x}} \sim p_g} [C(\tilde{\mathbf{x}})] \quad (3)$$

where p_g denotes the distribution of the generated mapped solutions. Optimizing the critic and generator using $\mathcal{L}_C$ and $\mathcal{L}_G$, respectively, enables WGAN-GP to learn a mapping from source-task solutions to promising regions of the target task's search space.

2.3 Related Work

As evolutionary multitask optimization (EMTO) expands to many-task optimization (EMaTO), key challenges include efficiently assessing task similarity, identifying useful knowledge for transfer, and developing generative models for complex or computationally expensive optimization problems. Drawing on recent surveys [12,13] and methodological advances, this section reviews related work across several thematic areas.

2.3.1 Architectural Design for EMTO and EMaTO

Early work on evolutionary multitasking frameworks, exemplified by the Multifactorial Evolutionary Algorithm (MFEA), established a foundation for implicit knowledge transfer within a unified search space. MFEA-II [4] subsequently introduced an explicit probabilistic model learned online to address more complex multitasking scenarios. As the number of tasks increases, many-task optimization (MaTO) frameworks have attracted growing research interest. For instance, the group-based multifactorial evolutionary algorithm (GMFEA) [14] groups similar tasks and restricts inter-task genetic information transfer to tasks within the same group. Dynamic resource allocation has also been investigated to improve the allocation of computational resources across tasks [15]. Evolutionary multitasking has also been extended to bi-level optimization,

where knowledge transfer is used to assist the solution of related lower-level optimization problems [16]. MaTEA (Many-Task Evolutionary Framework with Adaptive Archive) [17] maintains a separate archive for each task and adaptively selects auxiliary tasks using both Kullback-Leibler (KL) divergence and historical transfer rewards.

2.3.2 Cross-Task Knowledge Transfer Mechanisms and Negative Transfer Suppression

Determining what knowledge to transfer and how much to transfer is central to EMTO. Adaptive mechanisms have been widely explored to mitigate negative transfer. Maximum Mean Discrepancy (MMD) method [18] is used to measure inter-task distances. The Multi-Armed Bandit (MAB) model [19] treats potential auxiliary tasks as different arms of a bandit [20]. The MAB mechanism achieves a dynamic balance between exploring unknown tasks and exploiting known high-value tasks.

MTEA-DCK [21] employs a strength-based Pareto-competitive mechanism to divide the population into winners and losers with distinct roles. MTDE-MKTA [22] adopts a mechanism based on the Evolution-Path-Model. Liang et al. proposed an EMaTO framework based on multi-source knowledge transfer (EMaTO-MKT) [23], which adaptively adjusts the probability of knowledge transfer based on the evolutionary state. Dai et al. used graph attention networks to capture more complex non-linear topological correlations [24]. Wu et al. proposed a similarity measure based on Shift Invariance, enabling knowledge reuse under severe shifts [25]. An EMTO algorithm with adaptive transfer intensity evaluates skill factors and inter-task correlations to dynamically adjust the frequency of cross-task knowledge transfer [18]. A two-stage knowledge transfer framework [26] offers another approach for EMaTO.

2.3.3 Preservation and Precise Screening of Knowledge Structures

Preserving useful information while filtering out noise during knowledge extraction is essential for efficient transfer. Jiang et al. proposed a Knowledge Structure Preserving (KSP) method that projects source task knowledge into a low-dimensional subspace while preserving its intrinsic topological structure [27]. Zhang et al. extracted the distribution direction of elite solutions to characterize the evolutionary dynamics of the source task [26]. The Task-Averaged Natural Gradient (TNG) captures global evolutionary trends to reduce the complexity of knowledge transfer [28]. However, in multi-objective optimization or problems with complex constraints, not all elite individuals from source tasks provide beneficial guidance for the target task. To address this issue, a knowledge classification-assisted evolutionary multitasking framework [29] combines domain adaptation with classifier training to identify and filter out noisy knowledge.

2.3.4 Feature Mapping Based on Generative Strategies and Deep Models

Traditional linear mapping methods may be insufficient for complex EMaTO problems, motivating the use of deep learning models. Approaches based on denoising autoencoders (DAEs) [5] use latent-space feature extraction to align non-dominated solution sets across heterogeneous tasks. Connections between evolutionary algorithms and attention-based sequence models have also been investigated. Zhang et al. [30] analyzed the correspondence between evolutionary operators and components of Transformer architectures. Separately, Zhang et al. [31] explored the use of large language models as meta-surrogates for offline data-driven many-task optimization. Federated data-driven ensemble surrogates have also shown promise for optimization problems with expensive evaluations [32,33].

GMOEA, proposed by He et al. [34], uses GANs to generate offspring. EMT-GS, proposed by Liang et al. [6], incorporates GANs into EMTO and provides a foundation for the present study. Through adversarial training between a generator and a discriminator, EMT-GS implicitly learns source-task distribution patterns and generates transfer solutions adapted to the target task's search space.

StarGAN [35] provides an example of unified generative modeling for image-to-image translation across multiple domains. It uses a single generative model conditioned on domain labels, avoiding the need for separate generators for domain pairs. EMaTO-SGN adopts a different architecture tailored to evolutionary many-task optimization, in which the star topology refers to centralized knowledge routing and scheduling. The central hub performs dynamic task scoring rather than neural network-based mapping, while a separate lightweight WGAN-GP handles generative mapping for each task. This separation supports cross-task feature alignment with a number of generative models that grows linearly with the number of tasks.

2.3.5 Motivation of This Paper

EMT-GS enables knowledge transfer by incorporating GANs into offspring generation. However, its transfer efficiency remains limited in many-task settings, where high computational costs further restrict its applicability. These limitations motivate the development of EMaTO-SGN, which builds on EMT-GS and introduces a star-topology knowledge-transfer framework to improve transfer efficiency and computational scalability in evolutionary many-task optimization.

As shown in Fig. 1, the fully connected architecture of EMT-GS assigns a separate set of neural networks (blue nodes) to each pair of tasks (orange nodes) for knowledge transfer. Consequently, the number of networks grows quadratically with the number of tasks, resulting in substantial computational and memory overhead in many-task settings. To address this limitation, EMaTO-SGN replaces the pairwise mapping architecture with a star topology organized around a central hub for knowledge routing and scheduling (green node). The central hub evaluates inter-task relationships, identifies tasks that require assistance, and adaptively schedules the activation of generative models to reduce computational overhead. Each task is assigned a single independent generative mapping network (blue node). The hub uses a dynamic scoring and selection mechanism based on task similarity and transfer effectiveness to select source tasks and regulate the direction and intensity of knowledge transfer. Its role is to coordinate knowledge transfer, while the task-specific networks perform the generative mapping.

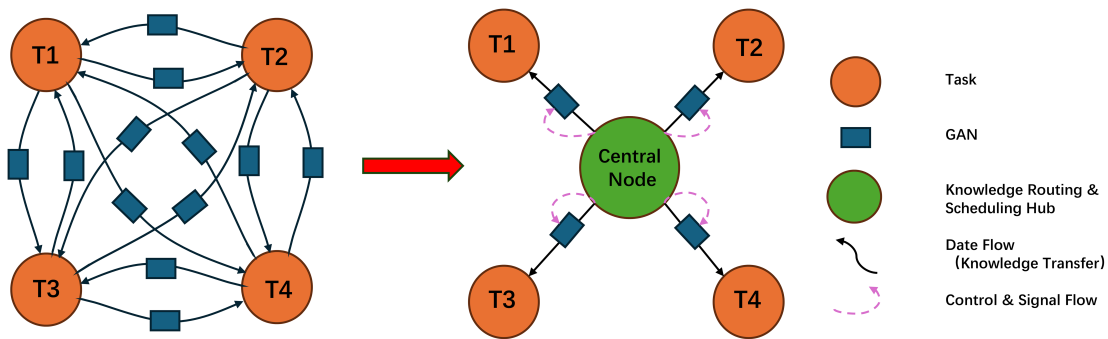


Figure 1: Structural comparison of the two algorithms (traditional fully connected topology vs. proposed star-shaped topology).

3 Evolutionary Many-Task Optimization with Star-Topology Generative Networks

3.1 Overall Workflow and Core Architecture

EMaTO-SGN aims to enable efficient knowledge transfer in many-task optimization while mitigating negative transfer through coordinated resource allocation. To address the scalability limitations of EMT-GS, it replaces the fully connected pairwise mapping architecture with a star-topology organized around centralized knowledge routing and scheduling. This design reduces the computational complexity of generative model training and sample generation from $\mathcal{O}(T^2)$ to $\mathcal{O}(T)$, where T is the number of tasks.

As shown in Fig. 2, EMaTO-SGN comprises four main stages: (1) System and Population Initialization, (2) Dynamic Task Scoring and Whitelist Extraction, (3) Integrated Evolution and Knowledge Transfer, and (4) Result Output. The corresponding pseudocode is presented in Algorithm 1.

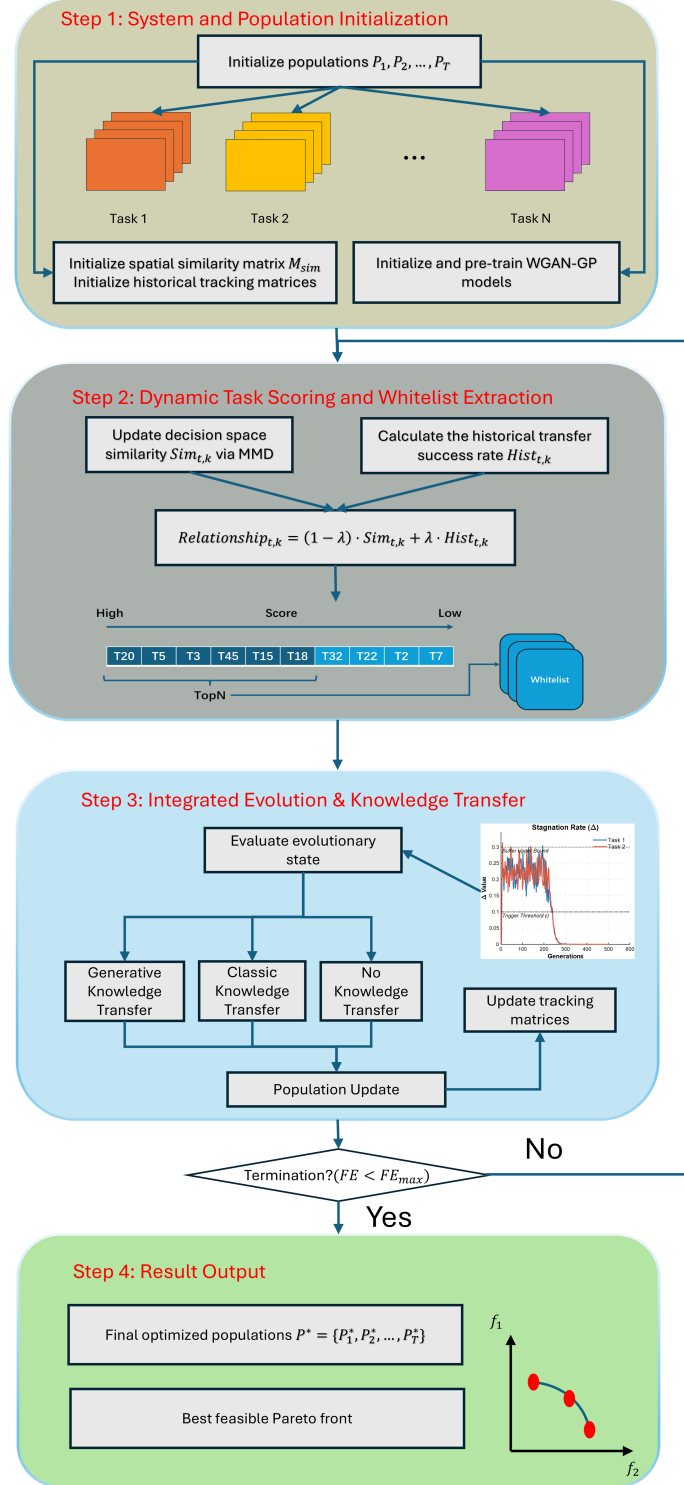


Figure 2: Overall workflow of the proposed EMaTO-SGN algorithm.

