

Article

An Algorithm for Large-Scale Multi-Objective Optimization Problems Digital Twin-Multi-Layer Cooperative Evolutionary Algorithm

Delu Li ^{1,2,3}, Fengying Gou ^{1,2,3}, Can Cui ^{1,2,3}, Huiyi Wang ^{1,2,3}, Haiyang Lyu ^{2,4}, Yutao Song ^{2,4}, Xin Lu ^{3,5}, Min Deng ^{2,6} and Wu Bo ^{1,2,3,*}

¹ School of Engineering, Xizang University, Lhasa 850000, China; 15506403318@163.com (D.L.); 15328260517@163.com (F.G.); 19803659887@163.com (C.C.); 15990402232@163.com (H.W.)

² Plateau Major Infrastructure Smart Construction and Resilience Safety Technology Innovation Center, Lhasa 850000, China; lvhaiyang@stu.xjtu.edu.cn (H.L.); syutao@njust.edu.cn (Y.S.); minlinda@163.com (M.D.)

³ Lhasa Key Laboratory of Lightweight and High-Strength Metal Structures and Engineering Materials for Plateau Regions, Lhasa 850000, China; luxin@ustb.edu.cn

⁴ Nanjing Hanhai Fuxi Defense Technology Co., Ltd., Nanjing 211100, China

⁵ Institute of Engineering Technology, University of Science and Technology Beijing, Beijing 102206, China

⁶ Nanjing Modern Multimodal Transportation Laboratory, Nanjing 211100, China

* Correspondence: thinfog@xzu.edu.cn

Abstract

Large-scale multi-objective optimization problems involve high-dimensional decision variables, vast search spaces, and conflicting objectives, which often lead to slow convergence, premature convergence, and poorly distributed solutions when conventional evolutionary algorithms operate under limited evaluation budgets. To address these challenges, this study proposes DT-MLCEA (Digital Twin-Multi-Layer Cooperative Evolutionary Algorithm), a digital twin-driven, multi-layer cooperative evolutionary algorithm. The method uses evolutionary search as its core and introduces a search-state-oriented digital twin feedback mechanism that jointly models convergence trends, diversity, distribution sparsity, stagnation, and state confidence. A multi-source adaptive control vector (MACV) is then generated to adjust key search parameters dynamically. The digital twin layer further integrates Mini-RL, Mini-AE, and Mini-Meta to refine its control recommendations and balance global exploration with local exploitation. DT-MLCEA is evaluated against LMEA, QLMGO, AMSLMOE, and MOEA/D-RDG on LSMOP1-LSMOP9 with 100, 500, 1000, and 5000 decision variables. Across 36 problem-dimension combinations, DT-MLCEA significantly outperforms the competing algorithms in terms of HV and IGD in 28 and 24 combinations, respectively. Extended experiments on LSMOP1-LSMOP3 indicate stable performance with up to 50,000 variables. Ablation, noisy-evaluation, and parameter-sensitivity analyses further demonstrate the effectiveness of the collaborative modules.

Keywords: digital twin; large-scale multi-objective optimization; reinforcement learning; multi-layer cooperative evolutionary algorithm; adaptive control vector



Academic Editor: Frank Werner

Received: 6 August 2026

Revised: 18 September 2026

Accepted: 23 September 2026

Published: 1 October 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The digitalization of complex-system design and decision-making has generated numerous optimization problems involving thousands or even tens of thousands of decision variables. Examples range from the layout and routing of very-large-scale integrated circuits [1] and end-to-end learning based on deep neural networks [2] to real-time power-grid

dispatch [3]. Such problems typically involve multiple conflicting objectives. Optimization problems with more than 100 decision variables are commonly referred to as large-scale multi-objective optimization problems (LSMOPs) [4]. Over several decades, evolutionary multi-objective optimization (EMO) has demonstrated the effectiveness of evolutionary algorithms for solving multi-objective problems. Classical algorithms, including NSGA-II [5], SPEA2 [6], and MOEA/D [7], perform well on low-dimensional problems, typically involving fewer than 100 decision variables, and are widely used as benchmark algorithms. However, their performance can deteriorate markedly when they are applied directly to LSMOPs because of the curse of dimensionality [8]. This limitation has become a major obstacle to the application of EMO techniques to complex real-world problems [9,10]. In 2013, Antonio and Coello proposed CCGDE3, the first algorithm specifically designed for large-scale multi-objective optimization [11]. Since then, increasing attention has been devoted to this field. According to recent reviews [12,13], existing methods can be classified into three categories according to how they handle large numbers of decision variables: decision-variable grouping methods, decision-space reduction methods, and methods based on novel search strategies. These categories are reviewed below.

The first category directly partitions large numbers of decision variables using divide-and-conquer methods (DCMs). For example, CCGDE3 randomly divides the decision variables into equally sized groups and employs third-generation generalized differential evolution [14] as the optimizer, achieving strong performance on benchmark problems. MOEA/D-VA instead groups variables according to their correlation characteristics, thereby decomposing complex problems more effectively and avoiding the limitations of random grouping when dependencies exist among variables. Subsequent studies have used clustering to divide decision variables into convergence-related and diversity-related groups [15]. However, this approach requires numerous function evaluations during grouping. To reduce this cost, the dynamic random grouping method MOEA/D-RDG [16] partitions dimensions dynamically according to heuristic information about variable dependencies rather than using fixed-size groups, substantially reducing the number of function evaluations required during evolution. WOF-MMOPSO-RDG [17] provides a hybrid framework that combines WOF [18] with the RDG strategy in a particle swarm optimizer, improving both grouping and search efficiency. Other approaches include difference-based grouping MOEAs [19,20] and methods based on relationships among variables [21–23]. In essence, these methods simplify a complex problem by decomposing it into independent subproblems that can be solved in parallel using existing optimizers. Their effectiveness therefore depends strongly on the decomposition strategy. Important limitations remain, including the large number of function evaluations required for grouping and relatively poor performance on problems with complex variable dependencies.

The second category reduces the decision space and performs the search in a lower-dimensional representation. One approach reformulates the original problem in a lower-dimensional space, solves the transformed problem, and maps the resulting solutions back to the original space. The Weighted Optimization Framework (WOF), for example, reduces dimensionality through problem transformation and can be combined with different evolutionary algorithms. LSMOF [24] constructs low-dimensional single-objective subproblems by establishing bidirectional associations among weight variables and then solves them using single-objective optimizers. Similarly, iLSMOA [25] reformulates the original problem as a single-objective optimization problem and uses the resulting solutions to initialize MOEA/D-DE [26]. This process is repeated to obtain the final solution set. However, problem reformulation may overemphasize convergence. To improve both effectiveness and efficiency, LMOEA-DS [27] follows the LSMOF principle to construct reference directions and randomly samples solutions along them. Another approach reduces the

number of variables by exploiting relationships among them. ReMO [28], for example, uses random embedding to address LSMOPs with low intrinsic dimensionality, whereas principal component analysis has also been applied for this purpose [29]. However, both approaches rely primarily on linear relationships. Consequently, dimensionality-reduction techniques may lose important information, fail to preserve the global structure of the original search space, and become sensitive to the selected reduction model.

Both DCMs and dimensionality-reduction techniques improve efficiency by simplifying the original search space, potentially at the expense of global exploration. This trade-off has motivated a third research direction: the design of specialized search strategies that operate directly in high-dimensional spaces without grouping or dimensionality reduction. Representative approaches include competitive or direction-guided mechanisms and specialized evolutionary operators [30–33]. Although these methods are promising, they may converge slowly and require many function evaluations to obtain satisfactory solutions. More recently, online learning models [34,35] and surrogate models [36] have been introduced to guide offspring generation, accelerate convergence, and reduce the number of expensive evaluations. However, these models may incur additional computational overhead, creating a further trade-off between effectiveness and efficiency.

Within this third category, digital twin (DT) technology offers a relevant perspective because it combines virtual–physical mapping, state synchronization, online feedback, and dynamic optimization. DT technology is increasingly being applied beyond manufacturing-system modeling to complex optimization and intelligent scheduling. Tao et al. [37] reviewed the principal components, applications, and development trends of industrial digital twins and highlighted their potential to connect physical systems with virtual spaces while supporting state perception and process optimization. In learning-assisted evolutionary optimization, Song et al. [38] proposed DLES, which learns discrete search paths from evolutionary samples and reconstructs potential continuous paths using deep networks to improve subsequent candidate generation. Yu et al. [39] developed the serial multilevel-learned differential evolution algorithm (SMLDE), which combines a tightly coupled multilevel learning structure, an adaptive current-best layer, single-iteration adaptation, Cauchy perturbation, and iterative constraints to balance exploration and exploitation dynamically. Fang et al. [40] proposed an adaptive job-shop scheduling mechanism that runs reinforcement learning in a digital twin environment, improving responsiveness to dynamic disturbances through real-time state interaction and feedback. Ouahabi et al. [41] reviewed the integration of digital twins into dynamic production scheduling and noted that existing studies focus mainly on state synchronization, simulation validation, and scheduling feedback, whereas closed-loop mechanisms tightly coupled with optimization remain limited. Surrogate-assisted methods have also advanced expensive and high-dimensional multi-objective optimization. Pan et al. [42] proposed a surrogate-assisted reformulation and decomposition method to reduce actual evaluations and improve high-dimensional search efficiency. Gu et al. [43] used low-dimensional surrogates of scalarization functions to guide large-scale multi-objective evolutionary search, balancing convergence and diversity. Collectively, these studies improve complex optimization through search-path reconstruction, hierarchical individual learning, digital twin-based scheduling, and surrogate evaluation. Nevertheless, most focus on candidate generation, differential-search strategies, external modeling, or the replacement of real evaluations rather than on a lightweight algorithmic digital twin that integrates persistent state mirroring, multi-source control, online feedback, strategy memory, and tight coupling with the main evolutionary process.

In summary, a central challenge in LSMOP research is to achieve efficient global convergence, well-distributed solutions, and controllable computational cost simultaneously

within a vast and complex search space. To address this challenge, this study proposes DT-MLCEA. By integrating algorithmic digital twin-driven state modeling, reinforcement learning-based parameter adjustment, meta-policy-based strategy switching, and lightweight autoencoder-based state representation, DT-MLCEA establishes a cross-layer closed-loop framework for evolutionary search. This framework coordinates perception, learning, adaptation, and stabilization to improve robustness and resource utilization in large-scale multi-objective optimization. The main contributions are as follows:

1. A digital twin-driven, multi-layer cooperative evolutionary algorithm, termed DT-MLCEA, is proposed. The evolutionary algorithm acts as the sole optimization engine, while the digital twin layer monitors the population state and search trends online. This design establishes a closed loop comprising state perception, control decision-making, and evolutionary execution, thereby improving adaptability and robustness in complex high-dimensional search spaces.
2. A multi-source adaptive control vector (MACV) mechanism based on information fusion is developed. The digital twin layer jointly represents population diversity, convergence trends, distribution dispersion, stagnation, and state reconstruction error to generate multidimensional control recommendations. These recommendations are integrated with a rule-based controller, lightweight reinforcement learning, and meta-policy gating to adjust crossover probability, mutation probability, elite bias, innovation strength, and contraction intensity dynamically, thereby balancing global exploration and local exploitation.
3. A Lightweight Enhancement Bundle comprising Mini-RL, Mini-AE, and Mini-Meta is introduced. Mini-RL performs low-cost policy correction through a discrete state space and Q-learning; Mini-AE detects anomalies and distribution drift using state reconstruction error; and Mini-Meta switches dynamically among the Explore, Exploit, and Recover modes according to the search state. Together, these modules improve environmental adaptability and search stability without the training overhead associated with complex deep networks.

2. Related Work

Section “Large-Scale Multi-Objective Optimization Problem” introduces the fundamental concepts of LSMOPs, whereas Meanwhile, this section reviews the components related to the proposed algorithm and explains the motivation for this study.

Large-Scale Multi-Objective Optimization Problem

The mathematical formulation of an LSMOP follows that of a standard multi-objective optimization problem (MOP). An unconstrained MOP can be formulated as follows.

$$\min_x F(x) = (f_1(x), f_2(x), \dots, f_m(x))^T \quad (1)$$

$x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$: The decision vector comprises n decision variables representing the parameters to be optimized. x : The corresponding space is the decision space, and the associated mapping defines the objective-function vector. $n \Omega F(x): \mathbb{R}^n \rightarrow \mathbb{R}^m$: This vector comprises several potentially conflicting objective functions. $m f_i(x)$ m : The corresponding space is called the objective space $\mathbb{R}^m$. $100 \leq n \leq 2 \leq m$: Problems satisfying the dimensionality condition above are referred to as LSMOPs. Because the objectives generally conflict, an MOP has no single optimal solution; instead, it has a set of trade-off solutions called Pareto-optimal solutions. The relevant Pareto-based definitions are given below:

1. Pareto dominance: Given two feasible solutions, one solution dominates the other if the following conditions hold: $x_1 \preceq x_2$ if $f_i(x_1) \leq f_i(x_2) \forall i$ and $f_j(x_1) < f_j(x_2)$ for at least one j . These conditions define Pareto dominance.
2. Non-dominated solution: A feasible solution is non-dominated if no other feasible solution in the entire feasible set dominates it: $x^* \in X^*$.
3. Pareto optimality: A solution is Pareto optimal if it is non-dominated within the feasible set: x^* .
4. Pareto optimal set: The set of all Pareto-optimal solutions to a given LSMOP is called the Pareto-optimal set.
5. Pareto front: The curve or surface formed by mapping the solutions in the Pareto-optimal set to the objective space is called the Pareto front.

These trade-off solutions constitute the Pareto-optimal set, whose image in the objective space is the Pareto front (PF). The principal objective is to identify a diverse and uniformly distributed solution set that closely approximates the true PF.

3. Algorithm

To address the high-dimensional search spaces of LSMOPs, the difficulty of balancing convergence and distribution, and the inability of fixed parameters to adapt to different search stages, this study proposes DT-MLCEA, a digital twin-driven, multi-layer cooperative evolutionary algorithm. In the current implementation, an evolutionary algorithm (EA) serves as the sole optimization engine, while an internal lightweight digital twin control layer performs online search-state perception, trend analysis, and parameter adjustment. Unlike earlier designs based on computationally intensive surrogate models or deep-network training, this implementation does not use deep surrogate prediction or generative candidate-solution mechanisms. Instead, a Lightweight Enhancement Bundle comprising Mini-RL, Mini-AE, and Mini-Meta applies low-frequency, small-magnitude, and interpretable corrections to the control recommendations generated by the digital twin layer. The resulting closed loop consists of search-state perception, parameter adjustment, and evolutionary updating.

3.1. DT-MLCEA Algorithm Design

Algorithm 1 outlines the workflow of DT-MLCEA. The procedure begins by initializing the EA core together with the lightweight digital twin module, generating an initial population P of size N , evaluating the true objective values, and setting the function-evaluation counter FE . During each generation, the DT layer is invoked at a low update frequency based on the current population state and recent performance, while an additional update can be triggered under stagnation; it then outputs parameter recommendations and mode signals for the EA search. Guided by the current MACV, the EA performs mating selection, crossover, and polynomial mutation to produce offspring Q . These offspring are evaluated using the true objective functions, optionally with batch or parallel execution, and FE is updated accordingly. Environmental selection is subsequently applied to $P \cup Q$ via fast non-dominated sorting and crowding-distance estimation to form the next population. Meanwhile, HV, the stagnation indicator, and the historical state are refreshed at predefined intervals to support the DT state estimation and the next control decision. The algorithm terminates when the evaluation budget or another stopping criterion is reached, returning the final population and its non-dominated solutions.