Algorithm 1: Main framework of EMaTO-SGN

Input: Number of tasks T , population size per task N , maximum function evaluations $FE_{\max}$, generative transfer allocation ratio p_{gan} , similarity update interval G_{sim}

Output: Final populations $\mathcal{P} = \{P_1, P_2, \dots, P_T\}$

- 1 // Step 1: Initialization
- 2 Initialize populations $\mathcal{P} = \{P_1, P_2, \dots, P_T\}$ and evaluate all initial individuals;
- 3 Set $FE \leftarrow TN$ and $g \leftarrow 0$;
- 4 Compute initial convergence distances $\{\tilde{d}_t^{(0)}\}_{t=1}^T$ from the evaluated populations;
- 5 Initialize the success matrix $\mathbf{S}$ and attempt matrix $\mathbf{A}$ to zero;
- 6 Construct the initial similarity matrix $\mathbf{M}_{\text{sim}}$ from population-centroid distances;
- 7 Construct the initial source-task whitelists;
- 8 Initialize the WGAN-GP models $\{G_t, C_t\}_{t=1}^T$;
- 9 Pretrain the models with nonempty initial whitelists using whitelisted source solutions as generator inputs and target populations as real samples;
- 10 **while** $FE + TN \leq FE_{\max}$ **do**
- 11 // Steps 2 and 3: Execute Algorithm 2
- 12 Execute Algorithm 2 using the current populations, stored similarity matrix, convergence-distance history, tracking matrices, generation index g , and evaluation counter FE ;
- 13 Store the updated populations, similarity matrix $\mathbf{M}_{\text{sim}}$, convergence distances $\{\tilde{d}_t^{(g+1)}\}_{t=1}^T$, and tracking matrices $\mathbf{A}$ and $\mathbf{S}$;
- 14 $FE \leftarrow FE + TN$;
- 15 $g \leftarrow g + 1$;
- 16 **end**
- 17 // Step 4: Output
- 18 **return** $\mathcal{P}$

3.2 System and Population Initialization

EMaTO-SGN initializes one independent population for each of the T tasks and records the initial objective-space extremes and convergence distances. It also initializes matrices for tracking task scores, spatial similarity, and transfer performance. Initial spatial similarity is assessed using Euclidean distances between the population centroids of different tasks in the decision space. The resulting distance matrix is then used to construct an initial related-task whitelist with a capacity of N_{src} for each task.

Algorithm 2: Scoring and evolution module

Input: Populations $\mathcal{P}$, similarity matrix $\mathbf{M}_{\text{sim}}$, convergence-distance history $\mathcal{D}$, tracking matrices $\mathbf{A}$ and $\mathbf{S}$, generation index g , evaluation counter FE , and configured parameters

Output: Updated $\mathcal{P}$, $\mathbf{M}_{\text{sim}}$, $\mathcal{D}$, $\mathbf{A}$, and $\mathbf{S}$

- 1 $\omega_g \leftarrow \min(1, FE/FE_{\max})$;
- 2 **if** $g > 0$ **and** $g \bmod G_{\text{sim}} = 0$ **then**
- 3 Update $\mathbf{M}_{\text{sim}}$ using MMD according to [Eq. \(4\)](#);
- 4 **end**
- 5 // Task scoring and candidate generation
- 6 **for** $t = 1$ **to** T **do**

(Continued)

Algorithm 2 (continued)

```

7   Compute  $H_{t,k}$  and  $R_{t,k}$  for  $k \neq t$  using Eqs. (5) and (6);
8   Construct  $\mathcal{W}_t$  by thresholding  $R_{t,k}$  and retaining up to  $N_{\text{src}}$  top candidates;
9   Set  $s_t \leftarrow 0$  and  $\mathcal{X}_t^{\text{gen}} \leftarrow \emptyset$ ;
10  if  $g > 0$  and  $\mathcal{W}_t \neq \emptyset$  then
11    Compute  $\Delta_t$  from  $\mathcal{D}$  and sample  $s_t \sim \text{Bernoulli}(P_{\text{tr},t})$  using the trigger rule in Section 3.4;
12  end
13  if  $s_t = 1$  and  $p_{\text{gan}} > 0$  then
14    Select source task  $k_t \in \mathcal{W}_t$  and sample source solutions  $X_{k_t}$ ;
15    if the model  $t$  has not yet been trained or  $g \bmod G_{\text{train}} = 0$  then
16      Train  $\{G_t, C_t\}$  for  $E_{\text{train}}$  epochs using  $P_t$  as real data and  $X_{k_t}$  as inputs;
17    end
18     $\mathcal{X}_t^{\text{gen}} \leftarrow G_t(X_{k_t})$ ;
19  end
20 end
21 // Offspring generation and selection
22 for  $t = 1$  to  $T$  do
23   if  $s_t = 1$  then
24     Generate  $N$  offspring  $O_t$  using the GAN-assisted route with  $\mathcal{X}_t^{\text{gen}}$  and the conventional KT
       route with  $\mathcal{W}_t$ , in proportions  $p_{\text{gan}}$  and  $1 - p_{\text{gan}}$ , respectively;
25   else
26     Generate  $N$  offspring  $O_t$  using task-specific evolution;
27   end
28   Record the generation method and contributing source-task identities for each offspring;
29   Evaluate all offspring in  $O_t$  on task  $t$ ;
30   Update  $\mathbf{A}$  by counting evaluated transfer offspring for each source–target task pair;
31    $P_t \leftarrow \text{NSGA\_II\_Selection}(P_t \cup O_t, N)$ ;
32   Update  $\mathbf{S}$  by counting the surviving transfer offspring from  $O_t$  for each source–target task pair;
33   Compute  $\tilde{d}_t^{(g+1)}$  from  $P_t$  and store it in  $\mathcal{D}$ ;
34 end
35 return  $\mathcal{P}, \mathbf{M}_{\text{sim}}, \mathcal{D}, \mathbf{A}, \mathbf{S}$ 

```

Finally, each task t is assigned an independent WGAN-GP model comprising a generator G_t and a critic C_t . During pretraining, source-task solutions selected from the initial whitelist are used as generator inputs, while solutions from the target population P_t serve as real samples for critic training. The critic compares these real samples with the mapped solutions produced by G_t . Pretraining uses the initialized populations without additional objective-function evaluations. If the initial whitelist is empty, pretraining is deferred until an eligible source task becomes available.

3.3 Dynamic Task Scoring and Whitelist Extraction

EMaTO-SGN updates the inter-task similarity matrix every G_{sim} generations and reuses the stored similarity scores between updates. Historical transfer performance and combined relationship scores are updated each generation to construct a source-task whitelist for each target task. Maximum mean discrepancy (MMD) [36] with a Gaussian kernel is used to quantify distributional differences between task samples in the decision space. The resulting discrepancy is transformed into a prior similarity score $\text{Sim}_{t,k}$ within the interval $(0, 1]$ as shown in Eq. (4):

$$\text{Sim}_{t,k} = \frac{1}{1 + \text{MMD}(X_t, X_k)} \quad (4)$$

Transfer effectiveness is assessed based on offspring survival during environmental selection. For transfer from source task k to target task t , $A_{t,k}$ counts the evaluated offspring produced using information from task k , and $S_{t,k}$ counts those offspring that survive environmental selection. Both counts are accumulated across generations. Generated candidates that are not used to produce evaluated offspring are excluded from these counts. The historical transfer performance score is computed as shown in Eq. (5):

$$H_{t,k} = \frac{S_{t,k} + 1}{A_{t,k} + 2} \quad (5)$$

This counting scheme ensures $0 \leq S_{t,k} \leq A_{t,k}$. The additive constants provide smoothing and yield an initial score of 0.5 when no transfer attempts have been recorded.

A dynamic fusion factor $\omega_g = \min(1.0, \frac{FE}{FE_{\max}})$ is introduced to balance the prior distribution similarity and historical transfer success. ω_g increases with the fraction of the evaluation budget consumed. The combined inter-task relationship score is constructed via a weighted sum as shown in Eq. (6)

$$R_{t,k} = (1 - \omega_g)\text{Sim}_{t,k} + \omega_g H_{t,k} \quad (6)$$

For each target task t , candidate source tasks $k \neq t$ with $R_{t,k} < \tau_{\text{rel}}$ are excluded, where τ_{rel} is a predefined relationship-score threshold. The remaining candidates are ranked in descending order of $R_{t,k}$, and up to N_{src} tasks are retained to form the source-task whitelist $\mathcal{W}_t$.

3.4 Integrated Evolution and Knowledge Transfer

The initialized populations are indexed by $g = 0$. Because no previous-generation distance is available at this stage, the first evolutionary iteration uses task-specific evolution. Adaptive transfer triggering is enabled once convergence-distance records from two successive generations are available. EMaTO-SGN monitors the evolutionary progress of each population to trigger knowledge transfer when needed. For task t , the relative convergence improvement Δ_t is calculated from the change in $\bar{d}_t^{(g)}$, the average Euclidean distance from the population members to the ideal point, as shown in Eq. (7)

$$\Delta_t = \frac{\bar{d}_t^{(g-1)} - \bar{d}_t^{(g)}}{\bar{d}_t^{(g-1)} + 10^{-6}} \quad (7)$$

where g denotes the current generation.

A predefined stagnation threshold $\varepsilon_{\text{stag}} > 0$ and a buffer multiplier $B \geq 1$ determine the task-specific transfer trigger probability $P_{\text{tr},t}$. For $B > 1$, the probability is set to 1 when $\Delta_t \leq \varepsilon_{\text{stag}}$ and to 0 when $\Delta_t \geq B\varepsilon_{\text{stag}}$. Between these thresholds, the probability decreases linearly from 1 to 0 as Δ_t increases. When $B = 1$, the transition region vanishes, and a hard-threshold rule is used: the probability is 1 at or below $\varepsilon_{\text{stag}}$ and 0 above it. A binary transfer indicator is then sampled as $s_t \sim \text{Bernoulli}(P_{\text{tr},t})$. This mechanism adjusts transfer activation according to the target task's evolutionary progress.

If transfer is not activated or the source-task whitelist is empty, all individuals undergo task-specific evolution. When transfer is activated, individuals are assigned to one of two routes. A proportion p_{gan} follows the generative transfer route, which uses candidate solutions $\mathbf{x}_{t \leftarrow k}^{\text{gen}}$ produced by the WGAN-GP mapping. The remaining proportion $(1 - p_{\text{gan}})$ follows the non-generative knowledge transfer (KT) route, which uses conventional crossover operators. In the generative route, mapped candidates are incorporated

into a differential evolution (DE) mutation operator to guide cross-task exploration. The resulting mutation vector is defined as in Eq. (8):

$$\mathbf{v}_i = \mathbf{x}_i + r(\mathbf{x}_{t \leftarrow k}^{\text{gen}} - \mathbf{x}_i) + F_{\text{DE}}(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}), \quad r \sim U(0, 1) \quad (8)$$

where $\mathbf{x}_{r_1}$ and $\mathbf{x}_{r_2}$ are mutually distinct individuals randomly selected from the current target task population, and the scaling factor F_{DE} is dynamically sampled from a normal distribution $\mathcal{N}(0.5, 0.2^2)$. The first two terms interpolate between the current individual and the GAN-mapped candidate, while the third adds a differential perturbation derived from the target population to promote exploration. Environmental selection is subsequently performed using the non-dominated sorting and crowding distance procedures of NSGA-II [8].

3.5 Generative Knowledge Transfer Module

WGAN-GP is incorporated into EMaTO-SGN to model and transfer information contained in the nonlinear distribution of high-quality source-task solutions. To accommodate the computational demands of evolutionary optimization, the generator G_t is adapted as a cross-task feature mapper that takes source-task decision vectors as input. When transferring knowledge from source task k to target task t , the source solution set X_k is mapped to a set of candidate solutions in the target task's search space:

$$X_{t \leftarrow k}^{\text{gen}} = G_t(X_k) \quad (9)$$

Using source-task solutions as generator inputs allows the generator to exploit geometric information from regions already explored by the source task. These inputs provide a structured basis for knowledge transfer by incorporating information from previously identified high-quality solutions.

The generator and critic use lightweight multilayer perceptron (MLP) architectures with layer widths determined by the decision-space dimension D , as detailed in Table 1. Standard WGAN-GP training typically performs multiple critic updates per generator update, for example, $n_{\text{critic}} = 5$. Repeated critic updates can incur substantial computational overhead when training models for many tasks. EMaTO-SGN therefore adopts a 1:1 update ratio, performing one critic update and one generator update per training iteration to reduce the number of network updates.

Table 1: Network architectures of the generator and critic, where D denotes the decision-variable dimension.

Network	Layer	Neurons	Activation	Normalization	Output Range
Generator	Hidden layer 1	D	Leaky ReLU (0.5)	Batch norm	–
	Hidden layer 2	D	Leaky ReLU (0.5)	Batch norm	–
	Output layer	D	Sigmoid	None	$[0, 1]$
Critic	Hidden layer 1	D	Leaky ReLU (0.5)	None (GP)	–
	Output layer	1	Linear	None	Unbounded

3.6 Complexity Analysis

The computational cost of EMaTO-SGN consists of evolutionary operations, inter-task similarity evaluation, and generative knowledge transfer. Let T denote the number of tasks, N the population size per task, M the number of objectives, and D the decision-space dimension. The following analysis concerns the main evolutionary loop and excludes objective-function evaluations and one-time initialization.

Environmental selection uses the non-dominated sorting and crowding distance procedures of NSGA-II. Selecting N individuals from a combined parent–offspring population of size $2N$ costs $\mathcal{O}(MN^2)$ per task, giving $\mathcal{O}(TMN^2)$ across all tasks. Offspring generation using DE or GA operators requires an additional $\mathcal{O}(TND)$.