Algorithm 1: Main Procedure of DT-MLCEA

Input: N (population size), FEmax (maximum number of evaluations), Bounds Ω
Output: P (final population), F(P) (objective values)

```

1  DT ← DigitalTwinController(update_interval, stagnation_threshold, control_strength)
2  P ← SampleInitialPopulation(N,  $\Omega$ )
3  P ← InjectAnchors(P) //overwrite first  $\leq 8$  individuals
4  F ← TrueEvaluate(P); FE ← |P|; t ← 0
5  A_dec ← empty; A_fit ← empty //external archive
6  best_front ← None; best_hv ← -inf
7  last_improve_gen ← 0
8  while true do
9      [P, F] ← UniqueByDecision(P, F)
10     [idx, rank, crowd] ← EnvironmentalSelection(F, min(N, |P|))
11     P ← P[idx]; F ← F[idx]; rank ← rank[idx]; crowd ← crowd[idx]
12     F1 ← {F_i | rank_i = 0}; P1 ← {P_i | rank_i = 0}
13     [A_dec, A_fit] ← UpdateArchive(A_dec, A_fit, P1, F1, N)
14     HV ← NormalizedHV(F1)
15     Append (t, FE, HV, |F1|, |A_fit|) to history
16     if HV improves then
17         best_front ← copy(F1); best_hv ← HV
18         last_improve_gen ← t
19     if FE ≥ FEmax then break end if
20     force_refresh ← (t − last_improve_gen ≥ DT.stagnation_threshold)
21     MACV ← DT.Update(t, P, F1, HV, force_refresh)
22     n_off ← min(N, FEmax − FE)
23     Q ← GenerateOffspring(P, F, rank, crowd, A_dec, A_fit, MACV, n_off)
24     FQ ← TrueEvaluate(Q)
25     FE ← FE + |Q|
26     P ← P union Q
27     F ← F union FQ
28     t ← t + 1
29  if best_front is None then best_front ← A_fit end if
30  return best_front, corresponding objective values

```

3.2. EA Core Evolution Mechanism

The EA core performs the evolutionary search. In the first generation, it receives the control vector generated by the DT layer. At generation t , the EA core receives the following five-dimensional multi-source adaptive control vector from the digital twin layer:

$$MACV_t = [p_m^t, p_c^t, b_e^t, r_i^t, r_c^t]^T \quad (2)$$

Here, p_m^t and p_c^t denote the mutation and crossover probabilities, respectively; b_e^t denotes the elite/archive bias; r_i^t denotes the innovation rate controlling exploratory offspring generation; and r_c^t denotes the contraction rate controlling the reduction in the perturbation range.

Simulated binary crossover (SBX) and polynomial mutation were combined in the mutation generation stage. For the j -th decision variable of the parent individuals, random

numbers are extracted, and the SBX crossover coefficient is defined as follows: x^a, x^b j $u \sim U(0,1)$ $\beta_{q,j}$.

$$\beta_{q,j} = \begin{cases} (2u)^{\frac{1}{\eta_c+1}}, & \text{if } u \leq 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{\eta_c+1}}, & \text{if } u > 0.5 \end{cases} \quad (3)$$

The random variable follows a uniform distribution over the specified interval; $u \sim U(0,1)$ $[0,1]$ η_c : the simulated binary crossover (SBX) distribution index controls the distribution of offspring around their parents. A larger index generates offspring closer to their parents and therefore favors local exploitation: η_c . A smaller index broadens the offspring distribution and enhances global exploration: η_c . The coefficient adjusts the search span of the decision variable during crossover and thus characterizes the strength of parental information mixing in that dimension: $\beta_{q,j}$.

The formula for generation of offspring variables is as follows.

$$y_{1,j} = 0.5 \times ((1 + \beta_{q,j}) \times x_j^a + (1 - \beta_{q,j}) \times x_j^b) \quad (4)$$

$$y_{2,j} = 0.5 \times ((1 - \beta_{q,j}) \times x_j^a + (1 + \beta_{q,j}) \times x_j^b) \quad (5)$$

where the variables represent the values of the two parents for the j -th decision variable, whereas x_j^a, x_j^b j $y_{1,j}$ the remaining variables denote the corresponding offspring values: $y_{2,j}$.

Subsequently, the polynomial mutation operation is performed. The random number is extracted, and the mutation perturbation term is: $r \sim U(0,1)$ δ_j .

$$\delta_j = \begin{cases} (2r)^{\frac{1}{\eta_m+1}} - 1, & \text{if } r < 0.5 \\ 1 - (2(1-r))^{\frac{1}{\eta_m+1}}, & \text{if } r \geq 0.5 \end{cases} \quad (6)$$

where denotes a uniformly distributed random variable, and $r \sim U(0,1)$ η_m the polynomial-mutation distribution index controls the magnitude of the mutation perturbation.

Apply the modulation factor and enforce the boundary restriction (for the decision variable boundary): $\rho_t l_j \leq u_j$.

$$y'_j = y_j + \rho_t \times \delta_j \times (u_j - l_j) \quad (7)$$

$$y' = \min(\max(y', l_j), u_j) \quad (8)$$

where the current exploration-intensity modulation factor dynamically controls the mutation step size: $\rho_t u_j$. The remaining variables denote the upper and lower bounds of the j -th decision variable, respectively: l_j .

The environment selection stage performs a fast non-dominated sort on the merged population: U_t . If the size of the next generation is exceeded due to the addition of an individual in the frontier, the current frontier is truncated in descending order of crowding distance: N . For the first objective, the crowding distance update rule for intermediate individuals is as follows: m i .

$$\text{crowding}_i = \frac{f_{i+1,m} - f_{i-1,m}}{f_m^{\max} - f_m^{\min} + \varepsilon} \quad (9)$$

where the variables denote the objective values of the neighboring individuals after sorting. $f_{i+1,m}, f_{i-1,m}$ m $f_m^{\max}$: The remaining terms denote the maximum and minimum values of the first objective in the current front. $f_m^{\min}$ m ε and the final term is a small positive constant to prevent the denominator from being zero.

Algorithm 2 presents the process of generating offspring under the guidance of MACV. Firstly, parameters such as crossover, mutation, innovation and archive bias are determined based on the control vector, and the number of conventional mating and immigrant individuals is also divided; subsequently, the parent generation is constructed through tournament selection and archive guidance, and differential generation or SBX crossover is adopted according to the current settings, combined with polynomial mutation to generate offspring. Finally, the immigrant individuals are added to the population to obtain the final offspring set Q.

Algorithm 2: Offspring Generation Under MACV

```

Input:      P, F, rank, crowd, A_dec, A_fit, MACV, n_off
Output:     Q
1  p_m ← MACV.mutation_rate
2  p_c ← MACV.crossover_rate
3  b_e ← MACV.elite_bias
4  r_i ← MACV.innovation_rate
5  r_c ← MACV.contraction_rate
6  r_de ← r_i
7  r_im ← clip(r_i × (1 − r_c))
8  r_arc ← b_e
9  n_im ← round(r_im × n_off), clipped to [0, n_off]
10 n_mate ← n_off − n_im
11 parent_idx ← BinaryTournament(rank, crowd, max(n_mate, 1))
12 Parents ← P[parent_idx]
13 if A_dec not empty and r_arc > 0 then
14     replace a subset of Parents by archive solutions
15 pool ← P if archive empty else P union A_dec
16 Q ← empty array of size n_off
17 for each pair (x_a, x_b) in Parents over first n_mate positions do
18     rand() < r_de and |pool| >= 3 then
19         sample 3 solutions from pool
20         y1 ← DifferentialTrial(x_a, a, b, c, scale = clip(0.35 + 0.5 × pm), cr = clip(pc))
21         y1 ← PolynomialMutation(y1, max(0.75 × pm, 0.02))
22         similarly generate y2 if needed
23     else
24         if rand() < pc then
25             (y1, y2) ← SBX(x_a, x_b)
26         else
27             (y1, y2) ← (x_a, x_b)
28         y1 ← PolynomialMutation(y1, pm)
29         y2 ← PolynomialMutation(y2, pm)
30     write offspring into Q
31 if n_im > 0 then
32     Q[last n_im positions] ← MakeImmigrants(n_im, A_dec, P)
33 return Q
  
```

3.3. Digital Twin State Modeling

In this study, the digital twin layer is defined as an algorithmic digital twin of the evolutionary search process. It continuously synchronizes with the evolving population

by monitoring convergence, diversity, sparsity, stagnation, and state confidence, and then updates the five-dimensional MACV defined in Section 3.2. Unlike conventional adaptive control based on a direct state-to-parameter mapping, the proposed mechanism maintains a persistent virtual state comprising historical windows, latent representations, action values, policy modes, and previous control information. Mini-RL estimates the expected long-term rewards of bounded control actions rather than predicting objective values. Through periodic synchronization and feedback of MACV to the EA core, the framework forms a closed loop of state perception, control decision-making, evolutionary execution, and search feedback. It is therefore characterized as a lightweight algorithm-level digital twin rather than a high-fidelity replica of a physical entity. Table 1 illustrates the differences between the Proposed Algorithm-Level Digital Twin and other methods.

Table 1. Comparison of conventional adaptive parameter control, surrogate-assisted optimization, typical digital twins, and the algorithm-level digital twin proposed in this study.

Dimension	Conventional Self-Adaptive Parameter Control	Surrogate-Assisted Optimization	Typical Digital Twin	Proposed Algorithm-Level Digital Twin
Modeling object	Current search statistics state	Objective function or fitness response surface	Physical entities and their operational processes Entity-Model	Persistent state of the evolutionary search process
Feedback form	Direct mapping from state to parameters	Predictive assistance for screening or evaluation	Bidirectional Mapping and Closed-Loop Optimization	MACV-driven closed-loop regulation
Mirror form	Typically no independent persistent mirror	Typically, full process images are not built.	High-fidelity mirror of physical objects	Lightweight state mirroring of the search process
Synchronization mechanism	Triggered by rules	Updated as new samples become available	Continuous data synchronization	Low-frequency refresh with stagnation-triggered synchronization
Main function	Parameter self-adaptation adjustment	Reduce evaluation costs	Monitoring, predicting, and controlling physical systems	Characterizing the search phase and closed loop control
Prediction target	Usually no explicit prediction	Unassessed solution's objective value	Sensor and model errors	State confidence and reconstruction error

The proposed digital twin is not a high-fidelity simulation of physical equipment but a lightweight algorithmic twin of the evolutionary search process. Its virtual entity stores not only current population metrics but also the historical window, latent states, reinforcement-learning value table, meta-policy modes, and control input from the previous time step. The twin state at generation t is defined as follows:

$$T_t = \{s_t, H_t, z_t, Q_t, m_t, a_{t-1}\} \quad (10)$$

Here, s_t denotes the current search state; H_t is the historical state window; z_t is the low-dimensional state representation; Q_t is the action-value table; m_t is the meta-policy mode; and a_{t-1} is the control vector from the preceding refresh. This persistent virtual state is synchronized online with the actual population, and its control output is fed back to the next-generation search.

The search state vector is represented as:

$$s_t = [div_t, \Delta HV_t, spread_t, sparse_t, stag_t, conf_t]^T \quad (11)$$

where div_t denotes diversity in the decision space, ΔHV_t denotes the change in HV , $spread_t$ denotes dispersion in the objective space, $sparse_t$ denotes the proportion of sparse regions, $stag_t$ denotes the stagnation state, and $conf_t$ denotes confidence in the mirrored state.

Let the population decision matrix be $X_t \in R^{N \times D}$. Diversity is then defined as follows:

$$div_t = \frac{1}{D} \sum_{j=1}^D \frac{std(X_t(:, j))}{\max(X_t(:, j)) - \min(X_t(:, j)) + \varepsilon} \quad (12)$$

The convergence trend is quantified by the difference in HV between two consecutive refresh points.

$$\Delta HV_t = HV_t - HV_{t^-} \quad (13)$$

where t^- denotes the most recent twin-state refresh rather than necessarily the preceding generation.

Up to 64 objective vectors are selected from the current population to form F_t . For each vector, the nearest-neighbor distance is computed as follows:

$$d_i = \min_{j \neq i} \|f_i - f_j\|_2 \quad (14)$$

Dispersion and the sparsity ratio in the objective space are defined as follows:

$$spread_t = \frac{std(\{d_i\})}{mean(\{d_i\}) + \varepsilon} \quad (15)$$

$$sparse_t = \frac{1}{|F_t|} \sum_{i=1}^{|F_t|} \prod (d_i > \bar{d} + \sigma_d) \quad (16)$$

where d_t and σ_d are the mean and standard deviation of the nearest neighbor distances, respectively.

Stagnation is detected using an HV sliding window of length $W = 8$:

$$stag_t = \prod \left(\max_{k \in W_t} HV_k - \min_{k \in W_t} HV_k < 10^{-6} \right) \quad (17)$$

State confidence is determined jointly from fluctuations in HV and diversity and from the current distribution dispersion:

$$conf_t = clip \left(\frac{1}{1 + 5\sigma_{HV,t} + spread_t}, 0.2, 0.95 \right) \quad (18)$$

Smaller fluctuations in HV and diversity indicate a more stable search state, whereas a larger $spread_t$ reduces confidence in the twin state.

$$a^0 = [0.1, 0.9, 0.55, 0.08, 0.08]^T \quad (19)$$

When $stag_t = 1$, the mutation, crossover, and innovation rates are increased, whereas elite bias and contraction are reduced. When $div_t < 0.05$ or $sparse_t > 0.35$, the mutation and innovation rates are increased. When $conf_t > 0.75$ and $\Delta HV_t > 0$, the mutation and innovation rates are reduced, while the crossover rate, elite bias, and contraction are increased to reinforce the current effective search direction.

The rule-based control vector is combined with the Mini-RL action offset Δa_t^{RL} as follows:

$$\bar{a}_t = \text{clip} \left[(I - \Lambda) a_t^{rule} + \Lambda (a_0 + \Delta a_t^{RL}) \right] \quad (20)$$

Among them:

$$\Lambda = \text{diag}(0.18, 0.18, 0.1, 0.1, 0.1) \quad (21)$$

Mini-RL estimates the long-term rewards associated with different control actions through Q-value estimation:

$$Q(s_{t-1}, a_{t-1}) \leftarrow Q(s_{t-1}, a_{t-1}) + \eta \left[r_t + \gamma \max_a Q(s_t, a) - Q(s_{t-1}, a_{t-1}) \right] \quad (22)$$

Thus, the twin layer responds to current indicators while using historical interactions to estimate the expected rewards of alternative control actions. The meta-policy generates an additional mode-correction term, Δa_t^{meta} , yielding:

$$a_t^* = \text{clip}(\bar{a}_t + \Delta a_t^{meta}) \quad (23)$$

To prevent abrupt changes in the control parameters, temporal smoothing is applied to the final output:

$$a_t = \text{clip}[\alpha a_{t-1} + (1 - \alpha) a_t^*] \quad (24)$$

By default, the digital twin is refreshed every $K = 12$ generations. Synchronization occurs earlier if the stagnation counter reaches the forced-refresh threshold; otherwise, the latest control output is reused. The complete closed loop is expressed as follows:

$$P_t \rightarrow s_t \rightarrow T_t \rightarrow a_t \rightarrow P_{t+1} \rightarrow T_{t+1} \quad (25)$$

The digital twin layer therefore integrates multiple search-state signals into $MACV_t$, preserves temporal continuity through boundary constraints and smoothing, and feeds the resulting vector back to the EA layer at a low frequency to coordinate exploration and exploitation.

Algorithm 3 describes the process of state modeling and control generation in the digital twin layer. Firstly, based on the current population and the first frontier, various state information such as diversity, convergence change, distribution sparsity, stagnation state and confidence level is calculated; if there is no need to refresh at the moment, the cached control vector is directly returned. Otherwise, the algorithm combines the outputs of the rule controller, Mini-AE, Mini-RL and Mini-Meta to generate and correct the current MACV, then performs constraint and smoothing processing according to different modes, and finally obtains the control parameters for the next round of evolution.

3.4. Lightweight Enhancement Bundle (Mini-RL, Mini-AE, Mini-Meta)

In DT-MLCEA, the Lightweight Enhancement Bundle does not participate directly in population evolution or candidate-solution generation. Instead, it applies small, low-frequency, and interpretable corrections to the output of the digital twin's base controller. Mini-RL learns the long-term rewards of alternative control actions from the search state; Mini-AE extracts a low-dimensional state representation and provides an auxiliary uncertainty signal through reconstruction error; and Mini-Meta switches among exploration, exploitation, and recovery modes by combining recent search history with reconstruction error. Together, these modules improve the balance between exploration and exploitation at different search stages without incurring the training cost of complex deep networks.

Algorithm 3: DT State Modeling and Control

Input: generation t , population P_t , first-front F_t^1 , current HV $_t$, cached MACV $_{t-1}$

Output: MACV $_t$

```

1  div $_t \leftarrow \text{Diversity}(P_t)$ 
2  spread $_t \leftarrow \text{Spread}(F_t^1)$ 
3  sparse $_t \leftarrow \text{SparseRatio}(F_t^1)$ 
4  hv_delta $_t \leftarrow \text{HV}_t - \text{HV}_{\{t-1\}}$  //if no history, use 0
5  stag $_t \leftarrow \text{DetectStagnation}(\text{HV window})$ 
6  conf $_t \leftarrow \text{Confidence}(\text{HV window, diversity window, spread}_t)$ 
7  s $_t \leftarrow [\text{div}_t, \text{hv\_delta}_t, \text{spread}_t, \text{sparse}_t, \text{stag}_t, \text{conf}_t]$ 
8  if not NeedRefresh( $t$ , stag $_t$ , force_refresh) then
9      return cached MACV
10 [pm_rule, pc_rule, be_rule, ri_rule, rc_rule]  $\leftarrow \text{RuleController}(s_t)$ 
11 x $_t \leftarrow [\text{div}_t, \text{hv\_delta}_t, \text{spread}_t, \text{sparse}_t, \text{conf}_t, \text{I}(\text{stag}_t > 0)]$ 
12 [z $_t$ , recon $_t$ ]  $\leftarrow \text{MiniAE}(x_t)$ 
13 a_rl  $\leftarrow \text{MiniRLStep}(s_t)$ 
14 meta  $\leftarrow \text{MetaStep}(s_t, \text{recon}_t)$  //explore/exploit/recover
15 p_m  $\leftarrow \text{clip}(0.7 \times \text{pm\_rule} + 0.3 \times (0.1 + \text{a\_rl.delta\_pm}) + \text{meta.delta\_pm})$ 
16 p_c  $\leftarrow \text{clip}(0.7 \times \text{pc\_rule} + 0.3 \times (0.9 + \text{a\_rl.delta\_pc}) + \text{meta.delta\_pc})$ 
17 b_e  $\leftarrow \text{clip}(\text{be\_rule} + \text{a\_rl.delta\_be} + \text{meta.delta\_be})$ 
18 r_i  $\leftarrow \text{clip}(\text{ri\_rule} + \text{a\_rl.delta\_ri} + \text{meta.delta\_ri})$ 
19 r_c  $\leftarrow \text{clip}(\text{rc\_rule} + \text{a\_rl.delta\_rc} + \text{meta.delta\_rc})$ 
20 apply mode-dependent clamps:
21     if mode = recover then enforce a high r $_i$  and a low r $_c$ 
22     if mode = explore then enforce a lower bound on r $_i$  and an upper bound on b $_e$ 
23     if mode = exploit then enforce lower bounds on b $_e$  and r $_c$  when confidence is high
24 smooth pm and pc with previous values
25 cache MACV $_t = \{p_m, p_c, b_e, r_i, r_c\}$ 
26 cache mode $_t$  separately
27 return MACV $_t$ 

```

Let s_t denote the continuous search state extracted by the digital twin at generation t , a_t^{rule} the base control vector generated by the rule-based controller, Δa_t^{RL} the action offset produced by Mini-RL, and Δa_t^{meta} the mode correction produced by Mini-Meta. The final control vector a_t is obtained through multi-source fusion, boundary constraints, and temporal smoothing:

$$a_t = [p_{m,t}, p_{c,t}, b_{e,t}, r_{i,t}, r_{c,t}]^T \quad (26)$$

In this expression, $p_{m,t}$ and $p_{c,t}$ denote the mutation and crossover rates, respectively; $b_{e,t}$ denotes elite bias; $r_{i,t}$ denotes the innovation rate; and $r_{c,t}$ denotes the contraction rate. Mini-RL and Mini-Meta make small online adjustments to these five control components, whereas Mini-AE does not output control parameters directly but supplies latent states and reconstruction errors to Mini-Meta.

To preserve the stability of the main EA search, the rule-based controller remains the primary source of control vectors. Mini-RL supplies only scaled discrete action offsets, and Mini-Meta applies corrections of limited magnitude according to the current mode. The final output is clipped to the parameter bounds and smoothed across refresh cycles. Thus, the Lightweight Enhancement Bundle refines the decisions of the digital twin controller rather than replacing either the controller or the evolutionary algorithm.

3.4.1. Mini-RL: Lightweight Policy Modification for Low-Dimensional Discrete States

Mini-RL uses discrete states and a compact Q-table to learn small action corrections to the digital twin's base control vector without generating candidate solutions directly. Unlike deep reinforcement learning, it uses neither neural networks nor experience replay. The Q-table is updated only after an effective refresh of the digital twin layer, thereby limiting online learning overhead. Let t^* denote the preceding digital twin refresh and n the number of decisions made by Mini-RL. The continuous state is discretized as follows.

$trend_t \in \{\text{degrade, stable, improve}\}$, is determined by $hv_{\Delta t, t}$.

$stagnation_flag_t \in \{0,1\}$ By decision $stag_t > 0$.

$low_div_flag_t \in \{0,1\}$ By decision $div_t < \tau_{div}$.

Discrete state index:

$$state_id_t = trend_t \times 4 + stagnation_flag_t \times 2 + low_div_flag_t \quad (27)$$

An epsilon-greedy policy is used to select actions, and the Q-values are updated at a low frequency. The reward is defined as follows.

$$r_t = \Delta HV_t + w_c \times conf_t - w_s \times I(stag_t > 0) \quad (28)$$

where denotes the reinforcement-learning reward; r_t ΔHV_t the next term denotes the improvement in convergence; $conf_t$ the following term denotes current-state confidence; w_c the associated coefficient is the confidence-reward weight; w_s the next coefficient is the stagnation-penalty weight; $I(stag_t > 0)$ the final term is the stagnation-indicator function.

Q Update:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \times (r_t + \gamma \times \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)) \quad (29)$$

where the first term denotes $Q(s_t, a_t)$ the action-value function for the current state-action pair; s_t a_t α the next parameter is the learning γ rate; the following parameter is the discount factor; and $\max_{a'} Q(s_{t+1}, a')$ the final term denotes the maximum action value in the subsequent state.

3.4.2. Mini-AE: State Compression and Uncertainty Auxiliary Signals

Mini-AE is a lightweight approximation module for search-state compression and uncertainty characterization. It does not generate offspring or otherwise dominate the evolutionary process. Instead, it encodes the current search state into a compact latent representation and provides an auxiliary uncertainty signal based on reconstruction error, thereby supporting adaptive decision-making with limited computational overhead.

$$z_t = \tanh(x_t \cdot W_{enc}) \quad (30)$$

$$\hat{x}_t = \tanh(z_t \cdot W_{dec}) \quad (31)$$

where x_t denotes the current search-state feature vector; W_{enc} the next term denotes the encoder weight matrix; z_t the following term denotes the encoded low-dimensional latent variable; $\tanh(\cdot)$ the activation function is the hyperbolic tangent; W_{dec} the next term denotes the decoder weight matrix; $\hat{x}_t$ the following term denotes the reconstructed state; z_t and the final term is the latent variable.

The reconstruction error is defined as:

$$err_{vae, t} = \text{mean}((\hat{x}_t - x_t)^2) \quad (32)$$

$err_{vae,t}$: The reconstruction error serves as auxiliary uncertainty information and is supplied to Mini-Meta to determine whether the current state exhibits distributional drift or a structural anomaly. A larger error favors exploration: $err_{vae,t}$.

The Mini-AE used in this paper is a lightweight state encoding and reconstruction module, whose purpose is not to build a complete probabilistic generative model, but rather to represent the current search state compactly under low computational cost, and to reflect the degree of state anomaly through reconstruction error. Its input vector is defined as:

$$x = [div, \Delta HV, spread, spread_ratio, conf, I_{stage}] \in \mathbb{R}^6 \quad (33)$$

where I_{stage} is the stall indicator variable.

Unlike a standard VAE, this lightweight implementation does not use a probabilistic latent distribution, a reparameterization procedure, or a Kullback–Leibler divergence term. It uses only reconstruction error as an auxiliary signal of structural deviation.

$$err_{vae} = \frac{1}{6} ||\hat{x} - x||_2^2 \quad (34)$$

The reconstruction error is not treated as an independent optimization objective. Instead, it is provided to Mini-Meta as an auxiliary uncertainty signal indicating distributional drift or structural anomalies in the current search state. A larger reconstruction error indicates reduced reconstructability relative to the recent reconstruction-error distribution and therefore favors exploratory control. Conversely, a persistently small error indicates a relatively stable state and favors exploitation-oriented parameter adjustments.

3.4.3. Mini-Meta: Merging Mode Gating with Strategy

Rather than using gradient-based training, Mini-Meta switches modes according to statistics computed over a sliding window of length k :

$$\overline{\Delta HV}_t = \text{mean}(\Delta HV_{t-k+1:t}) \quad (35)$$

where $\overline{\Delta HV}_t$ denotes the mean recent change in hypervolume; $\Delta HV_{t-k+1:t}$ the following terms denote the HV-change sequence within the window; $t - k + 1$ to t and the final term denotes the sliding-window length.

$$\overline{div}_t = \text{mean}(div_{t-k+1:t}) \quad (36)$$

where the first term is the mean population diversity over recent generations; $\overline{div}_t$ and the remaining terms denote the diversity sequence in the corresponding window.

$$stag_ratio_t = \text{mean}(I(stag > 0)) \quad (37)$$

where $stag_ratio_t$ denotes the proportion of stagnant generations within the sliding window; $I(stag > 0)$ the next term is the stagnation-indicator function; $\text{mean}(\cdot)$ and the final operator denotes the mean.

$$\overline{err}_t = \text{mean}(err_{vae,t-k+1:t}) \quad (38)$$

The first term denotes the mean reconstruction error over the most recent k generations of Mini-AE; $\overline{err}_t$ and the remaining terms denote the reconstruction-error sequence within the window.

Pattern determination rules:

$$\begin{aligned} &\text{if } stag_ratio_t > \tau_{stag_ratio} \text{ and } \overline{h}v_t \leq 0 : mode_t = \text{recover} \\ &\quad \text{elseif } \overline{div}_t < \tau_{div} \text{ or } \overline{err}_t > \tau_{err} : mode_t = \text{explore} \\ &\quad \text{else} : mode_t = \text{exploit} \end{aligned} \quad (39)$$

3.4.4. Unified Fusion of Lightweight Enhancements

During fusion, the base control vector is first combined with the RL correction, after which the meta-gating correction is applied:

$$p_m^t = \text{clip}(w_1 \cdot p_m(\text{rule}) + w_2 \cdot (p_m(\text{base}) + \text{delta}_{rl_m}) + \delta_m^{\text{meta}}, p_m^{\min}, p_m^{\max}) \quad (40)$$

$$p_c^t = \text{clip}(w_1 \cdot p_c(\text{rule}) + w_2 \cdot (p_c(\text{base}) + \text{delta}_{rl_c}) + \delta_c^{\text{meta}}, p_c^{\min}, p_c^{\max}) \quad (41)$$

Clip is used for boundary stabilization. $w_1 + w_2 = 1$: This structure ensures that the enhancement module only makes “small step corrections” and does not destroy the stability of the EA main process.

Algorithm 4 describes the workflow of the lightweight enhancement module. Firstly, the algorithm discretizes the current search state and updates the RL actions through Q-learning in a low-frequency manner. At the same time, Mini-AE encodes and reconstructs the state features, and uses the reconstruction error to provide an auxiliary uncertainty signal. Subsequently, the relevant state information is written into the meta-history window, which is used by MetaGate to periodically determine whether the current state should enter the recover, explore or exploit mode, and finally outputs the RL correction amount, Meta correction amount and the corresponding mode.

Algorithm 4: Lightweight Enhancement Bundle

Input: state s_t, feature x_t, previous RL state/action, meta history window
Output: enhancement terms and mode

```

1  sid_t ← Discretize(s_t)
2  every rl_update_interval generations:
3      choose RL action by epsilon-greedy on Q[sid_t]
4      reward_t ← Reward(s_t, u_t, u_{t-})
5      update Q by TD rule
6  z_t ← tanh(x_t W_enc)
7  xhat_t ← tanh(z_t W_dec)
8  recon_t ← mean((xhat_t - x_t)2)
9  push {hv_delta_t, diversity_t, stagnation_t, recon_t} into meta history
10 every meta_update_interval generations:
11     mode_t ← MetaGate(history) // recover/explore/exploit
12 output RL deltas + meta deltas + mode_t
```

3.5. Information Flow and Online Update Procedure

This section describes inter-module communication and the online update schedule used to maintain an effective closed loop with minimal communication overhead. The implementation uses a hierarchical clock in which the EA drives the main loop, the DT is refreshed at a low frequency, and the enhancement modules are updated at their respective sub-frequencies. Let:

1. T_{bu} : bottom-up update interval (EA → DT).
2. T_{dt} : DT internal refresh interval.

3. T_{rl} : Mini-RL update interval.
4. T_{meta} : Mini-Meta update interval.
5. $stag_{trigger}$: Stagnation force threshold.

3.5.1. Information-Flow Topology and Minimum Message Set

Only the following two core communication links are retained:

- (1) $EA \rightarrow DT$ (State uplink)

$$msg_{up,t} = \{t, P_t, F_t, metrics_t, stag_t\} \quad (42)$$

- (2) $DT \rightarrow EA$ (Control delivery)

$$msg_{down,t} = \{p_c^t, p_m^t, mode_t, confidence_t\} \quad (43)$$

3.5.2. Event Triggering and Low-Frequency Updates

DT refresh condition:

$$need_refresh_t = (t - t_{last_refresh} \geq T_{dt}) \vee (stag_t \geq stag_{trigger}) \quad (44)$$

where indicates whether the digital twin state must be refreshed at the current generation; $need_refresh_t$: the next term denotes the current generation; $t_{last_refresh}$: the following term denotes the most recent DT refresh; T_{dt} denotes the DT refresh interval; $stag_t$: the next term denotes the current stagnation level; $stag_{trigger}$ and the final term denotes the stagnation-trigger threshold.

3.5.3. Offline Initialization and Online Learning

To mitigate cold-start instability, each submodule uses a combination of offline initialization and lightweight online updating:

Offline phase (one-time).

The Q-table, projection matrix, and gating thresholds are initialized using default values or a small warm-start sample:

$$\theta_0 = \text{Init}(\theta; config, optional_offline_data) \quad (45)$$

Online phase (per generation/divider)

Only the required statistics and lightweight parameters are updated; no computationally intensive backpropagation is performed:

$$\theta_{t+1} = \text{OnlineUpdate}(\theta_t, minibatch_t, low_cost_rule) \quad (46)$$

where $minibatch_t$ learning uses the sliding-window history rather than large-scale experience replay.

3.5.4. Function-Evaluation Counting and Consistency

During online optimization, the true function-evaluation count is incremented only when the true objective function is called:

$$FE_{t+1} = FE_t + |Q_t| \quad (47)$$

Any predicted value is used only as an auxiliary signal and is not counted as a true function evaluation. This convention ensures a consistent evaluation budget across algorithms.

4. Experimental Design

The performance of DT-MLCEA is evaluated on the LSMOP benchmark suite using LMEA, QLMGO [44], AMSLMOEa [45], and MOEA/D-RDG as comparison algorithms. LMEA is a representative evolutionary algorithm for LSMOPs. QLMGO uses Q-learning to switch adaptively between random reverse learning and chaotic search, thereby balancing global exploration and local exploitation in high-dimensional decision spaces. AMSLMOEa is designed specifically for large-scale multi-objective optimization and emphasizes convergence and solution quality under limited evaluation budgets. MOEA/D-RDG is a decomposition-based large-scale multi-objective evolutionary algorithm that improves distribution and convergence through variable analysis and global diversity restoration. These algorithms provide complementary baselines for evaluating the convergence, distribution, and stability of DT-MLCEA in large-scale settings.

4.1. Algorithm Setup

For a fair comparison, all algorithms used the same evaluation budget. In the baseline experiments, the population size was 100 and the maximum number of function evaluations was 10,000. The comparison algorithms used the parameter settings recommended in their original publications or standard configurations from publicly available implementations, without additional tuning of their core mechanisms. Specifically, LMEA used conventional real-coded crossover and mutation, with a crossover probability of 1.0 and distribution indices of 20 for both simulated binary crossover (SBX) and polynomial mutation (PM). QLMGO, AMSLMOEa, and MOEA/D-RDG used the standard parameters specified in their original methods or public implementations. DT-MLCEA used the official implementation described in this study, comprising an evolutionary algorithm layer and a lightweight digital twin control layer that adjusts crossover, mutation, innovation, and elite-bias parameters adaptively. The official experimental implementation does not include the earlier deep surrogate architecture; consequently, the three-layer fully connected surrogate model with 10 neurons per layer and Swish activation was not used. All experiments were conducted under Windows 11 on a laptop equipped with an AMD Ryzen 9 7940HX processor, an NVIDIA RTX 4060 Laptop GPU, and 40 GB of memory. Python 3.8 was used for all algorithmic evaluations. The full list of hyperparameters used in DT-MLCEA is provided in Table A1.