With N samples per task, pairwise MMD evaluation using a Gaussian kernel requires $\mathcal{O}(N^2D)$ for each task pair. A full similarity-matrix update therefore costs $\mathcal{O}(T^2N^2D)$. Updating the matrix every G_{sim} generations gives an amortized cost of $\mathcal{O}(T^2N^2D/G_{\text{sim}})$ per generation. Historical-score updates require $\mathcal{O}(T^2)$, while constructing all whitelists by sorting candidate scores requires $\mathcal{O}(T^2 \log T)$.

Each task-specific WGAN-GP has a fixed number of layers with hidden widths proportional to D . For a batch size of N , a generator update costs $\mathcal{O}(ND^2)$. A critic update, including real and generated sample evaluation and gradient-penalty computation, has the same asymptotic order under this fixed architecture. With a 1:1 critic-to-generator update ratio and one batch per epoch, E_{train} training epochs cost $\mathcal{O}(E_{\text{train}}ND^2)$ per task. Generating N mapped candidates adds $\mathcal{O}(ND^2)$.

Let $\rho \in [0, 1]$ denote the average proportion of tasks that perform generative transfer in a generation. An upper bound on the average cost of training and generation is $\mathcal{O}(\rho T(E_{\text{train}} + 1)ND^2)$. This bound assumes that every active task may require model training before sample generation; periodic training can reduce the actual cost.

Combining these components gives the following amortized upper bound per generation:

$$\mathcal{O}\left(TMN^2 + TND + \frac{T^2N^2D}{G_{\text{sim}}} + T^2 \log T + \rho T(E_{\text{train}} + 1)ND^2\right).$$

The $\mathcal{O}(T^2)$ historical-score updates are subsumed by the whitelist-sorting term for $T \geq 2$.

For a generation in which the similarity matrix is updated and all task-specific generative models are trained and used, the worst-case upper bound is

$$\mathcal{O}(TMN^2 + TND + T^2N^2D + T^2 \log T + T(E_{\text{train}} + 1)ND^2).$$

With the remaining parameters fixed, generative model training and sample generation scale linearly with the number of tasks. The overall algorithm nevertheless retains superlinear costs from pairwise similarity evaluation and source-task selection. The star topology therefore improves the scalability of the generative mapping component without making the entire algorithm linear in T .

4 Experimental Setup and Results

4.1 Experimental Background and Metrics

In this section, we compare the performance of the proposed EMaTO-SGN against state-of-the-art algorithms and the baseline EMT-GS framework. These experiments employ many-task optimization test problems based on the CEC2019 [37] and WCCI2020 [38] test suites via MToP [39], which is a MATLAB benchmarking platform for evolutionary multitasking. Comparisons are performed with 10, 20, and 50 tasks. The comparison with EMT-GS based on CEC19-MaTMO1 examines both optimization performance, measured by IGD, and computational time. Ablation studies on CEC19-MaTMO1 and WCCI20-MaTMO1 assess the contributions of individual components of EMaTO-SGN.

The two benchmark suites are used to evaluate optimization performance across different problem instances and task counts. Experiments with 10, 20, and 50 tasks examine how performance changes as the number of concurrently optimized tasks increases.

Performance is quantified using three primary metrics: IGD, IGD-MTS, and Runtime. IGD evaluates the convergence and coverage of an approximation to the Pareto front, with lower values indicating better performance. In traditional single- or few-task (such as 2 tasks) optimization experiments, performance comparisons are typically conducted by directly computing the average IGD for each algorithm across all problems. However, in many-task environments (e.g., a 50-task benchmark suite), the true Pareto fronts of different tasks often exhibit severe scale disparities in the objective space. For instance, the IGD of certain local tasks might be on the order of 10^{-3} , while tasks with heavy scaling features might possess IGDs as high as 10^2 . If the IGD values are simply averaged, severe bias may be introduced. The performance advantages on small-scale tasks could be entirely eclipsed by the large-scale tasks, which can lead to severe distortion in the overall evaluation results.

The Multitask Score (MTS) was introduced to achieve a fair and comprehensive overall performance evaluation. This metric is recognized in the Multitask Multiobjective Optimization domain as IGD-MTS [39]. It adopts a more precise Z-score standardization to eliminate inter-task scale effects. The specific calculation procedure is as follows. First, for each independent task t , the raw IGD values produced by all compared algorithms across all independent runs are collected. For the t -th task in the g -th generation, the mean μ_t and standard deviation σ_t of the IGD values for all compared algorithms are calculated. Subsequently, the raw IGD value of each algorithm on that task is transformed into a standardized score. After completing the standardization transformation for all sub-tasks, the final multitask comprehensive score MTS for the algorithm is obtained by calculating the arithmetic mean of the standardized scores across all T tasks:

$$\text{IGD-MTS} = \frac{1}{T} \sum_{t=1}^T \frac{\text{IGD}_t - \mu_t}{\sigma_t}. \quad (10)$$

Runtime is recorded in the MATLAB environment to assess overall computational efficiency, including the overhead associated with generative model training and knowledge transfer. Because reducing the computational cost of EMT-GS is a central design objective, runtime is evaluated alongside solution quality.

4.2 Parameter Settings

The comparison algorithms include KR-MTEA [40], MO-EMaTO-MKT [23], MO-MaTDE [17], MO-SBO [41], MTDE-MKTA [22], MTEA-DCK [21] and EMT-GS [6]. Their algorithm-specific parameters are set according to the respective original publications.

Table 2 summarizes the experimental settings. The population size is set to $N = 100$, with a maximum number of fitness evaluations $FE_{\max} = 50,000 \times T$. Each experiment is repeated for 20 independent runs. For EMaTO-SGN, the inter-task similarity matrix is updated every $G_{\text{sim}} = 20$ generations. The stagnation threshold and buffer multiplier are set to $\epsilon_{\text{stag}} = 0.2$ and $B = 4$. The WGAN-GP models are trained using Adam, with learning rates of $lr_C = 0.0001$ for the critic and $lr_G = 0.00015$ for the generator. Additional network configurations include a batch size of $BS = 100$, a compressed training schedule of $E_{\text{train}} = 2$, a GAN resource allocation probability of $p_{\text{gan}} = 0.2$, and a whitelist capacity of $N_{\text{src}} = 6$.

Table 2: Experimental configurations of the MToP platform, EMaTO-SGN, and the generative models.

Parameter	Setting
Platform configuration	
Population size per task (N)	100
Decision-space dimension (D)	50
Maximum function evaluations	$50,000 \times T$
Independent runs	20
Differential evolution (DE)	
Scaling factor (F_{DE})	$\mathcal{N}(0.5, 0.2^2)$, bounded to $[0, 1]$
Crossover rate (CR)	0.6
Simulated binary crossover (SBX)	
Crossover probability (p_c)	0.5 per variable
Distribution index (η_c)	20
Polynomial mutation (PM)	
Mutation probability (p_m)	$1/D$
Distribution index (η_m)	15
EMaTO-SGN configuration	
GAN resource quota (p_{gan})	0.2
Whitelist capacity (N_{src})	6
Base stagnation threshold (ϵ_{stag})	0.2
Soft buffer multiplier (B)	4
MMD calculation interval (G_{sim})	20 generations
Generative-model configuration	
Critic learning rate (lr_C)	0.0001
Generator learning rate (lr_G)	0.00015
Batch size (BS)	100
Training interval (G_{train})	5 generations
Training epochs (E_{train})	2

Note: T denotes the number of component tasks. $\mathcal{N}(\mu, \sigma^2)$ denotes a Gaussian distribution with mean μ and variance σ^2 .

4.3 Experimental Results and Analysis

4.3.1 Overall Performance Comparison

EMaTO-SGN is compared with the baseline algorithms on the CEC19 and WCCI20 benchmark suites with 10, 20, and 50 tasks. As reported in [Tables 3–8](#), EMaTO-SGN achieves the best overall average rank in all six combinations of benchmark suite and task count. Its average ranks range from 2.49 to 2.79, with an average rank of 2.79 on the 50-task WCCI20 suite ([Table 8](#)).

Table 3: Average rankings of the compared algorithms on the 10-task CEC19 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19-MaTMO1	2.34	5.32	2.62	5.67	6.98	1.10	3.96
CEC19-MaTMO2	1.22	3.96	1.89	3.71	6.04	6.91	4.28
CEC19-MaTMO3	4.55	4.47	6.57	4.18	4.78	2.42	1.02
CEC19-MaTMO4	3.62	1.66	4.40	4.13	2.40	4.99	6.82
CEC19-MaTMO5	2.12	6.89	4.45	4.10	6.04	1.13	3.28
CEC19-MaTMO6	1.40	5.30	2.18	5.91	6.58	2.45	4.18
Average rank	2.54	4.60	3.69	4.62	5.47	3.17	3.92
p-value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*	0.0042*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Table 4: Average rankings of the compared algorithms on the 10-task WCCI20 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
WCCI20-MaTMO1	2.40	5.38	3.18	5.31	7.00	1.10	3.62
WCCI20-MaTMO2	6.13	5.98	5.58	2.51	3.27	1.32	3.21
WCCI20-MaTMO3	1.80	5.02	2.21	5.69	6.96	2.16	4.15
WCCI20-MaTMO4	1.76	3.33	2.01	5.78	6.13	4.78	4.21
WCCI20-MaTMO5	2.76	4.18	3.34	4.39	4.19	3.94	5.20
WCCI20-MaTMO6	1.83	3.70	2.08	5.58	5.65	4.55	4.60
WCCI20-MaTMO7	2.49	3.68	3.05	4.78	5.36	4.19	4.46

(Continued)

Table 4 (continued)

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
WCCI20-MaTMO8	2.47	3.75	3.12	5.17	5.20	3.46	4.83
WCCI20-MaTMO9	2.48	4.17	2.98	5.09	4.91	3.65	4.73
WCCI20-MaTMO10	2.52	4.34	2.83	5.13	5.38	3.38	4.41
Average rank	2.66	4.35	3.04	4.94	5.41	3.25	4.34
p-value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Table 5: Average rankings of the compared algorithms on the 20-task CEC19 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19-MaTMO1	2.21	5.20	2.84	5.75	6.97	1.13	3.89
CEC19-MaTMO2	1.17	3.93	1.84	3.82	6.07	6.89	4.28
CEC19-MaTMO3	4.12	4.62	6.68	4.27	4.75	2.51	1.05
CEC19-MaTMO4	3.75	1.00	4.64	4.23	2.36	5.16	6.86
CEC19-MaTMO5	2.16	6.84	4.46	4.04	6.11	1.12	3.27
CEC19-MaTMO6	1.51	5.24	2.08	5.86	6.66	2.52	4.14
Average rank	2.49	4.47	3.76	4.66	5.49	3.22	3.92
p-value	Reference	<0.0001*	0.0001*	<0.0001*	<0.0001*	0.0042*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Table 6: Average rankings of the compared algorithms on the 20-task WCCI20 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
WCCI20-MaTMO1	2.36	5.37	2.68	5.55	7.00	1.10	3.93
WCCI20-MaTMO2	5.42	6.39	5.93	2.34	3.43	1.38	3.11
WCCI20-MaTMO3	1.37	5.09	2.30	5.64	6.98	2.39	4.22
WCCI20-MaTMO4	1.83	3.22	2.08	5.76	6.08	4.69	4.34
WCCI20-MaTMO5	2.63	4.11	3.09	4.57	4.49	3.92	5.19
WCCI20-MaTMO6	1.85	3.67	2.17	5.54	5.59	4.53	4.65
WCCI20-MaTMO7	2.45	3.86	2.90	4.71	5.25	4.07	4.75
WCCI20-MaTMO8	2.69	3.85	2.87	5.11	5.09	3.58	4.83
WCCI20-MaTMO9	2.85	4.26	2.76	5.03	5.01	3.48	4.61
WCCI20-MaTMO10	2.42	4.18	2.67	5.43	5.15	3.52	4.62
Average rank	2.59	4.40	2.95	4.97	5.41	3.27	4.43
p-value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Table 7: Average rankings of the compared algorithms on the 50-task CEC19 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19-MaTMO1	2.22	5.17	2.77	5.77	6.99	1.11	3.96
CEC19-MaTMO2	1.19	3.90	1.81	3.76	6.19	6.79	4.35
CEC19-MaTMO3	4.21	4.62	6.71	4.25	4.67	2.48	1.06

(Continued)

Table 7 (continued)

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CECI9-MaTMO4	3.82	1.01	4.67	3.97	2.54	5.14	6.85
CECI9-MaTMO5	2.22	6.79	4.34	4.11	6.17	1.11	3.25
CECI9-MaTMO6	1.77	5.07	1.85	5.88	6.78	2.43	4.23
Average rank	2.57	4.43	3.69	4.62	5.56	3.18	3.95
p-value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*	0.0042*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Table 8: Average rankings of the compared algorithms on the 50-task WCCI20 benchmark suite.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
WCCI20-MaTMO1	2.32	5.37	2.74	5.56	6.99	1.10	3.91
WCCI20-MaTMO2	5.45	6.24	5.98	2.24	3.57	1.49	3.04
WCCI20-MaTMO3	1.36	4.97	2.46	5.70	6.99	2.32	4.20
WCCI20-MaTMO4	1.80	3.11	2.06	5.78	6.09	4.71	4.46
WCCI20-MaTMO5	2.60	4.11	3.07	4.64	4.56	3.79	5.22
WCCI20-MaTMO6	1.77	3.55	2.22	5.66	5.64	4.44	4.72
WCCI20-MaTMO7	2.61	3.83	3.04	4.57	5.23	3.98	4.74
WCCI20-MaTMO8	3.89	3.19	3.99	4.90	4.50	3.22	4.30
WCCI20-MaTMO9	3.19	3.73	3.77	4.96	4.57	3.31	4.46
WCCI20-MaTMO10	2.93	3.64	3.67	5.20	4.92	3.28	4.37

(Continued)

Table 8 (continued)

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
Average rank	2.79	4.17	3.30	4.92	5.31	3.16	4.34
p-value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*	<0.0001*

Note: EMaTO-SGN is used as the reference algorithm. * indicates a statistically significant difference from EMaTO-SGN according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best rank in each row is shown in bold.