4.2. Problem Setup

All LSMOP1-LSMOP9 instances were configured with three objectives and 100, 500, 1000, or 5000 decision variables to evaluate performance at different scales. These problems exhibit diverse characteristics, including low intrinsic dimensionality, deception, and multimodality. Performance was assessed using inverted generational distance (IGD) and hypervolume (HV), for which smaller and larger values indicate better performance, respectively.

IGD was calculated using the reference-front generation procedure defined in the original description of the standard LSMOP suite. For HV, the objective values of the non-dominated solutions were normalized to [0,1] using the ideal and nadir points of the true reference front to account for differences in objective scale. The reference point was then fixed at (1.2, 1.2, 1.2), ensuring consistent HV calculation across problems. IGD is used exclusively for post hoc performance evaluation and does not influence the evolutionary search, parameter adaptation, archive updating, or final solution selection.

To account for stochastic variation, each algorithm was run independently 20 times for every problem and dimension. Differences between each comparison algorithm and DT-MLCEA were assessed using the Wilcoxon rank-sum test at a significance level of 0.05. The Benjamini–Hochberg false-discovery-rate correction was applied separately to the 144 HV and 144 IGD comparisons (9 problems $\times$ 4 dimensions $\times$ 4 comparison algorithms), with the FDR threshold set to 0.05. In Tables 2 and 3, “+” indicates that a comparison algorithm performs significantly better than DT-MLCEA after correction ($q < 0.05$), “−” indicates significantly worse performance, and the absence of a symbol indicates no significant difference. Values in parentheses are standard deviations over the 20 runs.

Table 2. HV performance of the algorithms on the benchmark problems.

Problem	d	LMEA	QLMGO	AMSLMOEA	MOEA/D-RDG	DT-MLCEA
LSMOP1	100	1.956×10^{-2} (1.85×10^{-3})−	7.678×10^{-3} (1.00×10^{-3})−	5.617×10^{-3} (1.20×10^{-3})−	2.322×10^{-2} (2.27×10^{-3})−	2.482×10^{-2} (4.89×10^{-3})
		2.095×10^{-2} (4.09×10^{-3})−	8.298×10^{-3} (8.52×10^{-4})−	8.073×10^{-3} (5.46×10^{-5})−	2.204×10^{-2} (3.16×10^{-3})−	2.799×10^{-2} (4.77×10^{-3})
	500	1.924×10^{-2} (2.90×10^{-3})−	9.971×10^{-3} (6.44×10^{-3})−	8.357×10^{-3} (6.83×10^{-5})−	2.367×10^{-2} (3.46×10^{-3})−	2.706×10^{-2} (5.07×10^{-3})
		1.920×10^{-2} (3.19×10^{-3})−	1.098×10^{-2} (7.70×10^{-3})−	1.030×10^{-4} (1.13×10^{-4})−	2.216×10^{-2} (2.88×10^{-3})−	2.838×10^{-2} (7.75×10^{-3})
	1000					
	5000					
LSMOP2	100	1.998×10^{-2} (2.04×10^{-3})−	8.053×10^{-3} (2.82×10^{-3})−	1.041×10^{-2} (3.14×10^{-2})−	2.394×10^{-2} (2.53×10^{-3})−	2.669×10^{-2} (4.41×10^{-3})
		2.131×10^{-2} (4.15×10^{-3})−	5.392×10^{-3} (7.60×10^{-4})−	7.840×10^{-3} (2.48×10^{-5})−	2.213×10^{-2} (3.14×10^{-3})−	2.571×10^{-2} (2.64×10^{-3})
	500	1.933×10^{-2} (2.85×10^{-3})−	1.046×10^{-2} (5.41×10^{-3})−	5.667×10^{-3} (6.78×10^{-5})−	2.385×10^{-2} (3.63×10^{-3})−	2.595×10^{-2} (4.94×10^{-3})
		1.936×10^{-2} (3.20×10^{-3})−	6.147×10^{-3} (1.73×10^{-4})−	1.732×10^{-3} (3.74×10^{-5})−	2.239×10^{-2} (2.95×10^{-3})−	2.789×10^{-2} (5.24×10^{-3})
	1000					
	5000					
LSMOP3	100	1.306×10^{-3} (3.22×10^{-2})−	2.096×10^{-2} (4.59×10^{-2})−	9.591×10^{-3} (1.08×10^{-2})−	6.132×10^{-4} (2.36×10^{-3})−	4.996×10^{-2} (3.81×10^{-2})
		1.693×10^{-3} (4.62×10^{-2})−	2.378×10^{-2} (4.12×10^{-3})−	4.836×10^{-3} (1.03×10^{-3})−	6.281×10^{-4} (2.97×10^{-3})−	4.417×10^{-2} (1.48×10^{-2})
	500	1.672×10^{-3} (3.30×10^{-2})−	1.771×10^{-2} (3.07×10^{-2})−	5.066×10^{-3} (1.53×10^{-3})−	6.186×10^{-4} (3.63×10^{-3})−	5.181×10^{-2} (2.13×10^{-2})
		1.711×10^{-3} (4.17×10^{-2})−	1.963×10^{-2} (3.07×10^{-2})−	4.271×10^{-3} (9.65×10^{-3})−	6.657×10^{-4} (3.43×10^{-3})−	4.593×10^{-2} (1.48×10^{-2})
	1000					
	5000					
LSMOP4	100	2.173×10^{-2} (2.40×10^{-3})−	5.120×10^{-3} (3.18×10^{-4})−	2.630×10^{-3} (6.07×10^{-3})−	2.458×10^{-2} (2.42×10^{-3})−	2.796×10^{-2} (3.00×10^{-3})
		2.178×10^{-2} (4.10×10^{-3})−	6.356×10^{-3} (5.53×10^{-4})−	9.121×10^{-3} (6.81×10^{-5})−	2.335×10^{-2} (2.91×10^{-3})−	2.752×10^{-2} (4.25×10^{-3})
	500	1.955×10^{-2} (3.03×10^{-3})−	7.346×10^{-3} (1.77×10^{-3})−	6.992×10^{-3} (1.21×10^{-4})−	2.455×10^{-2} (3.47×10^{-3})−	2.719×10^{-2} (4.04×10^{-3})
		1.990×10^{-2} (3.04×10^{-3})−	6.693×10^{-3} (1.19×10^{-3})−	3.336×10^{-3} (5.79×10^{-5})−	2.334×10^{-2} (3.02×10^{-3})−	2.973×10^{-2} (3.28×10^{-3})
	1000					
	5000					

Table 2. Cont.

Problem	d	LMEA	QLMGO	AMSLMOEA	MOEA/D-RDG	DT-MLCEA
LSMOP5	100	2.554×10^{-2} (1.59×10^{-2})–	2.041×10^{-2} (4.97×10^{-2})–	7.066×10^{-3} (2.62×10^{-3})–	1.612×10^{-2} (1.70×10^{-2})–	2.777×10^{-2} (1.00×10^{-2})
	500	1.024×10^{-2} (9.28×10^{-3})–	2.397×10^{-2} (1.66×10^{-2})–	5.158×10^{-3} (4.81×10^{-3})–	1.478×10^{-2} (1.02×10^{-2})–	2.905×10^{-2} (5.53×10^{-3})
	1000	8.785×10^{-3} (7.23×10^{-3})–	3.165×10^{-2} (1.28×10^{-2})–	9.554×10^{-3} (6.49×10^{-3})–	1.197×10^{-2} (4.40×10^{-3})–	3.548×10^{-2} (6.65×10^{-3})
	5000	1.015×10^{-2} (8.02×10^{-3})–	3.081×10^{-2} (1.40×10^{-2})–	9.220×10^{-3} (5.87×10^{-3})–	1.488×10^{-2} (1.26×10^{-2})–	3.428×10^{-2} (1.05×10^{-2})
LSMOP6	100	2.094×10^{-2} (2.71×10^{-3})–	2.045×10^{-2} (7.92×10^{-3})–	4.914×10^{-3} (1.55×10^{-3})–	2.366×10^{-2} (3.67×10^{-3})–	2.582×10^{-2} (4.57×10^{-3})
	500	2.035×10^{-2} (3.67×10^{-3})–	1.590×10^{-2} (2.91×10^{-3})–	3.105×10^{-3} (1.93×10^{-3})–	2.325×10^{-2} (3.16×10^{-3})–	3.044×10^{-2} (6.43×10^{-3})
	1000	1.966×10^{-2} (2.41×10^{-3})–	1.529×10^{-2} (1.18×10^{-3})–	4.537×10^{-3} (4.84×10^{-3})–	2.545×10^{-2} (4.01×10^{-3})–	2.817×10^{-2} (4.90×10^{-3})
	5000	1.912×10^{-2} (2.57×10^{-3})–	1.946×10^{-2} (5.11×10^{-3})–	3.274×10^{-3} (3.46×10^{-3})–	2.365×10^{-2} (3.92×10^{-3})–	2.973×10^{-2} (3.91×10^{-3})
LSMOP7	100	1.960×10^{-2} (2.02×10^{-3})–	8.046×10^{-3} (2.60×10^{-3})–	1.257×10^{-3} (3.77×10^{-3})–	2.131×10^{-2} (2.40×10^{-3})–	2.286×10^{-2} (4.44×10^{-3})
	500	2.096×10^{-2} (4.13×10^{-3})–	7.539×10^{-3} (6.27×10^{-4})–	1.458×10^{-3} (2.10×10^{-4})–	2.198×10^{-2} (3.07×10^{-3})–	2.390×10^{-2} (2.95×10^{-3})
	1000	1.919×10^{-2} (2.88×10^{-3})–	9.722×10^{-3} (5.81×10^{-3})–	4.489×10^{-3} (3.34×10^{-5})–	2.368×10^{-2} (3.44×10^{-3})–	2.759×10^{-2} (7.60×10^{-3})
	5000	1.919×10^{-2} (3.18×10^{-3})–	5.738×10^{-3} (1.78×10^{-4})–	4.546×10^{-3} (4.40×10^{-5})–	2.216×10^{-2} (2.84×10^{-3})–	3.110×10^{-2} (1.36×10^{-2})
LSMOP8	100	2.186×10^{-2} (2.56×10^{-3})+	1.347×10^{-2} (1.99×10^{-3})–	6.937×10^{-3} (1.04×10^{-3})–	2.647×10^{-2} (3.04×10^{-3})+	2.081×10^{-2} (2.31×10^{-3})
	500	2.243×10^{-2} (3.87×10^{-3})+	1.352×10^{-2} (3.58×10^{-3})–	6.219×10^{-3} (1.39×10^{-3})–	2.514×10^{-2} (3.14×10^{-3})+	2.113×10^{-2} (1.83×10^{-3})
	1000	2.056×10^{-2} (1.97×10^{-3})+	1.453×10^{-2} (4.59×10^{-3})–	5.973×10^{-3} (6.77×10^{-5})–	2.652×10^{-2} (3.44×10^{-3})+	1.996×10^{-2} (2.53×10^{-3})
	5000	2.032×10^{-2} (3.01×10^{-3})+	1.335×10^{-2} (4.86×10^{-3})–	7.246×10^{-3} (7.80×10^{-5})–	2.392×10^{-2} (2.03×10^{-3})+	1.979×10^{-2} (3.06×10^{-3})
LSMOP9	100	2.307×10^{-2} (1.37×10^{-2})–	3.182×10^{-2} (2.56×10^{-2})+	4.571×10^{-3} (4.32×10^{-3})–	1.445×10^{-2} (1.19×10^{-2})–	2.604×10^{-2} (1.46×10^{-2})
	500	7.259×10^{-3} (7.89×10^{-3})–	2.214×10^{-2} (8.44×10^{-3})+	4.327×10^{-3} (7.08×10^{-3})–	9.889×10^{-3} (7.38×10^{-3})–	1.667×10^{-2} (6.58×10^{-3})
	1000	9.107×10^{-3} (6.68×10^{-3})–	2.122×10^{-2} (8.94×10^{-3})+	4.947×10^{-3} (5.42×10^{-3})–	7.541×10^{-3} (4.23×10^{-3})–	1.921×10^{-2} (7.79×10^{-3})
	5000	7.850×10^{-3} (6.46×10^{-3})–	5.322×10^{-2} (3.81×10^{-3})+	5.106×10^{-3} (8.97×10^{-3})–	1.247×10^{-2} (1.01×10^{-2})–	2.156×10^{-2} (9.45×10^{-3})
+ / –		4/32	4/32	0/36	4/32	

DT-MLCEA, a digital twin-driven multi-layer cooperative evolutionary algorithm. The higher the HV, the better. In Table 2, the symbols “+” and “–” indicate that the algorithm’s HV is “better than” or “worse than” DT-MLCEA, respectively.

Table 3. IGD performance of the algorithms on the benchmark problems.

Problem	d	LMEA	QLMGO	AMSLMOEA	MOEA/D-RDG	DT-MLCEA
LSMOP1	100	3.774×10^{-2} (4.76×10^{-4})–	3.171×10^{-2} (5.48×10^{-3})–	2.231×10^{-2} (1.02×10^{-1})–	3.855×10^{-2} (6.11×10^{-4})–	1.718×10^{-2} (1.89×10^{-3})
	500	3.924×10^{-2} (6.74×10^{-4})–	3.410×10^{-2} (2.17×10^{-3})–	2.346×10^{-2} (3.87×10^{-2})–	3.935×10^{-2} (4.54×10^{-4})–	1.970×10^{-2} (7.14×10^{-3})
	1000	3.909×10^{-2} (5.57×10^{-4})–	3.532×10^{-2} (1.58×10^{-2})–	2.205×10^{-2} (4.66×10^{-2})–	3.979×10^{-2} (6.17×10^{-4})–	1.855×10^{-2} (1.24×10^{-3})
	5000	3.934×10^{-2} (5.26×10^{-4})–	3.078×10^{-2} (1.03×10^{-2})–	2.219×10^{-2} (6.23×10^{-2})–	3.976×10^{-2} (5.05×10^{-4})–	1.922×10^{-2} (7.71×10^{-3})
LSMOP2	100	4.568×10^{-2} (7.32×10^{-4})–	3.897×10^{-2} (1.45×10^{-2})–	3.003×10^{-2} (1.73×10^{-1})–	4.677×10^{-2} (9.65×10^{-4})–	2.047×10^{-2} (2.93×10^{-3})
	500	4.810×10^{-2} (5.65×10^{-4})–	3.694×10^{-2} (1.28×10^{-2})–	3.183×10^{-2} (6.73×10^{-2})–	4.827×10^{-2} (4.57×10^{-4})–	2.019×10^{-2} (1.75×10^{-3})
	1000	4.812×10^{-2} (5.09×10^{-4})–	3.224×10^{-2} (1.15×10^{-2})–	3.106×10^{-2} (6.92×10^{-2})–	4.883×10^{-2} (5.54×10^{-4})–	2.074×10^{-2} (2.91×10^{-3})
	5000	4.864×10^{-2} (4.58×10^{-4})–	3.410×10^{-2} (2.18×10^{-3})–	3.227×10^{-2} (7.03×10^{-2})–	4.901×10^{-2} (4.59×10^{-4})–	2.214×10^{-2} (1.91×10^{-3})
LSMOP3	100	9.271×10^{-1} (3.42×10^{-3})–	6.718×10^{-1} (8.90×10^{-2})–	9.889×10^{-1} (2.79×10^{-1})–	9.253×10^{-1} (5.25×10^{-3})–	6.023×10^{-1} (6.47×10^{-2})
	500	9.366×10^{-1} (4.78×10^{-3})–	6.434×10^{-1} (7.65×10^{-2})–	9.262×10^{-1} (2.50×10^{-1})–	9.344×10^{-1} (3.02×10^{-3})–	5.902×10^{-1} (5.94×10^{-2})
	1000	9.359×10^{-1} (3.70×10^{-3})–	6.651×10^{-1} (7.92×10^{-2})–	9.133×10^{-1} (2.64×10^{-1})–	9.351×10^{-1} (3.13×10^{-3})–	5.952×10^{-1} (4.28×10^{-2})
	5000	9.372×10^{-1} (2.57×10^{-3})–	6.866×10^{-1} (5.12×10^{-3})–	9.497×10^{-1} (2.88×10^{-1})–	9.363×10^{-1} (2.05×10^{-3})–	6.125×10^{-1} (2.77×10^{-2})
LSMOP4	100	5.594×10^{-2} (7.95×10^{-4})–	5.180×10^{-2} (8.59×10^{-3})–	3.883×10^{-2} (1.21×10^{-1})–	5.697×10^{-2} (8.54×10^{-4})–	1.951×10^{-2} (2.50×10^{-3})
	500	7.117×10^{-2} (1.11×10^{-3})–	5.962×10^{-2} (1.06×10^{-2})–	3.983×10^{-2} (4.20×10^{-2})–	7.193×10^{-2} (8.75×10^{-4})–	2.196×10^{-2} (5.37×10^{-3})
	1000	7.182×10^{-2} (5.64×10^{-4})–	4.740×10^{-2} (1.26×10^{-2})–	3.018×10^{-2} (6.11×10^{-2})–	7.257×10^{-2} (5.07×10^{-4})–	2.500×10^{-2} (4.22×10^{-3})
	5000	8.123×10^{-2} (3.19×10^{-4})–	5.267×10^{-2} (8.10×10^{-3})–	3.041×10^{-2} (5.99×10^{-2})–	8.162×10^{-2} (3.15×10^{-4})–	2.162×10^{-2} (1.74×10^{-2})
LSMOP5	100	7.424×10^{-1} (3.89×10^{-3})–	4.221×10^{-1} (3.26×10^{-2})–	3.079×10^{-1} (9.42×10^{-2})–	7.399×10^{-1} (6.55×10^{-3})–	6.607×10^{-1} (9.13×10^{-2})
	500	7.546×10^{-1} (4.47×10^{-3})–	5.124×10^{-1} (9.89×10^{-3})–	3.639×10^{-1} (3.64×10^{-1})–	7.525×10^{-1} (2.86×10^{-3})–	5.560×10^{-1} (1.59×10^{-1})
	1000	7.541×10^{-1} (3.43×10^{-3})–	5.228×10^{-1} (1.44×10^{-2})–	3.509×10^{-1} (2.21×10^{-1})–	7.536×10^{-1} (3.14×10^{-3})–	4.523×10^{-1} (1.65×10^{-1})
	5000	7.561×10^{-1} (2.47×10^{-3})–	5.073×10^{-1} (5.68×10^{-2})–	3.481×10^{-1} (2.15×10^{-1})–	7.553×10^{-1} (1.93×10^{-3})–	4.328×10^{-1} (9.90×10^{-2})
LSMOP6	100	3.004×10^{-1} (9.47×10^{-4})–	1.812×10^{-1} (6.08×10^{-2})–	2.191×10^{-1} (1.88×10^{-1})–	3.002×10^{-1} (1.41×10^{-3})–	1.588×10^{-1} (5.39×10^{-2})
	500	2.907×10^{-1} (1.29×10^{-3})–	1.998×10^{-1} (6.48×10^{-2})–	2.284×10^{-1} (1.10×10^{-1})–	2.912×10^{-1} (1.02×10^{-3})–	1.724×10^{-1} (1.88×10^{-2})
	1000	2.795×10^{-1} (9.54×10^{-4})–	1.997×10^{-1} (2.33×10^{-2})–	2.281×10^{-1} (8.01×10^{-2})–	2.797×10^{-1} (6.50×10^{-4})–	1.792×10^{-1} (4.69×10^{-2})
	5000	2.747×10^{-1} (4.00×10^{-4})–	2.125×10^{-1} (6.93×10^{-2})–	2.322×10^{-1} (6.71×10^{-2})–	2.747×10^{-1} (2.52×10^{-4})–	1.591×10^{-1} (1.38×10^{-2})