Pairwise statistical comparisons are conducted using the Wilcoxon rank-sum test at $\alpha = 0.05$, with IGD-MTS results and comparison summaries reported in [Tables 9–11](#). Across all three task counts on CEC19, EMaTO-SGN shows no statistically significant losses against MO-EMaTO-MKT or MO-MaTDE. On the 20-task WCCI20 suite, EMaTO-SGN records eight significant wins and no significant losses against KR-MTEA, with no significant difference on the remaining two problems ([Table 10](#)). On the 50-task WCCI20 suite, it records seven significant wins and no significant losses against MO-EMaTO-MKT, with no significant difference on the remaining three problems ([Table 11](#)).

The IGD-MTS results support the overall competitiveness of EMaTO-SGN across the tested task counts, although its performance varies across problems. For example, MTDE-MKTA and MTEA-DCK achieve lower mean IGD-MTS values on CEC19-MaTMO3. Together, the average rankings and pairwise comparisons indicate strong overall performance, while showing that EMaTO-SGN does not outperform every comparison algorithm on every problem.

4.3.2 Convergence Curves and Pareto Front Approximations

[Fig. 3](#) presents the IGD convergence curves for four representative tasks using a logarithmic vertical axis. On WCCI20-MaTMO3-T17 ([Fig. 3c](#)) and CEC19-MaTMO2-T7 ([Fig. 3b](#)), EMaTO-SGN achieves low IGD values and remains competitive in the later stages of the search. On CEC19-MaTMO2-T5 ([Fig. 3a](#)), its IGD decreases sharply between approximately 25,000 and 30,000 evaluations, followed by more gradual improvement. On WCCI20-MaTMO8-T13 ([Fig. 3d](#)), EMaTO-SGN initially improves more slowly than several comparison algorithms but continues to reduce IGD in the later stages, achieving the lowest final IGD value among the compared methods.

[Fig. 4](#) compares the final non-dominated solution sets with the true Pareto fronts to assess convergence and coverage. On CEC19-MaTMO2-T7 ([Fig. 4a](#)), the solutions obtained by EMaTO-SGN closely follow the true front, whereas those produced by MO-SBO show a substantial deviation. On WCCI20-MaTMO3-T10 ([Fig. 4b](#)), EMaTO-SGN also produces a well-distributed approximation close to the true front.

Table 9: IGD-MTS results and Wilcoxon comparisons on the 10-task CEC19 and WCCI20 benchmark suites.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	CEC19 benchmark suite (10 tasks)			MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19-MaTMO1	-4.25e-01 (2.78e-04)	-3.52e-01 (7.67e-03)	-4.25e-01 (3.88e-04)	-1.11e-01 (2.69e-01)	-2.15e+00 (1.76e-01)	-4.26e-01 (1.83e-03)	-4.13e-01 (3.28e-03)	-	-
CEC19-MaTMO2	-7.50e-01 (5.43e-04)	-5.51e-01 (1.39e-02)	-7.06e-01 (6.09e-02)	-5.17e-01 (1.28e-01)	-1.17e+00 (1.43e-01)	1.82e+00 (7.58e-04)	-4.57e-01 (1.34e-01)	-	-
CEC19-MaTMO3	2.58e-01 (3.56e-01)	3.84e-01 (9.46e-02)	9.60e-01 (3.81e-02)	3.89e-01 (1.04e-01)	4.08e-01 (2.87e-01)	-5.89e-01 (1.70e-01)	-1.81e+00 (5.43e-02)	+	+
CEC19-MaTMO4	-2.76e-01 (6.16e-02)	-5.28e-01 (6.08e-01)	-1.72e-02 (1.43e-01)	-1.47e-01 (2.09e-01)	-6.10e-01 (8.56e-02)	1.55e-01 (1.29e-01)	1.43e+00 (2.24e-01)	-	-
CEC19-MaTMO5	-7.51e-01 (2.85e-02)	1.92e+00 (6.96e-02)	-1.74e-01 (6.89e-02)	-2.51e-01 (1.28e-01)	6.94e-01 (1.12e-01)	-9.45e-01 (3.50e-02)	-4.94e-01 (9.10e-02)	-	-
CEC19-MaTMO6	-8.70e-01 (9.02e-03)	4.28e-01 (8.47e-02)	-8.40e-01 (9.44e-03)	8.99e-01 (2.59e-01)	1.48e+00 (2.29e-01)	-8.12e-01 (3.00e-02)	-2.88e-01 (1.08e-01)	-	-
Summary (+ / - / =)	Reference	0/4/2	0/6/0	0/5/1	1/4/1	3/3/0	1/5/0		
WCCI20 benchmark suite (10 tasks)									
WCCI20-MaTMO1	-4.21e-01 (3.61e-04)	-3.71e-01 (4.07e-03)	-3.94e-01 (2.17e-02)	-3.22e-01 (6.39e-02)	2.34e+00 (2.07e-01)	-4.22e-01 (8.71e-03)	-4.11e-01 (4.16e-03)	-	-
WCCI20-MaTMO2	1.12e+00 (1.10e-01)	9.36e-01 (2.20e-01)	1.01e+00 (4.93e-02)	-8.18e-01 (1.05e-01)	-5.05e-01 (1.79e-01)	-1.18e+00 (5.66e-02)	-5.63e-01 (1.04e-01)	+	+
WCCI20-MaTMO3	-7.26e-01 (1.77e-02)	1.87e-02 (5.32e-02)	-6.59e-01 (2.24e-02)	3.59e-01 (1.55e-01)	2.07e+00 (2.40e-01)	-7.15e-01 (1.66e-02)	-3.46e-01 (6.50e-02)	-	-
WCCI20-MaTMO4	-6.40e-01 (4.51e-02)	-5.97e-01 (4.94e-02)	-6.50e-01 (3.39e-02)	4.18e-01 (1.98e-01)	1.07e+00 (5.33e-01)	3.49e-01 (1.71e-01)	4.68e-02 (9.80e-02)	-	-
WCCI20-MaTMO5	-3.14e-01 (1.80e-01)	-1.57e-01 (1.47e-01)	-1.42e-01 (5.00e-02)	-2.03e-02 (9.67e-02)	-2.08e-01 (8.21e-02)	2.76e-01 (2.31e-01)	5.65e-01 (1.60e-01)	-	-
WCCI20-MaTMO6	-6.92e-01 (8.53e-02)	-4.01e-01 (7.97e-02)	-6.84e-01 (6.08e-02)	4.75e-01 (2.25e-01)	4.66e-01 (2.68e-01)	6.44e-01 (5.28e-01)	1.92e-01 (9.03e-02)	-	-
WCCI20-MaTMO7	-3.32e-01 (1.34e-01)	-5.01e-01 (1.36e-01)	-2.05e-01 (7.39e-02)	1.66e-01 (1.72e-01)	6.34e-01 (1.16e-01)	5.94e-02 (3.84e-01)	1.80e-01 (1.27e-01)	-	-
WCCI20-MaTMO8	-4.90e-01 (1.38e-01)	-3.24e-01 (9.98e-02)	-3.83e-01 (1.47e-01)	3.57e-01 (3.03e-01)	6.32e-01 (1.64e-01)	3.63e-02 (2.08e-01)	1.72e-01 (1.92e-01)	-	-
WCCI20-MaTMO9	-4.14e-01 (8.24e-02)	2.58e-02 (1.85e-01)	-3.12e-01 (1.23e-01)	2.39e-01 (2.55e-01)	1.98e-01 (2.02e-01)	4.19e-03 (2.16e-01)	2.60e-01 (2.26e-01)	-	-
WCCI20-MaTMO10	-3.72e-01 (1.98e-01)	2.56e-02 (1.34e-01)	-2.79e-01 (1.30e-01)	1.43e-01 (2.12e-01)	4.86e-01 (3.25e-01)	-5.71e-02 (2.91e-01)	5.33e-02 (1.81e-01)	-	-
Summary (+ / - / =)	Reference	2/8/0	1/7/2	1/9/0	1/9/0	2/8/0	1/9/0		

Note: For each compared algorithm, the symbols +, -, and = indicate significantly better, significantly worse, and statistically similar performance relative to EMaTO-SGN, respectively, according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best mean value in each row is shown in bold.

Table 10: IGD-MTS results and Wilcoxon comparisons on the 20-task CEC19 and WCCI20 benchmark suites.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19 benchmark suite (20 tasks)							
CEC19-MaTMO1	-4.27e-01 (1.65e-04)	-3.65e-01 (3.42e-03)	-4.24e-01 (1.63e-03)	-8.46e-02 (1.96e-01)	2.14e+00 (1.77e-01)	-4.25e-01 (1.33e-02)	-4.16e-01 (2.66e-03)
CEC19-MaTMO2	-7.43e-01 (4.02e-04)	-5.63e-01 (7.42e-03)	-7.37e-01 (4.35e-03)	-5.06e-01 (6.21e-02)	1.22e+00 (1.27e-01)	1.79e+00 (1.85e-04)	-4.56e-01 (1.85e-01)
CEC19-MaTMO3	3.01e-02 (2.16e-01)	4.35e-01 (5.83e-02)	9.99e-01 (3.86e-02)	4.08e-01 (5.52e-02)	4.03e-01 (1.82e-01)	-5.12e-01 (1.18e-01)	-1.76e+00 (4.04e-02) +
CEC19-MaTMO4	-2.40e-01 (8.61e-02)	-1.13e+00 (1.21e-04) +	8.91e-02 (1.20e-01)	-5.07e-02 (2.00e-01)	-7.09e-01 (5.86e-02)	2.83e-01 (1.23e-01)	1.75e+00 (2.91e-01)
CEC19-MaTMO5	-7.48e-01 (2.28e-02)	1.90e+00 (5.75e-02)	-2.19e-01 (4.51e-02)	-2.70e-01 (7.47e-02)	7.78e-01 (7.59e-02)	-9.57e-01 (2.57e-02) +	-4.82e-01 (7.31e-02)
CEC19-MaTMO6	-8.58e-01 (7.39e-03)	3.59e-01 (7.60e-02)	-7.99e-01 (1.73e-02)	8.43e-01 (1.71e-01)	1.56e+00 (1.31e-01)	-7.96e-01 (1.84e-02)	-3.12e-01 (6.28e-02)
Summary (+ / - / =)	Reference	1/5/0	0/6/0	0/6/0	1/5/0	2/3/1	1/5/0
WCCI20 benchmark suite (20 tasks)							
WCCI20-MaTMO1	-4.18e-01 (1.51e-04)	-3.74e-01 (1.85e-03)	-4.17e-01 (2.20e-04)	-3.11e-01 (4.14e-02)	2.35e+00 (1.50e-01)	-4.19e-01 (2.90e-03) +	-4.08e-01 (2.42e-03)
WCCI20-MaTMO2	9.49e-01 (3.14e-02)	1.08e+00 (1.21e-01)	1.06e+00 (2.69e-02)	-8.63e-01 (8.39e-02)	-4.33e-01 (1.19e-01)	-1.19e+00 (4.97e-02) +	-6.05e-01 (8.15e-02)
WCCI20-MaTMO3	-7.31e-01 (5.32e-03)	-2.63e-03 (4.32e-02)	-6.71e-01 (1.21e-02)	3.28e-01 (1.13e-01)	2.11e+00 (1.46e-01)	-6.95e-01 (1.12e-02)	-3.34e-01 (4.90e-02)
WCCI20-MaTMO4	-5.80e-01 (4.91e-02)	-5.76e-01 (2.70e-02)	-5.98e-01 (3.75e-02) =	3.94e-01 (7.94e-02)	9.21e-01 (2.50e-01)	3.45e-01 (3.31e-01)	9.43e-02 (7.28e-02)
WCCI20-MaTMO5	-3.58e-01 (1.26e-01)	-2.17e-01 (1.09e-01)	-2.14e-01 (3.99e-02)	1.48e-02 (7.28e-02)	-7.94e-02 (4.69e-02)	3.12e-01 (4.92e-01)	5.42e-01 (9.65e-02)
WCCI20-MaTMO6	-6.51e-01 (7.13e-02)	-4.11e-01 (6.06e-02)	-6.65e-01 (6.04e-02) =	4.43e-01 (1.84e-01)	4.20e-01 (1.74e-01)	6.60e-01 (4.49e-01)	2.05e-01 (7.35e-02)
WCCI20-MaTMO7	-3.77e-01 (1.03e-01)	-4.00e-01 (1.12e-01) =	-2.89e-01 (2.12e-02)	1.57e-01 (2.07e-01)	6.59e-01 (9.28e-02)	-8.13e-02 (1.66e-01)	3.30e-01 (1.45e-01)
WCCI20-MaTMO8	-3.76e-01 (8.72e-02)	-2.31e-01 (8.68e-02)	-3.62e-01 (8.65e-02)	2.25e-01 (1.49e-01)	4.03e-01 (1.91e-01)	1.16e-01 (2.02e-01)	2.25e-01 (2.00e-01)
WCCI20-MaTMO9	-2.72e-01 (2.64e-01)	5.06e-02 (1.60e-01)	-3.01e-01 (7.37e-02) =	1.38e-01 (1.12e-01)	2.63e-01 (1.96e-01)	-8.19e-02 (1.43e-01)	2.03e-01 (1.89e-01)
WCCI20-MaTMO10	-3.37e-01 (9.97e-02)	-1.39e-01 (8.59e-02)	-3.49e-01 (5.94e-02) =	3.06e-01 (1.65e-01)	2.30e-01 (1.45e-01)	1.06e-01 (2.36e-01)	1.82e-01 (1.46e-01)
Summary (+ / - / =)	Reference	0/8/2	0/5/5	1/9/0	1/9/0	2/8/0	1/9/0

Note: For each compared algorithm, the symbols +, -, and = indicate significantly better, significantly worse, and statistically similar performance relative to EMaTO-SGN, respectively, according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best mean value in each row is shown in bold.

Table II: IGD-MTS results and Wilcoxon comparisons on the 50-task CEC19 and WCCI20 benchmark suites.

Problem	EMaTO-SGN	KR-MTEA	MO-EMaTO-MKT	MO-MaTDE	MO-SBO	MTDE-MKTA	MTEA-DCK
CEC19 benchmark suite (50 tasks)							
CEC19-MaTMO1	-4.28e-01 (9.33e-05)	-3.76e-01 (2.51e-03)	-4.27e-01 (3.23e-04)	-2.15e-01 (5.64e-02)	2.29e+00 (8.26e-02)	-4.29e-01 (6.27e-04) +	-4.16e-01 (1.46e-03)
CEC19-MaTMO2	-7.28e-01 (2.74e-04)	-5.68e-01 (4.93e-03)	-7.23e-01 (5.92e-04)	-5.11e-01 (5.80e-02)	1.32e+00 (8.44e-02)	1.71e+00 (9.09e-03)	-4.99e-01 (1.70e-02)
CEC19-MaTMO3	8.53e-02 (1.32e-01)	4.27e-01 (2.46e-02)	1.01e+00 (1.94e-02)	3.99e-01 (4.34e-02)	3.55e-01 (1.57e-01)	-5.17e-01 (8.99e-02)	-1.76e+00 (2.93e-02) +
CEC19-MaTMO4	-2.27e-01 (5.35e-02)	-1.13e+00 (2.37e-02) +	8.76e-02 (8.10e-02)	-1.43e-01 (1.26e-01)	-6.60e-01 (6.24e-02)	2.85e-01 (1.08e-01)	1.79e+00 (1.03e-01)
CEC19-MaTMO5	-7.44e-01 (1.45e-02)	1.85e+00 (2.84e-02)	-2.35e-01 (2.83e-02)	-2.72e-01 (4.92e-02)	8.68e-01 (4.46e-02)	-9.54e-01 (1.15e-02) +	-5.07e-01 (5.80e-02)
CEC19-MaTMO6	-8.28e-01 (1.52e-03)	2.15e-01 (2.70e-02)	-8.14e-01 (6.52e-03)	8.34e-01 (1.35e-01)	1.67e+00 (1.01e-01)	-7.82e-01 (1.54e-02)	-2.91e-01 (6.03e-02)
Summary (+ / - / =)	Reference	1/5/0	0/6/0	0/6/0	1/5/0	3/3/0	1/5/0
WCCI20 benchmark suite (50 tasks)							
WCCI20-MaTMO1	-4.14e-01 (8.63e-05)	-3.78e-01 (1.18e-03)	-4.13e-01 (1.69e-04)	-3.01e-01 (5.87e-02)	2.32e+00 (9.44e-02)	-4.14e-01 (7.07e-03)	-4.05e-01 (2.03e-03)
WCCI20-MaTMO2	9.60e-01 (1.86e-02)	1.01e+00 (1.09e-01)	1.07e+00 (2.19e-02)	-9.04e-01 (3.09e-02)	-3.31e-01 (8.32e-02)	-1.16e+00 (3.29e-02) +	-6.43e-01 (4.11e-02)
WCCI20-MaTMO3	-7.01e-01 (2.64e-03)	-8.72e-02 (1.87e-02)	-6.30e-01 (8.11e-03)	2.63e-01 (5.22e-02)	2.16e+00 (1.06e-01)	-6.67e-01 (7.34e-03)	-3.38e-01 (3.04e-02)
WCCI20-MaTMO4	-6.01e-01 (3.39e-02)	-5.96e-01 (1.97e-02)	-6.03e-01 (2.40e-02) =	4.39e-01 (9.39e-02)	9.26e-01 (1.43e-01)	2.96e-01 (1.32e-01)	1.39e-01 (6.71e-02)
WCCI20-MaTMO5	-3.58e-01 (7.67e-02)	-2.02e-01 (7.25e-02)	-2.48e-01 (3.58e-02)	4.74e-02 (3.49e-02)	-2.16e-02 (5.06e-02)	2.53e-01 (2.83e-01)	5.30e-01 (5.98e-02)
WCCI20-MaTMO6	-6.76e-01 (3.48e-02)	-4.24e-01 (4.06e-02)	-6.55e-01 (3.06e-02)	4.17e-01 (9.10e-02)	4.47e-01 (1.66e-01)	6.10e-01 (2.83e-01)	2.81e-01 (4.30e-02)
WCCI20-MaTMO7	-3.07e-01 (3.47e-02)	-3.74e-01 (7.12e-02) +	-2.73e-01 (4.94e-02)	8.41e-02 (6.19e-02)	6.84e-01 (5.04e-02)	-1.10e-01 (1.12e-01)	2.96e-01 (1.15e-01)
WCCI20-MaTMO8	-1.73e-02 (2.32e-01)	-2.76e-01 (7.17e-02) +	6.90e-02 (1.06e-01)	1.07e-01 (8.07e-02)	1.40e-02 (1.26e-01)	-3.63e-02 (1.18e-01)	1.40e-01 (1.09e-01)
WCCI20-MaTMO9	-2.82e-01 (2.15e-01)	-1.08e-01 (6.77e-02)	-2.18e-02 (8.57e-02)	1.99e-01 (1.03e-01)	6.78e-02 (9.10e-02)	-3.72e-02 (1.05e-01)	1.82e-01 (8.25e-02)
WCCI20-MaTMO10	-3.18e-01 (1.53e-01)	-1.84e-01 (3.55e-02)	-4.93e-02 (1.08e-01)	2.23e-01 (9.91e-02)	1.91e-01 (1.00e-01)	4.61e-02 (1.97e-01)	9.17e-02 (6.07e-02)
Summary (+ / - / =)	Reference	2/6/2	0/7/3	1/9/0	1/8/1	1/8/1	1/9/0

Note: For each compared algorithm, the symbols +, -, and = indicate significantly better, significantly worse, and statistically similar performance relative to EMaTO-SGN, respectively, according to the Wilcoxon rank-sum test at $\alpha = 0.05$. The best mean value in each row is shown in bold.

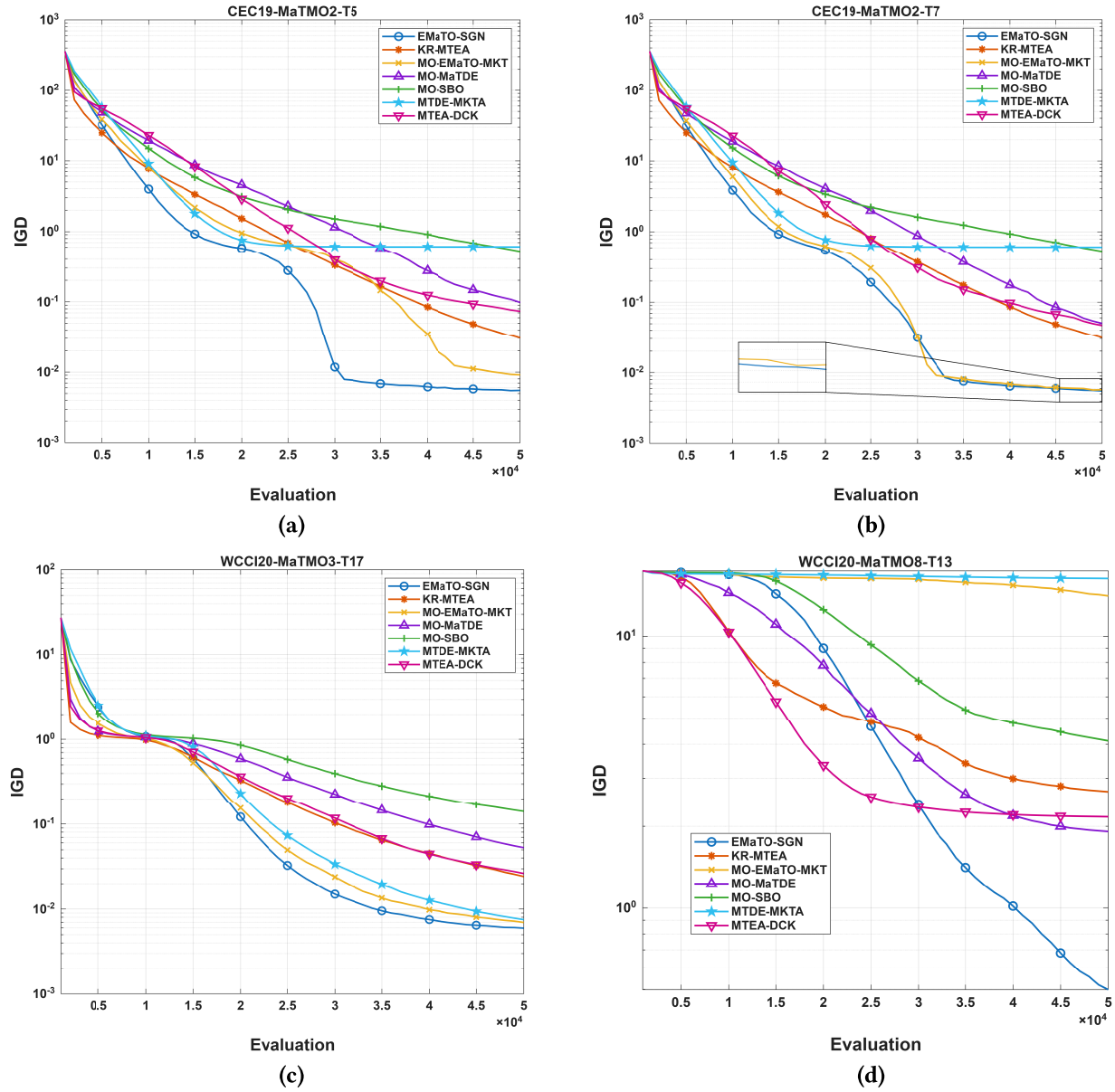


Figure 3: Comparison of the IGD convergence curves. (a) Performance on task CEC19-MaTMO2-T5; (b) Performance on task CEC19-MaTMO2-T7; (c) Performance on task WCCI20-MaTMO3-T17; (d) Performance on task WCCI20-MaTMO8-T13.