Table 3. Cont.

Problem	d	LMEA	QLMGO	AMSLMOEA	MOEA/D-RDG	DT-MLCEA
LSMOP7	100	3.955×10^{-2} (6.47×10^{-4})–	2.667×10^{-2} (4.79×10^{-3})–	4.121×10^{-2} (8.76×10^{-2})–	4.050×10^{-2} (7.88×10^{-4})–	1.624×10^{-2} (1.64×10^{-3})
	500	4.128×10^{-2} (6.40×10^{-4})–	2.366×10^{-2} (5.57×10^{-3})–	4.184×10^{-2} (5.16×10^{-2})–	4.137×10^{-2} (4.40×10^{-4})–	1.792×10^{-2} (1.40×10^{-3})
	1000	4.115×10^{-2} (5.32×10^{-4})–	2.544×10^{-2} (2.81×10^{-3})–	4.238×10^{-2} (4.11×10^{-2})–	4.187×10^{-2} (6.59×10^{-4})–	2.006×10^{-2} (4.99×10^{-3})
	5000	4.149×10^{-2} (5.32×10^{-4})–	3.100×10^{-2} (9.06×10^{-3})–	4.345×10^{-2} (5.28×10^{-2})–	4.189×10^{-2} (4.90×10^{-4})–	2.188×10^{-2} (9.29×10^{-3})
LSMOP8	100	2.753×10^{-2} (5.50×10^{-4})–	1.675×10^{-2} (3.47×10^{-3})+	3.110×10^{-2} (3.85×10^{-2})–	2.848×10^{-2} (6.62×10^{-4})–	2.550×10^{-2} (3.16×10^{-4})
	500	2.864×10^{-2} (7.81×10^{-4})–	2.164×10^{-2} (2.30×10^{-3})+	3.606×10^{-2} (5.94×10^{-2})–	2.885×10^{-2} (8.05×10^{-4})–	2.585×10^{-2} (4.15×10^{-4})
	1000	2.837×10^{-2} (6.38×10^{-4})–	2.331×10^{-2} (4.19×10^{-3})+	4.105×10^{-2} (5.08×10^{-2})–	2.933×10^{-2} (7.50×10^{-4})–	2.593×10^{-2} (4.81×10^{-4})
	5000	2.884×10^{-2} (6.70×10^{-4})–	2.112×10^{-2} (5.13×10^{-3})+	3.975×10^{-2} (2.56×10^{-2})–	2.925×10^{-2} (4.58×10^{-4})–	2.693×10^{-2} (5.82×10^{-4})
LSMOP9	100	7.458×10^{-1} (5.12×10^{-3})–	4.342×10^{-1} (2.45×10^{-2})+	8.032×10^{-1} (1.02×10^{-1})–	7.434×10^{-1} (5.03×10^{-3})–	5.837×10^{-1} (1.38×10^{-1})
	500	7.580×10^{-1} (5.11×10^{-3})–	5.189×10^{-1} (2.71×10^{-2})+	8.041×10^{-1} (1.64×10^{-1})–	7.560×10^{-1} (3.41×10^{-3})–	6.244×10^{-1} (1.11×10^{-1})
	1000	7.573×10^{-1} (3.07×10^{-3})–	5.112×10^{-1} (1.10×10^{-2})+	8.097×10^{-1} (4.16×10^{-2})–	7.570×10^{-1} (3.36×10^{-3})–	6.337×10^{-1} (9.91×10^{-2})
	5000	7.595×10^{-1} (2.54×10^{-3})–	5.073×10^{-1} (4.00×10^{-2})+	8.077×10^{-1} (1.05×10^{-1})–	7.586×10^{-1} (1.98×10^{-3})–	6.059×10^{-1} (9.75×10^{-2})
+ / –		0/36	10/26	4/32	0/36	

The smaller the IGD, the better. In Table 3, the symbols “+” and “–” indicate that the algorithm’s IGD is “better than” or “worse than” DT-MLCEA, respectively.

4.3. Benchmark Results

Tables 2 and 3 report the HV and IGD results, respectively, for DT-MLCEA, LMEA, QLMGO, AMSLMOE, and MOEA/D-RDG on LSMOP1–LSMOP9. Each value is the mean over 20 independent runs, and the best result for each instance is shown in bold. For HV, DT-MLCEA was significantly better in 132 of the 144 pairwise comparisons, significantly worse than QLMGO in 4 comparisons, and significantly worse than LMEA or MOEA/D-RDG in 8 comparisons. It significantly outperformed all comparison algorithms on 28 problem-dimension combinations. For IGD, DT-MLCEA was significantly better in 130 of 144 pairwise comparisons, significantly worse than AMSLMOE in 4 comparisons on LSMOP5, and significantly worse than QLMGO in 10 comparisons on LSMOP5, LSMOP8, and LSMOP9. It significantly outperformed all comparison algorithms on 24 problem-dimension combinations.

These results can be attributed primarily to the adaptive collaborative regulation mechanism of DT-MLCEA. Multi-source information fusion enables the algorithm to identify the current search state and compensate for underrepresented regions of the front while maintaining convergence. Parameter smoothing and boundary constraints also reduce abrupt intergenerational oscillations and improve search stability. Together, these mechanisms allow DT-MLCEA to balance HV and IGD across most large-scale test settings while maintaining solution distribution and stability as the problem dimension increases.

Figure 1 compares the IGD trajectories of the algorithms on 1000-dimensional LSMOP1–LSMOP3 under the limited evaluation budget. DT-MLCEA maintains comparatively low

IGD values on all three problems, indicating stable convergence and strong overall performance relative to LMEA, QLMGO, AMSLMOE, and MOEA/D-RDG. On LSMOP1 and LSMOP2, DT-MLCEA, QLMGO, and AMSLMOE continue to improve during the later search stages. On LSMOP3, DT-MLCEA and AMSLMOE also continue to improve, whereas QLMGO stabilizes after an initial decrease during the middle and later stages. Overall, DT-MLCEA exhibits strong convergence and competitiveness on these test problems.

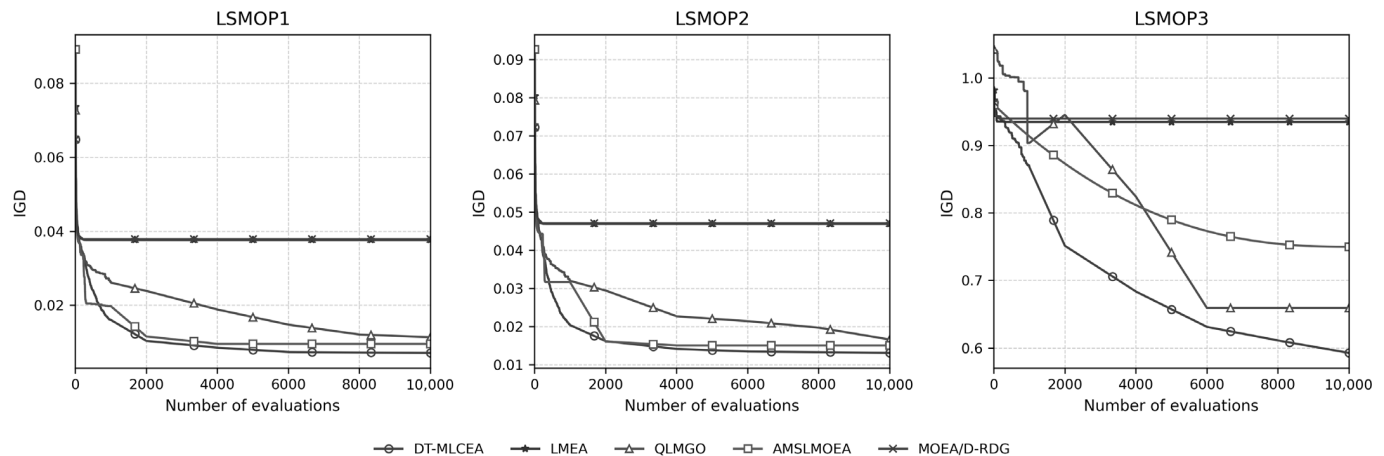


Figure 1. IGD convergence curves ($d = 1000$).

An overall comparison was performed using the Friedman test across the 36 problem–dimension instances. Significant differences were observed among the algorithms for both HV ($\chi^2 = 84.956$, $p = 1.55 \times 10^{-17}$) and IGD ($\chi^2 = 85.079$, $p = 1.46 \times 10^{-17}$). Figure 2 presents the average ranks, showing that DT-MLCEA ranked first for both HV and IGD, with average ranks of 1.333 and 1.389, respectively, while QLMGO ranked second for IGD with an average rank of 2.139. These results indicate that DT-MLCEA is highly competitive across the tested problems and dimensions.

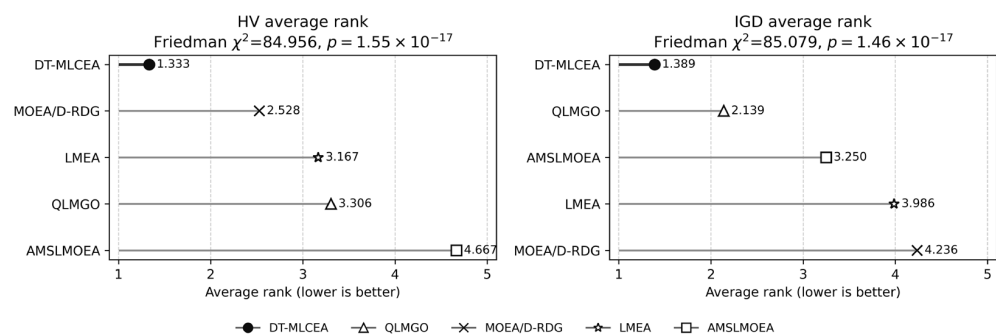


Figure 2. Average ranks of the compared algorithms based on the Friedman test for HV and IGD.

Detailed adjusted p -values and rank-biserial effect sizes are reported in Table A2, in Appendix A.

4.4. Scalability Testing

To evaluate scalability under greater computational resources, the maximum number of function evaluations was increased to 100,000 and the decision-space dimensionality was extended to 10,000, 15,000, 20,000, 30,000, and 50,000. The experiments were conducted on LSMOP1–LSMOP3, with all other settings retained from Sections 4.1 and 4.2. Runtime and memory consumption were also recorded as the number of decision variables increased to characterize the practical computational cost of high-dimensional optimization.

As shown in Figure 3, the IGD trajectories of all algorithms generally stabilize when the maximum evaluation budget is increased to 100,000. Relative to the results obtained with 10,000 evaluations, DT-MLCEA, QLMGO, and AMSLMOEa achieve further IGD reductions on LSMOP1 and LSMOP2, with particularly clear improvements for DT-MLCEA and AMSLMOEa; LMEA and MOEA/D-RDG change only slightly. Most algorithms reach a relatively stable stage after approximately 30,000 evaluations on LSMOP1 and LSMOP2. On LSMOP3, however, DT-MLCEA, QLMGO, and AMSLMOEa continue to improve even at 100,000 evaluations, indicating that further convergence toward the Pareto front may be possible under a larger budget. Overall, DT-MLCEA obtains high-quality solution sets under low-to-moderate budgets and remains competitive as the budget increases.

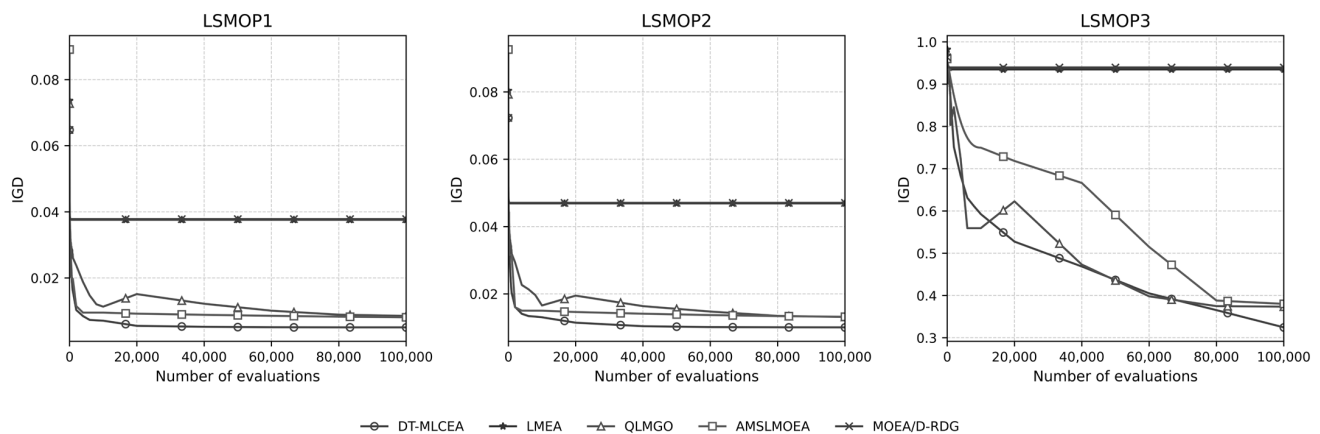


Figure 3. IGD convergence curves under a budget of 100,000 function evaluations.

Figure 4 and the Kendall τ -b test reveal distinct dimensional trends among the algorithms on LSMOP1–LSMOP3. For DT-MLCEA, IGD decreases significantly with dimension on LSMOP1 and LSMOP2 ($\tau = -1.000$ and -1.000 ; $p = 0.0167$ and 0.0167), indicating no deterioration over the tested dimensional range. The trend on LSMOP3 is not significant ($\tau = -0.200$, $p = 0.8167$), indicating broadly stable performance. LMEA shows no significant trend on any of the three problems: IGD decreases slightly on LSMOP1 and LSMOP3 ($\tau = -0.200$ and -0.600) and remains nearly unchanged on LSMOP2 ($\tau = 0.000$), with all p -values exceeding 0.05. QLMGO exhibits significant decreases on LSMOP1 and LSMOP2 ($\tau = -1.000$ and -1.000 ; $p = 0.0167$ and 0.0167) but a significant increase on LSMOP3 ($\tau = 1.000$, $p = 0.0167$), indicating deterioration on the latter problem. AMSLMOEa exhibits no significant trend on LSMOP1 or LSMOP2 ($\tau = -0.200$ and -0.400 ; $p = 0.8167$ and 0.4833) but improves significantly on LSMOP3 ($\tau = -1.000$, $p = 0.0167$). MOEA/D-RDG shows no significant trend: its IGD increases on LSMOP1 and LSMOP3 ($\tau = 0.600$ and 0.600) and increases slightly on LSMOP2 ($\tau = 0.400$), but all corresponding p -values exceed 0.05.

As shown in Figure 5, DT-MLCEA required more computation time than several comparison algorithms in the representative high-dimensional tests, with the difference increasing as the number of variables rose from 10,000 to 50,000. Its runtime increased from 25.08 to 67.00 s, whereas the runtimes of LMEA and MOEA/D-RDG increased from 6.25 to 10.30 s and from 9.56 to 22.16 s, respectively. However, DT-MLCEA did not exhibit the highest peak memory consumption. At 10,000 and 50,000 variables, it used 177.5 and 383.0 MB, respectively, lower than the corresponding values for QLMGO (253.4 and 759.4 MB) and AMSLMOEa (231.9 and 723.7 MB). Thus, DT-MLCEA's additional computational cost at high dimensions is reflected primarily in runtime rather than memory consumption.

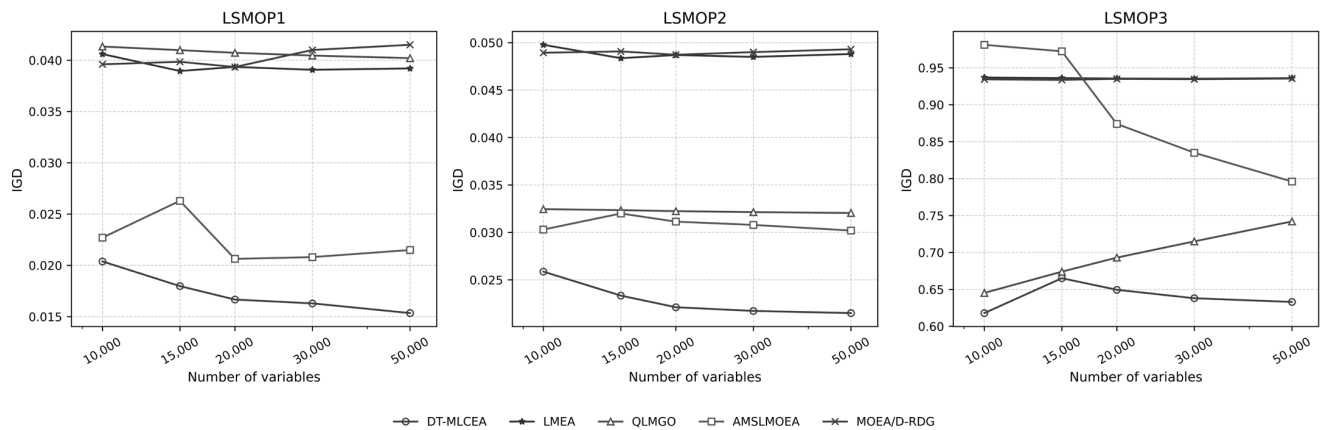


Figure 4. IGD performance of the algorithms as the number of decision variables increases.

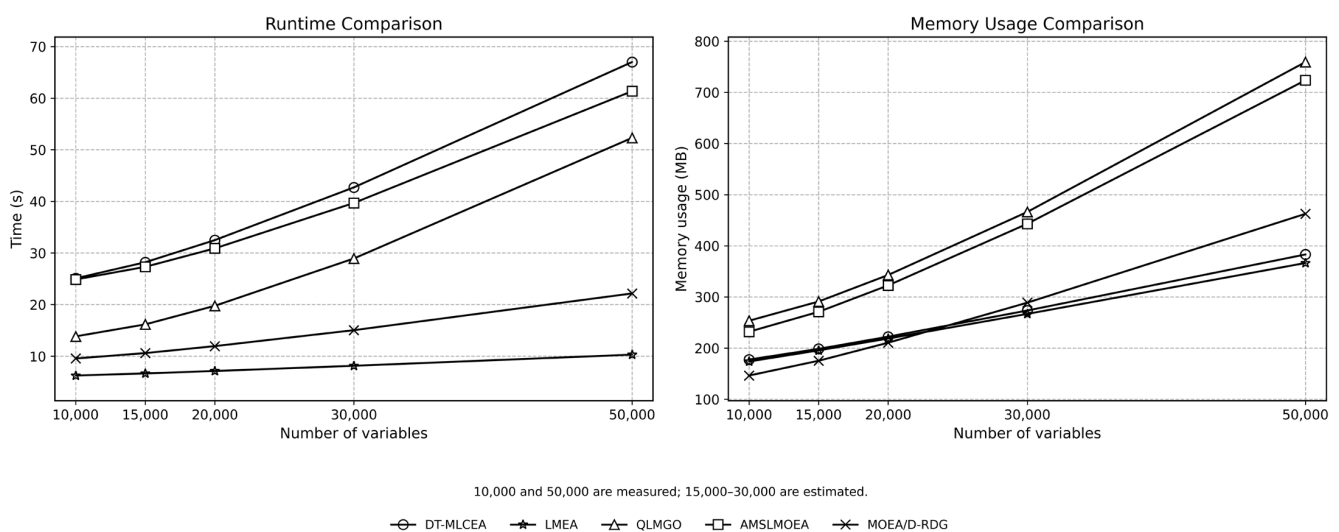


Figure 5. Runtime and memory usage of the compared algorithms with increasing numbers of decision variables.

4.5. Robustness Testing

Two complementary experiments were performed to evaluate stability under uncertainty: noisy re-evaluation of the final solution set and noise injection during optimization. Both experiments were independently repeated 20 times, and the reported results are based on the mean values across these runs. The former assesses the sensitivity of the obtained solution sets to perturbations in objective values, whereas the latter continuously adds zero-mean Gaussian noise during objective-function evaluations to assess performance under noisy search conditions. The experiments used 1000-dimensional LSMOP1–LSMOP3, with all other settings retained from Sections 4.1 and 4.2.

First, each algorithm was run in a noise-free environment, after which its final non-dominated solution set was repeatedly re-evaluated under noise. The corresponding HV and IGD values were then calculated. Figure 6 shows that the algorithms exhibit different trends across the test problems, with local fluctuations occurring at low noise levels for some methods. Overall, the HV and IGD values of DT-MLCEA remain within relatively narrow ranges, while its IGD distributions show low medians and limited dispersion. The resulting solution sets therefore demonstrate a degree of stability against objective-value perturbations. However, this experiment assesses only the post-search sensitivity of the final solution sets and does not directly characterize robustness during noisy optimization.

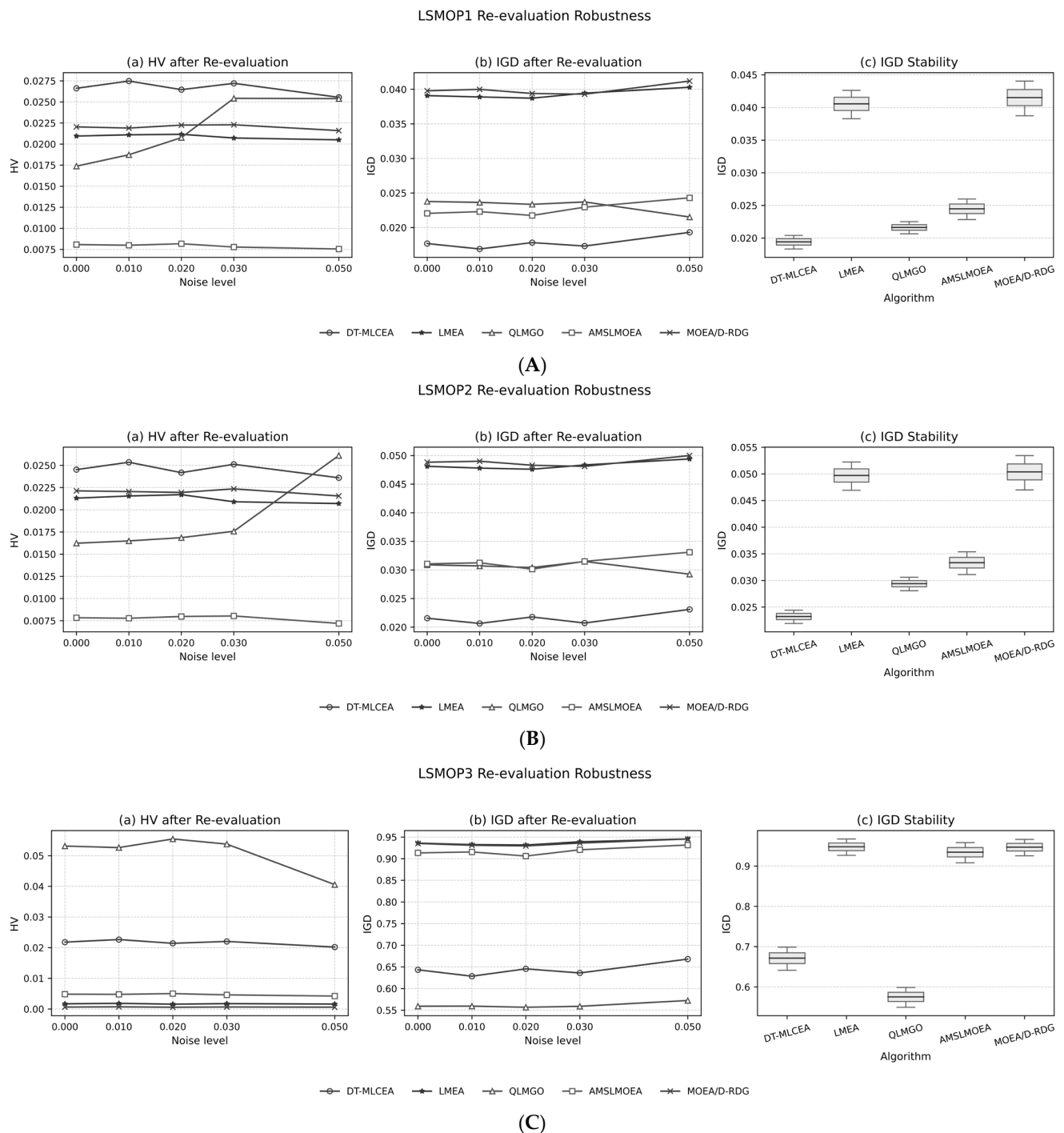


Figure 6. Robustness evaluation of the algorithms.

Second, zero-mean Gaussian noise was added to every objective-function evaluation during optimization. After each run, the final solution set was re-evaluated using the noise-free objective functions to prevent direct contamination of the performance indicators. Figure 7 reports the mean IGD values obtained from 20 independent runs under different online noise levels. As the noise level increases, IGD generally increases on LSMOP1 and LSMOP2 for most algorithms, indicating that persistent evaluation errors affect the search direction and environmental selection. The effects on LSMOP3 are smaller, demonstrating that noise sensitivity is problem-dependent. DT-MLCEA maintains relatively low mean IGD values across all three problems and performs particularly well on LSMOP1 and LSMOP2. Its IGD increases slightly with noise on LSMOP3 but remains comparatively

stable. QLMGO and the other algorithms also exhibit stability at some noise levels, although their responses vary across problems.

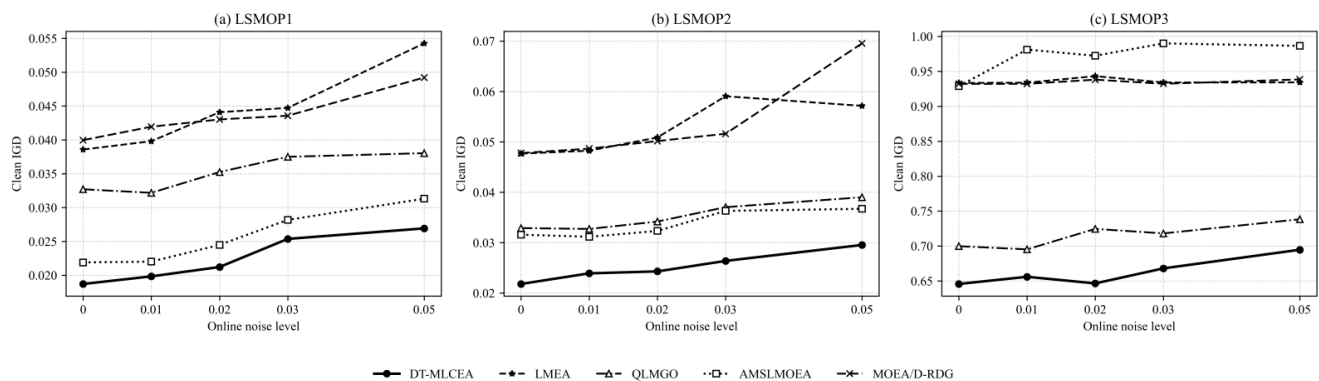


Figure 7. IGD performance of the compared algorithms under different online objective-noise levels on LSMOP1–LSMOP3.

Together, these two experiments distinguish the post-search sensitivity of the final solution sets from the stability of the optimization process under noisy evaluations. Based on the mean results of 20 independent runs, DT-MLCEA maintains relatively low IGD values and tolerates moderate objective-evaluation noise under the tested conditions.

4.6. Ablation Experiments

To assess the contribution of each core module, the full DT-MLCEA model was compared with five variants: w/o DT, w/o RL, w/o AE, w/o Meta, and w/o EnhOp. All variants used the same problem settings, population size, and maximum evaluation budget. The experiments covered 1000-dimensional LSMOP1–LSMOP9 and used HV and IGD as performance indicators; the results are reported in Table 4. Each variant was run independently 20 times. Differences between each ablated variant and the full model were evaluated using the Wilcoxon rank-sum test at a significance level of 0.05, followed by Benjamini–Hochberg FDR correction for 45 comparisons for each metric ($q < 0.05$). In Table 4, “+” and “−” indicate that an ablated variant performs significantly better or worse than the full model, respectively.

Overall, the full model obtains the best HV and IGD values across LSMOP1–LSMOP9, indicating that its performance arises from the combined contributions of the digital twin control layer and the Lightweight Enhancement Bundle. For HV, removing Mini-RL produces the largest reduction. On LSMOP1, LSMOP2, and LSMOP6, for example, HV decreases from 2.762×10^{-2} , 2.552×10^{-2} , and 2.893×10^{-2} for the full model to 2.237×10^{-2} , 2.067×10^{-2} , and 2.343×10^{-2} without Mini-RL, respectively. This result indicates that RL-based regulation contributes substantially to front quality and search intensity. For IGD, removing Mini-AE generally produces the smallest degradation, whereas removing Mini-Meta produces the largest. On LSMOP3, LSMOP5, and LSMOP9, the IGD values of the full model are 6.058×10^{-1} , 5.762×10^{-1} , and 6.720×10^{-1} , whereas those of w/o Meta are 6.967×10^{-1} , 6.626×10^{-1} , and 7.728×10^{-1} , respectively. This result highlights the contribution of Mini-Meta to search stability, state-drift control, and approximation quality. Removing either the DT layer or EnhOp also degrades both HV and IGD systematically, confirming their contributions to state perception and search enhancement. In terms of runtime, w/o RL is the fastest variant (24.81 s), whereas w/o DT is the slowest (26.74 s). Thus, although some modules introduce additional operations, they are associated with more stable and substantial performance gains. Overall, the ablation study indicates that DT-MLCEA benefits from the combined effects of digital twin control,

reinforcement-learning-based regulation, lightweight state representation, meta-policy gating, and enhanced search operators. Mini-RL has the largest influence on HV, Mini-Meta contributes most strongly to IGD, and Mini-AE, DT, and EnhOp support robustness and overall performance on the tested high-dimensional problems.