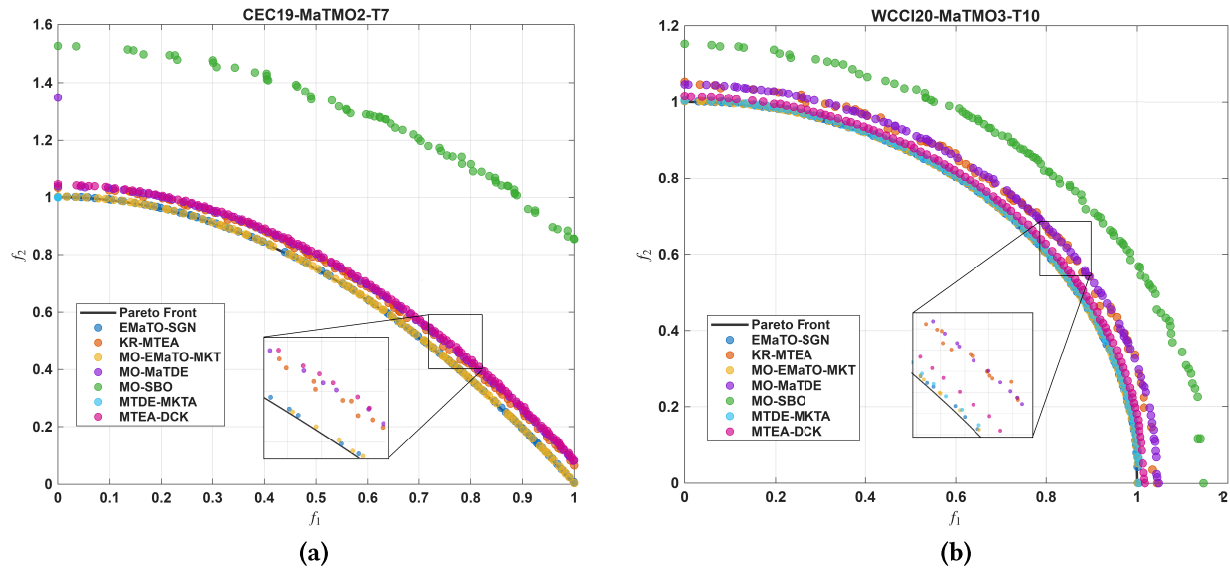


Figure 4: Final distribution and front coverage of the non-dominated solution sets compared against the true Pareto front. (a) Solution sets on task CEC19-MaTMO2-T7; (b) Solution sets on task WCCI20-MaTMO3-T10.

4.3.3 Performance Comparison with the EMT-GS Algorithm

Table 12 compares the IGD results of EMaTO-SGN and EMT-GS on CEC19-MaTMO1 with 2, 4, 6, 8, and 10 tasks. EMaTO-SGN achieves lower mean IGD values on every reported task. In the 10-task scenario, the mean IGD averaged across tasks is approximately 5.22×10^{-3} for EMaTO-SGN, compared with 5.51×10^1 for EMT-GS. EMaTO-SGN also exhibits low variability across independent runs: the per-task standard deviations in this setting do not exceed 3.67×10^{-4} .

Table 12: Mean IGD results of EMaTO-SGN and EMT-GS across different task scales.

Benchmark Instance	EMaTO-SGN	EMT-GS
2-task scenario		
CEC19-MaTMO1-T1	5.50e-03 (3.11e-04)	4.90e-02 (3.09e-02)
CEC19-MaTMO1-T2	5.52e-03 (3.51e-04)	3.34e-02 (1.49e-02)
4-task scenario		
CEC19-MaTMO1-T1	5.20e-03 (2.03e-04)	4.15e+00 (3.37e+00)
CEC19-MaTMO1-T2	5.31e-03 (2.85e-04)	4.29e+00 (3.65e+00)
CEC19-MaTMO1-T3	5.29e-03 (2.59e-04)	2.90e+00 (2.71e+00)
CEC19-MaTMO1-T4	5.28e-03 (1.82e-04)	1.90e+00 (9.63e-01)
6-task scenario		
CEC19-MaTMO1-T1	5.12e-03 (2.13e-04)	1.92e+01 (9.48e+00)
CEC19-MaTMO1-T2	5.15e-03 (3.48e-04)	1.49e+01 (4.04e+00)
CEC19-MaTMO1-T3	5.20e-03 (2.27e-04)	1.68e+01 (1.02e+01)
CEC19-MaTMO1-T4	5.12e-03 (1.66e-04)	9.16e+00 (2.15e+00)
CEC19-MaTMO1-T5	5.29e-03 (4.39e-04)	2.19e+01 (5.36e+00)
CEC19-MaTMO1-T6	5.16e-03 (2.62e-04)	1.27e+01 (6.81e+00)

(Continued)

Table 12 (continued)

Benchmark Instance	EMaTO-SGN	EMT-GS
8-task scenario		
CEC19-MaTMO1-T1	5.14e-03 (3.02e-04)	4.35e+01 (5.99e+00)
CEC19-MaTMO1-T2	5.22e-03 (1.84e-04)	3.37e+01 (1.41e+01)
CEC19-MaTMO1-T3	5.14e-03 (4.58e-04)	3.77e+01 (8.46e+00)
CEC19-MaTMO1-T4	5.19e-03 (2.61e-04)	2.65e+01 (6.93e+00)
CEC19-MaTMO1-T5	5.27e-03 (3.27e-04)	2.51e+01 (9.29e+00)
CEC19-MaTMO1-T6	5.15e-03 (1.89e-04)	2.68e+01 (1.56e+01)
CEC19-MaTMO1-T7	5.43e-03 (4.39e-04)	2.79e+01 (1.16e+01)
CEC19-MaTMO1-T8	5.16e-03 (1.92e-04)	2.61e+01 (7.01e+00)
10-task scenario		
CEC19-MaTMO1-T1	5.21e-03 (3.67e-04)	9.37e+01 (1.88e+01)
CEC19-MaTMO1-T2	5.31e-03 (2.22e-04)	6.34e+01 (9.63e+00)
CEC19-MaTMO1-T3	5.25e-03 (1.61e-04)	5.04e+01 (1.27e+01)
CEC19-MaTMO1-T4	5.25e-03 (1.25e-04)	2.93e+01 (2.86e+00)
CEC19-MaTMO1-T5	5.20e-03 (2.38e-04)	5.41e+01 (2.26e+01)
CEC19-MaTMO1-T6	5.25e-03 (3.16e-04)	5.36e+01 (1.97e+01)
CEC19-MaTMO1-T7	5.16e-03 (2.19e-04)	4.16e+01 (5.66e+00)
CEC19-MaTMO1-T8	5.12e-03 (1.72e-04)	4.89e+01 (5.58e+00)
CEC19-MaTMO1-T9	5.21e-03 (2.33e-04)	6.65e+01 (1.52e+01)
CEC19-MaTMO1-T10	5.23e-03 (1.60e-04)	4.95e+01 (2.01e+01)

Note: Lower values indicate better performance. The lower mean value in each row is shown in bold.

Table 13 and Fig. 5a compare computational time as the number of tasks increases. EMT-GS requires 27.20 s for 2 tasks, 1125.30 s for 10 tasks, and 3762.50 s for 20 tasks. The corresponding runtimes of EMaTO-SGN are 12.31, 41.75, and 93.47 s, yielding speedup factors of approximately 27 and 40 in the 10- and 20-task settings, respectively. Together, these results show improved solution quality in the reported IGD comparisons and lower runtime wherever measurements are available for both algorithms.

Table 13: Runtime of EMaTO-SGN and EMT-GS across different task scales.

Number of Tasks	EMaTO-SGN (s)	EMT-GS (s)
2	12.31 (2.53e-01)	27.20 (2.58e-01)
4	19.36 (3.45e-01)	117.49 (1.01e+00)
6	26.17 (1.97e-01)	288.51 (1.23e+00)
8	32.59 (2.75e-01)	567.25 (5.60e+00)
10	41.75 (2.59e-01)	1125.30 (3.78e+00)
15	66.50 (3.56e-01)	2162.50 (5.26e+00)
20	93.47 (4.25e-01)	3762.50 (4.79e+00)
30	155.05 (6.81e-01)	N/A*
40	227.14 (1.46e+00)	N/A*
50	316.19 (1.73e+00)	N/A*

Note: For settings in which both algorithms returned results, the lower mean runtime is shown in bold. * Runtime results for EMT-GS were unavailable because the required computation time was prohibitively long.

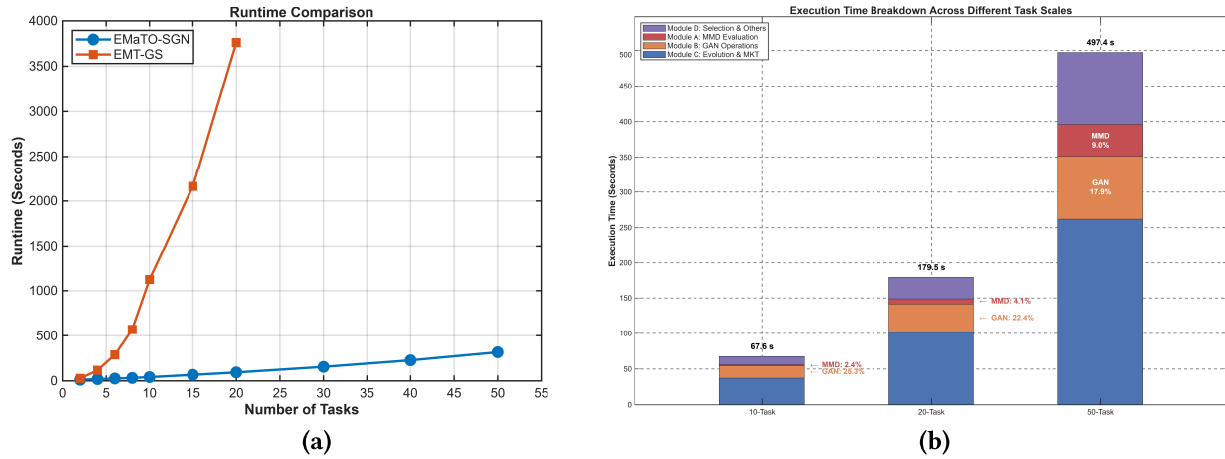


Figure 5: Evaluation of computational efficiency. (a) Total runtime comparison between the proposed EMaTO-SGN and EMT-GS; (b) Breakdown of execution time for different components of EMaTO-SGN across different task scales.

4.3.4 Ablation Study

Ablation experiments on CEC19-MaTMO1 and WCCI20-MaTMO1 compare the full EMaTO-SGN model with four variants: w/o DTS (replaces dynamic task scoring with random auxiliary-task selection), w/o ATg (disables adaptive triggering and activates knowledge transfer in every generation), w/o GKT (replaces GAN-based mapping with direct crossover), and w/o KT (removes the knowledge transfer entirely, degrading to an independent single-task optimizer).

As shown in Table 14, the Full Model achieves a top Friedman ranking of 1.82. Removing the knowledge transfer entirely (w/o KT, rank 4.97) performs the worst across all test instances, supporting the benefit of cross-task knowledge transfer on these benchmarks. The w/o DTS and w/o ATg variants obtain Friedman ranks of 2.60 and 2.76, respectively, compared with 1.82 for the full model. These results suggest that source-task selection and transfer timing contribute to overall optimization performance.

Table 14: IGD results, Friedman ranks, and Wilcoxon comparisons for the EMaTO-SGN variants.

Benchmark Instance	Full Model	w/o DTS	w/o ATg	w/o GKT	w/o KT
CEC19 benchmark suite					
CEC19-MaTMO1-T1	5.56e-03 (5.41e-04)	5.58e-03 (3.30e-04)	5.58e-03 (3.30e-04)	5.59e-03 (3.68e-04)	7.46e-03 (4.29e-04)
CEC19-MaTMO1-T2	6.12e-03 (1.82e-03)	5.54e-03 (3.03e-04)	5.67e-03 (1.68e-04)	5.63e-03 (2.73e-04)	7.46e-03 (4.36e-04)
CEC19-MaTMO1-T3	5.38e-03 (2.68e-04)	5.62e-03 (2.58e-04)	5.68e-03 (2.80e-04)	5.62e-03 (3.23e-04)	7.45e-03 (3.42e-04)
CEC19-MaTMO1-T4	5.35e-03 (3.36e-04)	5.57e-03 (3.02e-04)	5.74e-03 (3.59e-04)	5.72e-03 (3.14e-04)	7.46e-03 (3.91e-04)

(Continued)

Table 14 (continued)