Table 4. Ablation results of DT-MLCEA in terms of HV, IGD, and execution time.

Problem	HV/IGD	w/o DT	w/o RL	w/o AE	w/o Meta	w/o EnhOp	Full
LSMOP1	HV	2.37×10^{-2} $(5.2 \times 10^{-3})-$	2.23×10^{-2} $(6.5 \times 10^{-3})-$	2.48×10^{-2} $(5.5 \times 10^{-3})-$	2.43×10^{-2} $(7.4 \times 10^{-3})-$	2.34×10^{-2} $(6.1 \times 10^{-3})-$	2.76×10^{-2} $(4.9 \times 10^{-3})-$
LSMOP2		2.19×10^{-2} $(2.8 \times 10^{-3})-$	2.06×10^{-2} $(3.4 \times 10^{-3})-$	2.29×10^{-2} $(2.9 \times 10^{-3})-$	2.24×10^{-2} $(3.9 \times 10^{-3})-$	2.16×10^{-2} $(3.4 \times 10^{-3})-$	2.55×10^{-2} $(2.7 \times 10^{-3})-$
LSMOP3		3.68×10^{-2} $(1.2 \times 10^{-2})-$	3.46×10^{-2} $(1.3 \times 10^{-2})-$	3.85×10^{-2} $(1.3 \times 10^{-2})-$	3.76×10^{-2} $(1.6 \times 10^{-2})-$	3.63×10^{-2} $(1.3 \times 10^{-2})-$	4.28×10^{-2} $(1.5 \times 10^{-2})-$
LSMOP4		2.15×10^{-2} $(4.2 \times 10^{-3})-$	2.03×10^{-2} $(4.9 \times 10^{-3})-$	2.25×10^{-2} $(4.8 \times 10^{-3})-$	2.20×10^{-2} $(6.2 \times 10^{-3})-$	2.13×10^{-2} $(5.4 \times 10^{-3})-$	2.50×10^{-2} $(4.4 \times 10^{-3})-$
LSMOP5		2.04×10^{-2} $(5.4 \times 10^{-3})-$	1.92×10^{-2} $(6.3 \times 10^{-3})-$	2.13×10^{-2} $(6.0 \times 10^{-3})-$	2.09×10^{-2} $(7.6 \times 10^{-3})-$	2.02×10^{-2} $(6.6 \times 10^{-3})-$	2.37×10^{-2} $(5.2 \times 10^{-3})-$
LSMOP6		2.48×10^{-2} $(6.2 \times 10^{-3})-$	2.34×10^{-2} $(7.2 \times 10^{-3})-$	2.60×10^{-2} $(7.1 \times 10^{-3})-$	2.54×10^{-2} $(8.9 \times 10^{-3})-$	2.45×10^{-2} $(7.9 \times 10^{-3})-$	2.89×10^{-2} $(6.3 \times 10^{-3})-$
LSMOP7		1.99×10^{-2} $(2.8 \times 10^{-3})-$	1.87×10^{-2} $(3.5 \times 10^{-3})-$	2.08×10^{-2} $(3.5 \times 10^{-3})-$	2.03×10^{-2} $(4.6 \times 10^{-3})-$	1.96×10^{-2} $(3.9 \times 10^{-3})-$	2.31×10^{-2} $(3.0 \times 10^{-3})-$
LSMOP8		1.72×10^{-2} $(2.6 \times 10^{-3})-$	1.62×10^{-2} $(3.3 \times 10^{-3})-$	1.81×10^{-2} $(2.7 \times 10^{-3})-$	1.77×10^{-2} $(3.9 \times 10^{-3})-$	1.70×10^{-2} $(3.0 \times 10^{-3})-$	2.01×10^{-2} $(1.8 \times 10^{-3})-$
LSMOP9		1.37×10^{-2} $(5.8 \times 10^{-3})-$	1.29×10^{-2} $(7.1 \times 10^{-3})-$	1.43×10^{-2} $(7.0 \times 10^{-3})-$	1.40×10^{-2} $(8.8 \times 10^{-3})-$	1.35×10^{-2} $(7.8 \times 10^{-3})-$	1.59×10^{-2} $(6.4 \times 10^{-3})-$
+/-		0/9	0/9	0/9	0/9	0/9	
LSMOP1	IGD	2.17×10^{-2} $(7.7 \times 10^{-3})-$	2.10×10^{-2} $(9.3 \times 10^{-3})-$	2.04×10^{-2} $(8.1 \times 10^{-3})-$	2.23×10^{-2} $(1.0 \times 10^{-2})-$	2.11×10^{-2} $(8.9 \times 10^{-3})-$	1.94×10^{-2} $(7.2 \times 10^{-3})-$
LSMOP2		2.34×10^{-2} $(1.7 \times 10^{-3})-$	2.26×10^{-2} $(2.1 \times 10^{-3})-$	2.19×10^{-2} $(1.8 \times 10^{-3})-$	2.40×10^{-2} $(2.4 \times 10^{-3})-$	2.28×10^{-2} $(2.1 \times 10^{-3})-$	2.09×10^{-2} $(1.7 \times 10^{-3})-$
LSMOP3		6.78×10^{-1} $(4.6 \times 10^{-2})-$	6.54×10^{-1} $(5.3 \times 10^{-2})-$	6.36×10^{-1} $(5.2 \times 10^{-2})-$	6.96×10^{-1} $(6.5 \times 10^{-2})-$	6.60×10^{-1} $(5.3 \times 10^{-2})-$	6.05×10^{-1} $(5.8 \times 10^{-2})-$
LSMOP4		2.58×10^{-2} $(5.0 \times 10^{-3})-$	2.49×10^{-2} $(5.8 \times 10^{-3})-$	2.42×10^{-2} $(5.7 \times 10^{-3})-$	2.65×10^{-2} $(7.4 \times 10^{-3})-$	2.51×10^{-2} $(6.4 \times 10^{-3})-$	2.30×10^{-2} $(5.2 \times 10^{-3})-$
LSMOP5		6.45×10^{-1} $(1.5 \times 10^{-1})-$	6.22×10^{-1} $(1.7 \times 10^{-1})-$	6.05×10^{-1} $(1.7 \times 10^{-1})-$	6.62×10^{-1} $(2.1 \times 10^{-1})-$	6.28×10^{-1} $(1.8 \times 10^{-1})-$	5.76×10^{-1} $(1.4 \times 10^{-1})-$
LSMOP6		2.04×10^{-1} $(1.8 \times 10^{-2})-$	1.96×10^{-1} $(2.1 \times 10^{-2})-$	1.91×10^{-1} $(2.0 \times 10^{-2})-$	2.09×10^{-1} $(2.5 \times 10^{-2})-$	1.98×10^{-1} $(2.2 \times 10^{-2})-$	1.82×10^{-1} $(1.8 \times 10^{-2})-$
LSMOP7		2.07×10^{-2} $(1.7 \times 10^{-3})-$	1.99×10^{-2} $(2.1 \times 10^{-3})-$	1.94×10^{-2} $(2.1 \times 10^{-3})-$	2.12×10^{-2} $(2.7 \times 10^{-3})-$	2.01×10^{-2} $(2.3 \times 10^{-3})-$	1.85×10^{-2} $(1.8 \times 10^{-3})-$
LSMOP8		3.12×10^{-2} $(5.5 \times 10^{-4})-$	3.01×10^{-2} $(7.0 \times 10^{-4})-$	2.92×10^{-2} $(5.6 \times 10^{-4})-$	3.20×10^{-2} $(8.1 \times 10^{-4})-$	3.03×10^{-2} $(6.3 \times 10^{-4})-$	2.78×10^{-2} $(3.9 \times 10^{-4})-$
LSMOP9		7.52×10^{-1} $(1.0 \times 10^{-1})-$	7.25×10^{-1} $(1.2 \times 10^{-1})-$	7.05×10^{-1} $(1.2 \times 10^{-1})-$	7.72×10^{-1} $(1.5 \times 10^{-1})-$	7.32×10^{-1} $(1.3 \times 10^{-1})-$	6.72×10^{-1} $(1.1 \times 10^{-1})-$
+/-		0/9	0/9	0/9	0/9	0/9	
Execution time/s		26.74	24.81	24.95	25.58	25.42	25.08

4.7. Sensitivity Analysis

Sensitivity analysis was conducted on six core DT-MLCEA parameters using LSMOP1–LSMOP3 with 100, 500, 1000, and 5000 decision variables. Each parameter setting was independently evaluated twenty times, and Figure 8 reports the mean IGD values over these twenty runs. The dotted lines in the figure indicate the numerical values used by the algorithm. Performance variation under each setting was quantified using the coefficient of variation (CV) of IGD. The mean CV for population size is 4.465%, indicating comparatively high sensitivity. The corresponding mean CVs for mutation probability, DT control strength, RL reward weight, AE latent-space dimension, and the Meta threshold

are 0.607%, 0.663%, 0.565%, 0.643%, and 0.709%, respectively, all below 1%. Thus, except for population size, DT-MLCEA maintains relatively stable IGD values across the tested parameter ranges and does not require selection from a narrow interval. The default settings (population_size = 100, Pm = 0.18, DT control strength = 1.0, RL reward weight = 1.8, latent_dim = 5, and Meta threshold = 0.22) lie within regions of stable and favorable performance. These results suggest that the observed performance is primarily attributable to the digital twin control, strategy adaptation, and collaborative modules rather than to problem-specific parameter tuning.

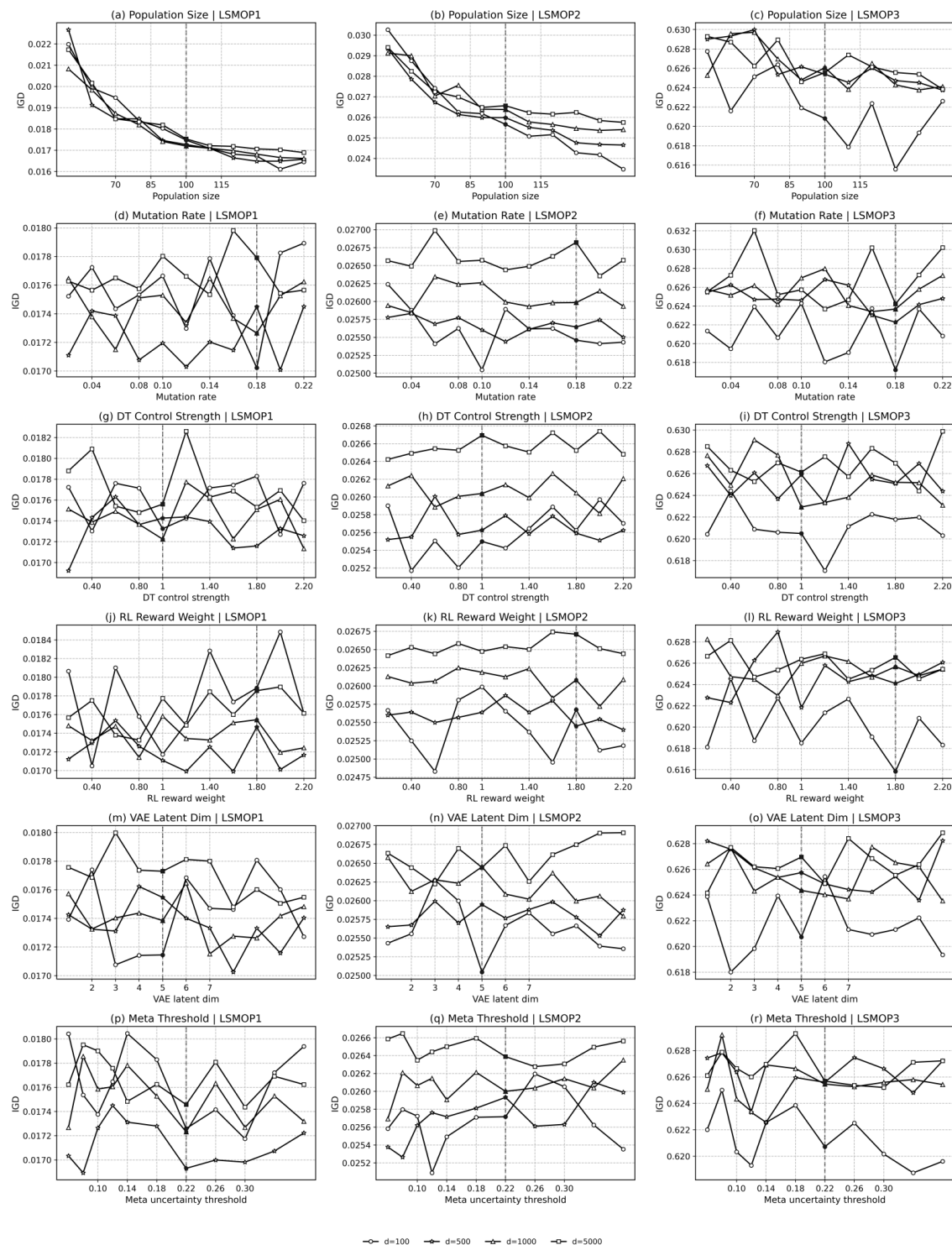


Figure 8. Single-Factor Sensitivity Analysis on LSMOP1–LSMOP3 across Four Dimensions.

To examine interactions between parameters, a two-parameter sensitivity analysis was performed for DT control strength and the Meta reconstruction-error threshold (Figure 9). Across the tested parameter combinations, the CVs of IGD are 4.013%, 4.345%, and 1.733% for LSMOP1, LSMOP2, and LSMOP3, respectively, all below 5%. None of the plots exhibits consistent deterioration as either parameter changes. DT-MLCEA therefore maintains relatively stable performance across the tested combinations of DT control strength and the Meta threshold, indicating limited dependence on a specific pair of parameter values.

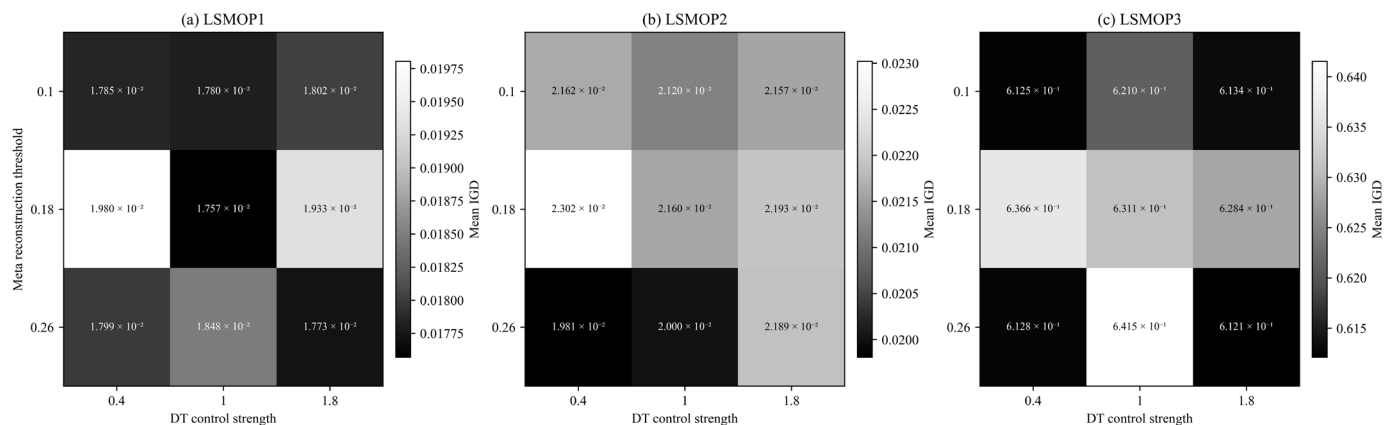


Figure 9. Two-parameter sensitivity analysis of mean IGD with respect to DT control strength and Meta reconstruction-error threshold on LSMOP1–LSMOP3.

5. Conclusions

This study proposes DT-MLCEA, a lightweight digital twin-based cooperative framework for large-scale multi-objective optimization. Unlike earlier designs based on computationally intensive models, DT-MLCEA combines an EA search process with lightweight DT control. Mini-RL, Mini-AE, and Mini-Meta provide low-overhead collaborative regulation, enabling adaptive adjustment of crossover rate, mutation rate, and search intensity through state monitoring, mode switching, and temporal smoothing. This mechanism balances convergence and distribution in complex high-dimensional search spaces. Experiments on LSMOP1–LSMOP9 with 100, 500, 1000, and 5000 decision variables show that DT-MLCEA significantly outperforms LMEA, QLMGO, AMSLMOE, and MOEA/D-RDG in terms of HV and IGD on 28 and 24 problem-dimension combinations, respectively. The results also indicate stable HV, limited IGD degradation as dimensionality increases, and robustness to the tested noise conditions.