Benchmark Instance	Full Model	w/o DTS	w/o ATg	w/o GKT	w/o KT
CECI9-MaTMOI-T5	5.72e-03 (1.62e-03)	5.65e-03 (3.03e-04)	5.71e-03 (2.22e-04)	5.62e-03 (2.98e-04)	7.36e-03 (6.08e-04)
CECI9-MaTMOI-T6	5.49e-03 (2.90e-04)	5.61e-03 (2.11e-04)	5.61e-03 (2.03e-04)	5.76e-03 (2.98e-04)	7.75e-03 (6.39e-04)
CECI9-MaTMOI-T7	5.42e-03 (3.52e-04)	5.71e-03 (3.03e-04)	5.66e-03 (2.73e-04)	5.72e-03 (3.07e-04)	7.39e-03 (3.72e-04)
CECI9-MaTMOI-T8	5.83e-03 (2.06e-03)	5.59e-03 (2.18e-04)	5.66e-03 (4.01e-04)	5.64e-03 (2.24e-04)	7.44e-03 (5.96e-04)
CECI9-MaTMOI-T9	5.34e-03 (4.21e-04)	5.65e-03 (3.31e-04)	5.63e-03 (3.48e-04)	5.78e-03 (3.34e-04)	7.63e-03 (7.43e-04)
CECI9-MaTMOI-T10	6.81e-03 (6.24e-03)	5.59e-03 (2.64e-04)	5.75e-03 (2.59e-04)	5.55e-03 (1.71e-04)	7.61e-03 (4.42e-04)
WCCI20 benchmark suite					
WCCI20-MaTMOI-T1	5.38e-03 (4.43e-04)	5.56e-03 (2.16e-04)	5.73e-03 (2.39e-04)	5.75e-03 (2.06e-04)	7.33e-03 (4.94e-04)
WCCI20-MaTMOI-T2	5.39e-03 (3.85e-04)	5.55e-03 (2.65e-04)	5.75e-03 (3.76e-04)	5.73e-03 (2.26e-04)	7.66e-03 (4.81e-04)
WCCI20-MaTMOI-T3	5.50e-03 (3.63e-04)	5.52e-03 (1.95e-04)	5.69e-03 (3.22e-04)	5.66e-03 (3.30e-04)	7.51e-03 (3.71e-04)
WCCI20-MaTMOI-T4	5.40e-03 (2.29e-04)	5.78e-03 (3.24e-04)	5.57e-03 (2.39e-04)	5.66e-03 (2.58e-04)	7.44e-03 (5.05e-04)
WCCI20-MaTMOI-T5	5.40e-03 (2.96e-04)	5.66e-03 (2.34e-04)	5.68e-03 (3.30e-04)	5.65e-03 (3.19e-04)	7.24e-03 (4.44e-04)
WCCI20-MaTMOI-T6	5.55e-03 (1.25e-03)	5.67e-03 (2.22e-04)	5.58e-03 (1.87e-04)	5.66e-03 (2.47e-04)	7.39e-03 (4.47e-04)
WCCI20-MaTMOI-T7	5.42e-03 (3.92e-04)	5.68e-03 (3.15e-04)	5.53e-03 (1.86e-04)	5.74e-03 (3.64e-04)	7.42e-03 (5.51e-04)
WCCI20-MaTMOI-T8	5.34e-03 (3.16e-04)	5.67e-03 (3.60e-04)	5.77e-03 (4.45e-04)	5.57e-03 (2.57e-04)	7.67e-03 (5.99e-04)
WCCI20-MaTMOI-T9	5.39e-03 (2.74e-04)	5.75e-03 (3.54e-04)	5.58e-03 (2.73e-04)	5.81e-03 (4.56e-04)	7.56e-03 (5.75e-04)

(Continued)

Table 14 (continued)

Benchmark Instance	Full Model	w/o DTS	w/o ATg	w/o GKT	w/o KT
WCCI20-MaTMO1-T10	5.45e-03 (4.60e-04)	5.56e-03 (1.70e-04)	5.72e-03 (4.41e-04)	5.89e-03 (3.42e-04)	7.44e-03 (6.28e-04)
Friedman rank	1.82	2.60	2.76	2.83	4.97
<i>p</i> -value	Reference	<0.0001*	<0.0001*	<0.0001*	<0.0001*

Note: The lowest mean value in each row and the best Friedman rank are shown in bold. * indicates a statistically significant difference from the full model at $\alpha = 0.05$.

Taken together, the ablation results support the combined use of dynamic task scoring, adaptive triggering, and generative mapping. Task scoring uses similarity and historical transfer performance to guide source-task selection, while adaptive triggering regulates transfer according to evolutionary progress. These mechanisms are intended to limit unsuitable or unnecessary transfers. The observed performance differences are consistent with this rationale, but do not directly establish the frequency or causes of negative transfer.

4.3.5 Parameter Sensitivity Analysis

We conduct a parameter sensitivity analysis to examine the effects of four key parameters and assess the selected default settings. We focus on four critical parameters: the GAN resource quota (p_{gan}), the whitelist capacity (N_{src}), the base stagnation threshold (ϵ_{stag}), and the soft buffer multiplier (B). Table 15 shows the results.

Table 15: Parameter sensitivity analysis for EMaTO-SGN.

Parameter Setting	IGD-MTS	Friedman Rank	<i>p</i> -Value
GAN resource quota (p_{gan})			
$p_{\text{gan}} = 0.1$	-5.35e-01 (5.43e-03)	2.00	0.0168*
$p_{\text{gan}} = 0.2$	-5.48e-01 (4.25e-03)	1.00	Reference
$p_{\text{gan}} = 0.4$	-5.05e-01 (7.23e-03)	3.00	<0.0001*
$p_{\text{gan}} = 0.6$	-4.21e-01 (8.16e-03)	4.00	<0.0001*
$p_{\text{gan}} = 0.8$	-1.50e-01 (1.67e-02)	5.00	<0.0001*
$p_{\text{gan}} = 1.0$	2.16e+00 (8.43e-02)	6.00	<0.0001*
Whitelist capacity (N_{src})			
$N_{\text{src}} = 2$	-2.01e-01 (1.03e-01)	3.40	0.0225*
$N_{\text{src}} = 4$	-2.70e-01 (9.37e-02)	2.55	0.0398*
$N_{\text{src}} = 5$	-2.87e-01 (1.11e-01)	2.30	0.0673
$N_{\text{src}} = 6$	-3.14e-01 (1.20e-01)	2.05	Reference
$N_{\text{src}} = 8$	-4.35e-02 (7.86e-02)	4.70	<0.0001*
$N_{\text{src}} = 10$	1.12e+00 (1.57e-01)	6.00	<0.0001*
Base stagnation threshold (ϵ_{stag})			
$\epsilon_{\text{stag}} = 0.01$	8.44e-01 (1.04e-01)	7.00	<0.0001*
$\epsilon_{\text{stag}} = 0.05$	3.00e-01 (1.48e-01)	5.95	<0.0001*

(Continued)

Table 15 (continued)

Parameter Setting	IGD-MTS	Friedman Rank	p-Value
$\varepsilon_{\text{stag}} = 0.10$	-1.49e-01 (1.36e-01)	3.90	0.0404*
$\varepsilon_{\text{stag}} = 0.15$	-2.12e-01 (1.37e-01)	3.20	0.0306*
$\varepsilon_{\text{stag}} = 0.20$	-2.71e-01 (9.92e-02)	2.50	Reference
$\varepsilon_{\text{stag}} = 0.30$	-2.57e-01 (1.15e-01)	2.70	0.0397*
$\varepsilon_{\text{stag}} = 0.35$	-2.56e-01 (1.61e-01)	2.75	0.0314*
Soft buffer multiplier (B)			
$B = 1$	9.64e-02 (1.14e-01)	5.40	0.0034*
$B = 2$	-3.82e-02 (1.59e-01)	3.90	0.0464*
$B = 3$	-2.24e-02 (1.29e-01)	3.50	0.0484*
$B = 4$	-4.71e-02 (1.20e-01)	3.40	Reference
$B = 5$	-2.45e-02 (1.22e-01)	3.40	1.0000
$B = 6$	5.75e-02 (1.63e-01)	4.85	0.0338*
$B = 7$	-2.17e-02 (1.53e-01)	3.55	0.0482*

Note: The adopted default setting is used as the reference for the Wilcoxon rank-sum tests and is shown in bold. * indicates a statistically significant difference from the corresponding reference setting at $\alpha = 0.05$.

The parameter p_{gan} determines the proportion of individuals assigned to the generative mapping route when the transfer state is activated. The tested values are $\{0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Among these settings, $p_{\text{gan}} = 0.2$ achieves the best Friedman ranking of 1.00. Notably, setting $p_{\text{gan}} = 1.0$, which assigns all individuals to the generative route when transfer is active, results in the worst rank of 6.00.

The N_{src} parameter limits the number of candidate source tasks in each whitelist. The tested range was set to $\{2, 4, 5, 6, 8, 10\}$. Among the tested whitelist capacities, $N_{\text{src}} = 6$ achieves the lowest IGD-MTS. When $N_{\text{src}} = 10$, IGD-MTS increases to 1.12, indicating a substantial decline in performance. Expanding the source-task pool does not necessarily improve transfer quality.

The threshold $\varepsilon_{\text{stag}}$ defines the lower boundary of the transfer-probability transition region. Together with B , it determines how the trigger probability responds to relative convergence improvement. We tested $\varepsilon_{\text{stag}}$ over the set $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.3, 0.35\}$. The best performance is observed at $\varepsilon_{\text{stag}} = 0.2$. Larger thresholds make transfer activation more permissive, which may explain the poorer performance observed at these settings.

The buffer multiplier B controls the width of the transition region over which the transfer probability decreases linearly from 1 to 0. Integer values from 1 to 7 are tested. When $B = 1$, the transition region vanishes, yielding the hard-threshold variant described in Section 3.4, which has a Friedman rank of 5.40. Values between 3 and 5 yield relatively stable performance, with $B = 4$ adopted as the default setting.

4.3.6 Computational Efficiency and Execution Time Profiling

The wall-clock runtime of a complete run is divided into four components: Module A: MMD Evaluation (similarity matrix construction), Module B: GAN Operations (adversarial training and generative transfer), Module C: Evolution & MKT (native evolutionary operators and multi-source classic knowledge transfer), and Module D: Others (environmental selection and evaluation).

As shown in Fig. 5b on the 50-task test suite, MMD evaluation accounts for 9.01% of the total runtime in the 50-task setting, requiring 44.80 s out of 497.39 s. Updating the similarity matrix every $G_{\text{sim}} = 20$ generations reduces the frequency of pairwise comparisons and helps control their practical computational cost. Although this periodic schedule reduces the amortized overhead, a full similarity update retains a quadratic dependence on the number of tasks.

The results demonstrate that the execution time is not dominated by the advanced transfer mechanisms. The GAN operations (Module B) stably consume approximately 17.94% of the computational cost. The actual time cost fundamentally lies in Module C (52.68%). These results show that similarity evaluation and generative modeling do not dominate execution time under the profiled conditions.

5 Discussion

This section discusses the mechanisms that may explain the observed performance of EMaTO-SGN and examines the trade-offs between computational cost and optimization performance.

5.1 Mechanism of WGAN-GP Mapping

EMaTO-SGN uses source-task decision vectors as generator inputs, providing a direct way to incorporate information from previously explored solutions into generative knowledge transfer. These inputs carry information about the source population's distribution and geometric structure. The generator learns a nonlinear mapping from source-task solutions to candidate solutions in the target task's search space. By operating on previously identified high-quality solutions, the mapping can exploit existing search information and provide structural guidance for candidate generation. This design may facilitate the reuse of useful solution features across tasks, although its effects on generated-solution diversity and training stability requires separate evaluation.

WGAN-GP provides a framework for learning cross-task mappings through adversarial training with gradient regularization. However, the performance difference between EMaTO-SGN and EMT-GS cannot be attributed solely to WGAN-GP, because the algorithms also differ in their transfer architectures and scheduling mechanisms. In EMaTO-SGN, this mapping is intended to retain useful features from source-task solutions while adapting them to the target task's search space. The ablation results in Section 4.3.4 support the contribution of generative mapping to overall optimization performance.

5.2 Trade-off between Computational Time and Evolutionary Performance

GAN training and offspring generation introduce additional computational costs in EMaTO-SGN. In many-task optimization, frequent network updates and repeated generative operations across tasks can increase the overall runtime. EMaTO-SGN therefore seeks to balance the computational cost of generative knowledge transfer with its potential benefits for optimization performance.

The runtime comparisons and ablation results suggest that EMaTO-SGN can benefit from generative knowledge transfer while keeping its computational overhead manageable under the tested settings. To control training costs, the algorithm uses a 1:1 critic-to-generator update ratio and sets the training batch size equal to the population size. These choices limit the number of network updates and the volume of data processed in each batch. Periodic training allows the models to incorporate updated population information without retraining from scratch, providing a practical compromise between model adaptation and computational cost.

By selectively activating knowledge transfer and periodically updating the generative models, EMaTO-SGN aims to balance optimization performance against the additional computational cost of generative learning.

5.3 Comparative Analysis of Adaptive Transfer Mechanisms

EMaTSaNES [42], proposed by Zhao et al., provides a relevant comparison for adaptive transfer design. Its Adaptive Knowledge Transfer and Sharing (AKTS) mechanism determines a continuous transfer probability from recent fitness trends. A Hybrid Resource Sampling (HRS) module switches among probability distributions to promote search diversity.

These mechanisms adapt transfer and sampling behavior during the search. EMaTO-SGN combines a bounded transition region for transfer activation with a fixed generative transfer allocation ratio, p_{gan} . The trigger probability decreases linearly within the transition region and becomes zero when relative convergence improvement reaches the upper threshold. This rule limits transfer when the target task is making sufficient progress independently. When transfer is activated, p_{gan} controls the proportion of individuals assigned to generative mapping, with the remainder assigned to conventional knowledge transfer.

6 Conclusion

This paper presents EMaTO-SGN, an evolutionary many-task multi-objective optimization framework designed to mitigate negative transfer and reduce the computational cost of generative knowledge transfer. By assigning one generative model to each task, EMaTO-SGN makes the cost of generative model training and sample generation scale linearly with the number of tasks when the remaining parameters are fixed. Pairwise similarity evaluation still introduces a quadratic component into the overall complexity. Source-task whitelists are constructed using both inter-task similarity and historical transfer success. An adaptive trigger regulates transfer activation according to evolutionary progress, balancing cross-task exploration with task-specific exploitation.

Experiments on the CEC19 and WCCI20 benchmark suites with up to 50 tasks demonstrate the overall competitiveness of EMaTO-SGN. It achieves the best overall average rank in all six combinations of benchmark suite and task count, although its IGD-MTS performance varies across individual problems. Moreover, EMaTO-SGN shows a clear advantage over the original EMT-GS by effectively mitigating the computational “multitasking curse”, achieving remarkable runtime acceleration. The results of 10-task scenario tests show that this framework can achieve much lower IGD values and much shorter runtime than EMT-GS. The runtime comparisons show speedup factors of approximately 27 and 40 over EMT-GS in the 10- and 20-task settings, respectively. Ablation studies support the contributions of dynamic task scoring, adaptive triggering, and generative mapping to overall performance.

While this study focuses predominantly on the theoretical design of the EMaTO-SGN framework and its validation on continuous benchmark suites, the ultimate objective of evolutionary many-tasking is to tackle complex, real-world problems. In future work, we plan to extend and adapt the proposed generative knowledge transfer mechanisms to highly constrained, practical applications. Promising and impactful directions include robust facility location and allocation in supply chain networks under fuzzy and random uncertainties [43], as well as enhancing the computational efficiency of large-scale linear optimization models [44]. Deploying EMaTO-SGN to navigate the rugged fitness landscapes of these challenging industrial and discrete scenarios represents a significant next step.

Acknowledgement: Not applicable.

Funding Statement: This work was supported by the National Natural Science Foundation of China (Grant No. 62303197).

Author Contributions: Concept and design: Xu Chen and Haogang Li; software, validation and visualization: Haogang Li; statistical analysis: Haogang Li; supervision: Xu Chen; writing—original draft: Haogang Li; writing—review & editing: Xu Chen; project administration and resources: Xu Chen; All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: Given his role as Editorial Board Member of this journal, Xu Chen had no involvement in the peer review or editorial handling of this article and had no access to information concerning its peer review. Full responsibility for the editorial process was delegated to another independent journal editor. The authors declare no other conflicts of interest.

References

1. Zhang Q, Li H. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans Evol Comput.* 2007;11(6):712–31. doi:10.1109/tevc.2007.892759.
2. Gupta A, Ong YS, Feng L. Multifactorial evolution: toward evolutionary multitasking. *IEEE Trans Evol Comput.* 2016;20(3):343–57. doi:10.1109/TEVC.2015.2458037.
3. Ong YS, Gupta A. Evolutionary multitasking: a computer science view of cognitive multitasking. *Cogn Comput.* 2016;8(2):125–42. doi:10.1007/s12559-016-9395-7.
4. Bali KK, Ong YS, Gupta A, Tan PS. Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II. *IEEE Trans Evol Comput.* 2020;24(1):69–83. doi:10.1109/TEVC.2019.2906927.
5. Feng L, Zhou L, Zhong J, Gupta A, Ong YS, Tan KC, et al. Evolutionary multitasking via explicit autoencoding. *IEEE Trans Cybern.* 2019;49(9):3457–70. doi:10.1109/TCYB.2018.2845361.
6. Liang ZP, Zhu YM, Wang XY, Li Z, Zhu ZX. Evolutionary multitasking for optimization based on generative strategies. *IEEE Trans Evol Comput.* 2023;27(4):1042–56. doi:10.1109/TEVC.2022.3189029.
7. Coello CAC, Lamont GB, Van Veldhuizen DA. *Evolutionary algorithms for solving multi-objective problems*. New York, NY, USA: Springer; 2007.
8. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput.* 2002;6(2):182–97. doi:10.1109/4235.996017.
9. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial networks. *Commun ACM.* 2020;63(11):139–44. doi:10.1145/3422622.
10. Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. *Proc Mach Learn Res.* 2017;70(1):214–23. doi:10.1111/rssb.12476.
11. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC. Improved training of wasserstein GANs. *arXiv:1704.00028*. 2017. doi:10.48550/arXiv.1704.00028.
12. Li SJ, Li YC, Wang R, Gong WY, Zhang TY, Wang L. Survey on evolutionary multitask optimization: technical advances, problem classification, and applications. *Control Decis.* 2026;41(4):987–1004. (In Chinese). doi:10.13195/j.kzyjc.2025.0564.
13. Tan KC, Feng L, Jiang M. Evolutionary transfer optimization—a new frontier in evolutionary computation research. *IEEE Comput Intell Mag.* 2021;16(1):22–33. doi:10.1109/MCI.2020.3039066.
14. Tang J, Chen Y, Deng Z, Xiang Y, Joy CP. A group-based approach to improve multifactorial evolutionary algorithm. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*; 2018 Jul 13–19; Stockholm, Sweden. p. 3870–6. doi:10.24963/ijcai.2018/538.

15. Gong MG, Tang ZD, Li H, Zhang J. Evolutionary multitasking with dynamic resource allocating strategy. *IEEE Trans Evol Comput.* 2019;23(5):858–69. doi:10.1109/TEVC.2019.2893614.
16. Gupta A, Mandziuk J, Ong YS. Evolutionary multitasking in bi-level optimization. *Complex Intell Syst.* 2015;1:83–95. doi:10.1007/s40747-016-0011-y.
17. Chen YL, Zhong JH, Feng L, Zhang J. An adaptive archive-based evolutionary framework for many-task optimization. *IEEE Trans Emerg Top Comput Intell.* 2020;4(3):369–84. doi:10.1109/TETCI.2019.2916051.
18. Zhou XY, Mei N, Zhong MS, Wang MW. Evolutionary multi-task optimization with adaptive intensity of knowledge transfer. *IEEE Trans Emerg Top Comput Intell.* 2024. doi:10.1109/tetci.2024.3418810.
19. Thanh LT, Van Cuong L, Thang TB, Thi Thanh Binh H. Multi-armed bandits for many-task evolutionary optimization. In: *Proceedings of the 2021 IEEE Congress on Evolutionary Computation (CEC); 2021 Jun 28– Jul 1; Virtual.* p. 1664–71. doi:10.1109/CEC45853.2021.9504691.
20. Fialho Á., Da Costa L, Schoenauer M, Sebag M. Dynamic multi-armed bandits and extreme value-based rewards for adaptive operator selection in evolutionary algorithms. In: *International Conference on Learning and Intelligent Optimization.* Berlin/Heidelberg, Germany: Springer; 2009. p. 176–90. doi:10.1007/978-3-642-11169-3_13.
21. Li YC, Li DC, Gong WY, Gu Q. Multiobjective multitask optimization via diversity- and convergence-oriented knowledge transfer. *IEEE Trans Syst Man Cybern Syst.* 2025;55(3):2367–79. doi:10.1109/TSMC.2024.3520526.
22. Li YC, Gong WY. Multiobjective multitask optimization with multiple knowledge types and transfer adaptation. *IEEE Trans Evol Comput.* 2025;29(1):205–16. doi:10.1109/tevc.2024.3353319.
23. Liang ZP, Xu XJ, Liu L, Tu YF, Zhu ZX. Evolutionary many-task optimization based on multisource knowledge transfer. *IEEE Trans Evol Comput.* 2022;26(2):319–33. doi:10.1109/TEVC.2021.3101697.
24. Dai YT, Liu XF, Fang YC, Zhan ZH, Zhang J. Multisource knowledge fusion based on graph attention networks for many-task optimization. *IEEE Trans Evol Comput.* 2025;29(5):2116–30. doi:10.1109/tevc.2024.3465542.
25. Wu SH, Zhan ZH, Tan KC, Zhang J. Transferable adaptive differential evolution for many-task optimization. *IEEE Trans Cybern.* 2023;53(11):7295–308. doi:10.1109/TCYB.2023.3234969.
26. Zhang TY, Wu XY, Li YC, Gong WY, Qin H. Distribution direction-assisted two-stage knowledge transfer for many-task optimization. *IEEE Trans Syst Man Cybern Syst.* 2025;55(10):7551–65. doi:10.1109/tsmc.2025.3598800.
27. Jiang Y, Zhan ZH, Tan KC, Kwong S, Zhang J. Knowledge structure preserving-based evolutionary many-task optimization. *IEEE Trans Evol Comput.* 2025;29(2):287–301. doi:10.1109/TEVC.2024.3355781.
28. Li YC, Gong WY, Gu Q. Transfer task-averaged natural gradient for efficient many-task optimization. *IEEE Trans Evol Comput.* 2025;29(5):1952–65. doi:10.1109/tevc.2024.3459862.
29. Wang XL, Kang Q, Zhou MC, Deng Q, Fan Z, Liu HY. Knowledge classification-assisted evolutionary multitasking for two-task multiobjective optimization problems. *IEEE/CAA J Autom Sin.* 2025;12(6):1176–93. doi:10.1109/JAS.2024.125070.
30. Zhang JN, Xu C, Li J, Chen WZ, Wang YB, Tai Y, et al. Analogous to evolutionary algorithm: designing a unified sequence model. *Adv Neural Inf Process Syst.* 2021;34:26674–88.
31. Zhang XR, Gong YJ, Zhong YT, Huang T, Zhang J. Large language model as meta-surrogate for offline data-driven many-task optimization: a proof-of-principle study. *Inf Sci.* 2025;726(1):122762. doi:10.1016/j.ins.2025.122762.
32. Wang XL, Jin YC. Distilling ensemble surrogates for federated data-driven many-task optimization. *IEEE Trans Evol Comput.* 2025;29(6):2401–15. doi:10.1109/tevc.2024.3428701.
33. Tan S, Wang Y, Sun G, Pang T, Tang K. A surrogate-assisted evolutionary framework for expensive multitask optimization problems. *IEEE Trans Evol Comput.* 2025;29(3):779–93. doi:10.1109/TEVC.2024.3370937.
34. He C, Huang S, Cheng R, Tan KC, Jin Y. Evolutionary multiobjective optimization driven by generative adversarial networks (GANs). *IEEE Trans Cybern.* 2021;51(6):3129–42. doi:10.1109/TCYB.2020.2985081.
35. Choi Y, Choi J, Kim M, Ha J, Kim S, Choo J. StarGAN: unified generative adversarial networks for multi-domain image-to-image translation. In: *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA.* p. 8789–97. doi:10.1109/CVPR.2018.00916.
36. Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A. A kernel two-sample test. *J Mach Learn Res.* 2012;13(4):723–73. doi:10.1007/s11222-026-10912-x.

37. Feng L, Qin K, Gupta A, Yuan Y, Ong YS, Chi X. CEC 2019 competition on evolutionary multi-task optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation 2019; 2019 Jun 10–13; Wellington, New Zealand.
38. Feng L, Qin K, Gupta A, Yuan Y, Ong YS, Chi X. WCCI 2020 competition on evolutionary multi-task optimization. In: Proceedings of the IEEE World Congress on Computational Intelligence 2020; 2020 Jul 19–24; Glasgow, UK.
39. Li YC, Gong WY, Zhang TY, Ming F, Li SJ, Gu Q, et al. MToP: a MATLAB benchmarking platform for evolutionary multitasking. *ACM Trans Evol Learn Optim.* 2026. doi:10.1145/3812535.
40. Cui Z, Zhao B, Zhao T, Cai X, Chen J. Adaptive multi-task evolutionary algorithm based on knowledge reuse. *Inf Sci.* 2023;648:119568. doi:10.1016/j.ins.2023.119568.
41. Liaw RT, Ting CK. Evolutionary manytasking optimization based on symbiosis in biocoenosis. *Proc AAAI Conf Artif Intell.* 2019;33(1):4295–303. doi:10.1609/aaai.v33i01.33014295.
42. Zhao J, Zhang Y, Zhong R, Chen H, Lian JJ, Gao ZM. A paradigm of evolutionary manytasking optimization for solving nonlinear equation systems: a two-stage framework with adaptive knowledge transfer, sharing and hybrid resource sampling. *Expert Syst Appl.* 2026;297:129459. doi:10.1016/j.eswa.2025.129459.
43. Liu Z, Huang R, Shao S. Data-driven two-stage fuzzy random mixed integer optimization model for facility location problems under uncertain environment. *AIMS Math.* 2022;7(7):13292–312. doi:10.3934/math.2022734.
44. Liu L, Hua T. A polynomial interior-point algorithm with improved iteration bounds for linear optimization. *Jpn J Ind Appl Math.* 2024;41(2):739–56. doi:10.1007/s13160-023-00630-6.