Future work will investigate how problem-specific heuristics and domain knowledge can be used to customize digital twin fidelity and guidance vectors for real-world LSMOP applications, such as feature selection. Multi-agent reinforcement learning may also be incorporated to coordinate guidance vectors, mutation and crossover operators, and key parameters, while interpretable preferences and online noise models may further improve applicability and efficiency in complex environments.

Author Contributions: Conceptualization, D.L. and W.B.; methodology, D.L., F.G. and W.B.; software, C.C., H.W., H.L. and Y.S.; validation, D.L., F.G., C.C., H.W., H.L., Y.S., X.L. and M.D.; formal analysis, D.L., F.G. and X.L.; investigation, D.L., F.G., C.C., H.W., H.L., Y.S., X.L. and M.D.; resources, D.L., F.G., C.C. and H.W.; data curation, D.L., C.C., H.W., H.L., Y.S., X.L. and M.D.; writing—original draft preparation, D.L., F.G., C.C., H.W., H.L. and Y.S.; writing—review and editing, D.L., F.G., C.C., H.W., H.L., Y.S., X.L., M.D. and W.B.; visualization, D.L., C.C., H.W., H.L. and Y.S.; supervision, D.L., F.G. and W.B.; project administration, D.L. and W.B.; funding acquisition, W.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Lhasa Regional Collaborative Innovation Special Project (“Unveiling the List and Appointing Talents” Program), grant number LSKJ202463.

Data Availability Statement: The data presented in this study are openly available at <https://github.com/621-time/DT-MLCEA> (accessed on 6 August 2026).

Acknowledgments: The authors thank the Lhasa Regional Collaborative Innovation Special Project for its support.

Conflicts of Interest: Author Haiyang Lyu and Yutao Song are employed by Nanjing Hanhai Fuxi Defense Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A

Appendix A.1

This appendix lists the principal hyperparameter values used for each DT-MLCEA module in the reported experiments. These parameters affect search behavior, convergence, and final solution quality and are therefore provided to facilitate reproducibility and interpretation.

Table A1. Principal hyperparameters used in the experiments.

Hyperparameters	Default_Value	Description
Population size	100	Number of individuals in each generation
Maximum function evaluations	10,000	Termination budget used in the experiments
Metric update interval	5	Frequency of performance metric updates
Selection method	nsga2	Environmental selection framework
Crossover probability	0.9	Controls the recombination strength
Mutation probability	0.18	Controls the perturbation strength and diversity maintenance
SBX distribution index	20	Distribution index for simulated binary crossover
Polynomial mutation index	20	Distribution index for polynomial mutation
Elitism	TRUE	Preserves high-quality solutions during environmental selection
DT update interval	12	Low-frequency refresh interval of the digital twin layer in the experiment setting
Forced refresh stagnation threshold	8	Triggers an early DT refresh when stagnation is detected
DT control strength	1	Controls the adjustment magnitude of DT-guided parameter updates
DT smoothing factor	0.65	Smooths parameter updates to reduce oscillation
Confidence floor	0.25	Lower bound of the DT confidence output
State history size	32	Length of the state history buffer
Stagnation threshold	8	Window length used for stagnation detection
Mini-RL enabled	TRUE	Enables the lightweight reinforcement learning controller
RL update interval	3	Update frequency of the Q-table controller
Learning rate	0.08	Learning rate used in Q-learning
Discount factor	0.9	Discount factor for future rewards
Exploration rate	0.12	Probability of random action exploration
Convergence reward weight	1.8	Weight assigned to convergence improvement in the reward
Confidence reward weight	0.2	Weight assigned to stability confidence in the reward
Stagnation penalty	0.05	Penalty applied when the search is stagnant
Mini-AE enabled	TRUE	Enables lightweight state encoding and reconstruction
Input feature dimension	6	Dimension of the input state feature vector
Latent dimension	5	Dimension of the latent representation
Random seed	42	Seed used for Mini-AE weight initialization
Mini-Meta enabled	TRUE	Enables the lightweight meta-gating module
Meta update interval	5	Frequency of meta-mode updates
Sliding window length	8	Window size used for recent-state statistics
High reconstruction error threshold	0.22	Threshold for uncertainty-aware mode switching
Low diversity threshold	0.06	Threshold used to trigger explore mode
Bottom-up interval	12	Frequency of state transmission from EA to DT

Appendix A.2

Figure A1 presents a representative control trajectory of DT-MLCEA on LSMOP3 with 1000 variables and 50,000 function evaluations. Mini-RL dynamically switches among four bounded actions, while Mini-Meta activates the exploration mode at several search stages. The resulting MACV variations remain bounded, illustrating the problem-dependent interaction between RL-based corrections and meta-policy gating.

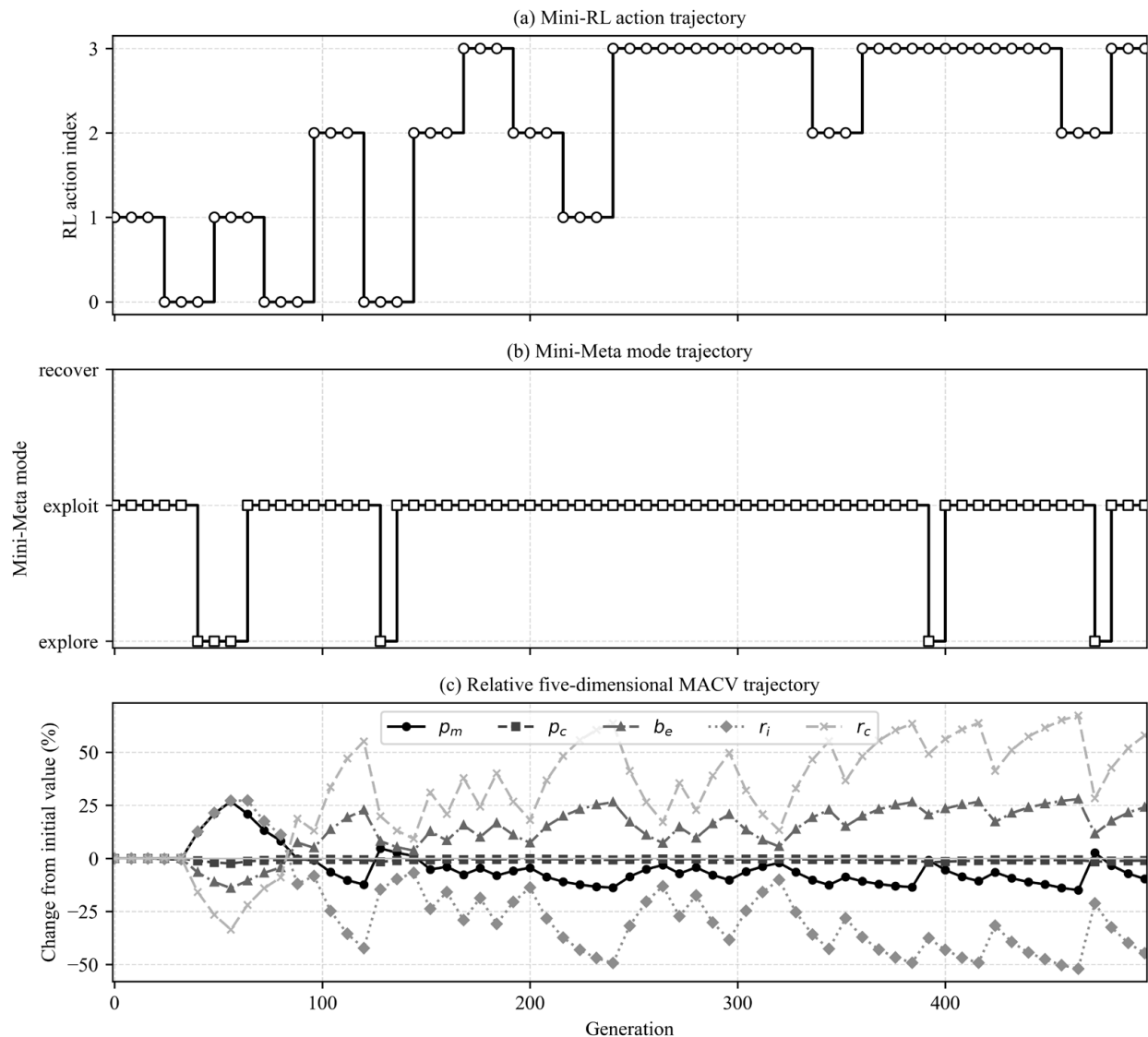


Figure A1. Trajectories of Mini-RL actions, Mini-Meta modes, and relative MACV changes on LSMOP3 under 50,000 function evaluations.

Appendix A.3

Following the Friedman omnibus test, two-sided paired Wilcoxon signed-rank tests were conducted between DT-MLCEA and each competing algorithm using the mean HV and IGD values from the 36 problem-dimension instances. Holm correction was applied separately to the four comparisons for each metric. As shown in Table A2, DT-MLCEA achieved significantly better overall performance than all four competitors for both HV and IGD at the 0.05 significance level. The rank-biserial effect sizes ranged from 0.802 to 1.000 for HV and from 0.375 to 1.000 for IGD, indicating large effects in most comparisons, although the advantage over QLMGO in IGD was comparatively modest.

Table A2. Holm-adjusted post hoc comparisons between DT-MLCEA and the competing algorithms across 36 problem-dimension instances.

Metric	Comparison	Wilcoxon W	Raw p -Value	Holm-Adjusted p -Value	Rank-Biserial Effect Size (r_{rb})	Significant at 0.05
HV	DT-MLCEA vs. LMEA	10	1.251×10^{-9}	3.754×10^{-9}	0.97	Yes
HV	DT-MLCEA vs. QLMGO	51	1.024×10^{-6}	2.049×10^{-6}	0.847	Yes
HV	DT-MLCEA vs. AMSLMOEa	0	2.910×10^{-11}	1.164×10^{-10}	1	Yes
HV	DT-MLCEA vs. MOEA/D-RDG	66	5.422×10^{-6}	5.422×10^{-6}	0.802	Yes
IGD	DT-MLCEA vs. LMEA	0	2.910×10^{-11}	1.164×10^{-10}	1	Yes
IGD	DT-MLCEA vs. QLMGO	208	4.964×10^{-2}	4.964×10^{-2}	0.375	Yes
IGD	DT-MLCEA vs. AMSLMOEa	115	3.595×10^{-4}	7.190×10^{-4}	0.655	Yes
IGD	DT-MLCEA vs. MOEA/D-RDG	0	2.910×10^{-11}	1.164×10^{-10}	1	Yes

References

1. Rajeswari, P.; Theodore Chandra, S.; Smitha, S. Efficient k-way partitioning of very-large-scale integration circuits with evolutionary computation algorithms. *Bull. Electr. Eng. Inform.* **2024**, *13*, 4002–4007. [\[CrossRef\]](#)
2. Yu, C.; Wang, Y.; Tang, C.; Feng, W.; Lv, J. EU-Net: Automatic U-Net neural architecture search with differential evolutionary algorithm for medical image segmentation. *Comput. Biol. Med.* **2023**, *167*, 107579. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Albogamy, F.R.; Paracha, M.Y.I.; Hafeez, G.; Khan, I.; Murawwat, S.; Rukh, G.; Khan, S.; Khan, M.U.A. Real-time scheduling for optimal energy optimization in smart grid integrated with renewable energy sources. *IEEE Access* **2022**, *10*, 35498–35520. [\[CrossRef\]](#)
4. Tian, Y.; Si, L.; Zhang, X.; Cheng, R.; He, C.; Tan, K.C.; Jin, Y. Evolutionary large-scale multi-objective optimization: A survey. *ACM Comput. Surv.* **2021**, *54*, 1–34. [\[CrossRef\]](#)
5. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [\[CrossRef\]](#)
6. Zitzler, E.; Laumanns, M.; Thiele, L. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*; Computer Engineering and Networks Laboratory, ETH Zurich: Zurich, Switzerland, 2001.
7. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [\[CrossRef\]](#)
8. Tian, Y.; Lu, C.; Zhang, X.; Tan, K.C.; Jin, Y. Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks. *IEEE Trans. Cybern.* **2021**, *51*, 3115–3128. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Ma, X.; Liu, F.; Qi, Y.; Wang, X.; Li, L.; Jiao, L.; Yin, M.; Gong, M. A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables. *IEEE Trans. Evol. Comput.* **2016**, *20*, 275–298. [\[CrossRef\]](#)
10. Durillo, J.J.; Nebro, A.J.; Coello, C.A.C.; García-Nieto, J.; Luna, F.; Alba, E. A study of multiobjective metaheuristics when solving parameter scalable problems. *IEEE Trans. Evol. Comput.* **2010**, *14*, 618–635. [\[CrossRef\]](#)
11. Antonio, L.M.; Coello, C.A.C. Use of cooperative coevolution for solving large scale multiobjective optimization problems. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*; IEEE: Piscataway, NJ, USA, 2013; pp. 2758–2765.
12. Hong, W.J.; Yang, P.; Tang, K. Evolutionary computation for large-scale multi-objective optimization: A decade of progresses. *Int. J. Autom. Comput.* **2021**, *18*, 155–169. [\[CrossRef\]](#)
13. Zille, H. Large-Scale Multi-Objective Optimisation: New Approaches and a Classification of the State-of-the-Art. Ph.D. Thesis, Otto von Guericke University Magdeburg, Magdeburg, Germany, 2019.
14. Kukkonen, S.; Lampinen, J. GDE3: The third evolution step of generalized differential evolution. In *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*; IEEE: Piscataway, NJ, USA, 2005; Volume 1, pp. 443–450.

15. Zhang, X.; Tian, Y.; Cheng, R.; Jin, Y. A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. *IEEE Trans. Evol. Comput.* **2018**, *22*, 97–112. [[CrossRef](#)]
16. Song, A.; Yang, Q.; Chen, W.N.; Zhang, J. A random-based dynamic grouping strategy for large scale multi-objective optimization. In *Proceedings of the 2016 IEEE Congress on Evolutionary Computation*; IEEE: Piscataway, NJ, USA, 2016; pp. 468–475.
17. Liu, R.; Li, Y.; Liu, J. A random dynamic grouping based weight optimization framework for large-scale multi-objective optimization problems. *Swarm Evol. Comput.* **2020**, *55*, 100684. [[CrossRef](#)]
18. Zille, H.; Ishibuchi, H.; Mostaghim, S.; Nojima, Y. A framework for large-scale multiobjective optimization based on problem transformation. *IEEE Trans. Evol. Comput.* **2018**, *22*, 260–275. [[CrossRef](#)]
19. Sander, F.; Zille, H.; Mostaghim, S. Transfer strategies from single- to multi-objective grouping mechanisms. In *Proceedings of the Genetic and Evolutionary Computation Conference*; ACM: New York, NY, USA, 2018; pp. 729–736.
20. Li, M.; Wei, J. A cooperative co-evolutionary algorithm for large-scale multi-objective optimization problems. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*; ACM: New York, NY, USA, 2018; pp. 1716–1721.
21. Zhang, J.; Wei, L.; Fan, R.; Sun, H.; Hu, Z. Solve large-scale many-objective optimization problems based on dual analysis of objective space and decision space. *Swarm Evol. Comput.* **2022**, *70*, 101045. [[CrossRef](#)]
22. Yang, L.; Liu, J.; Li, F.; Tan, S.; Zheng, T.; Liu, Y. A matrix adaptation evolution strategy based evolution algorithm for large-scale many-objective optimization. In *Proceedings of the 2020 Chinese Control and Decision Conference*; IEEE: Piscataway, NJ, USA, 2020; pp. 2518–2523.
23. Zhang, M.; Li, W.; Zhang, L.; Jin, H.; Mu, Y.; Wang, L. A Pearson correlation-based adaptive variable grouping method for large-scale multi-objective optimization. *Inf. Sci.* **2023**, *639*, 118737. [[CrossRef](#)]
24. He, C.; Li, L.; Tian, Y.; Zhang, X.; Cheng, R.; Jin, Y.; Yao, X. Accelerating large-scale multiobjective optimization via problem reformulation. *IEEE Trans. Evol. Comput.* **2019**, *23*, 949–961. [[CrossRef](#)]
25. He, C.; Cheng, R.; Tian, Y.; Zhang, X. Iterated problem reformulation for evolutionary large-scale multiobjective optimization. In *Proceedings of the 2020 IEEE Congress on Evolutionary Computation*; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
26. Li, H.; Zhang, Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Trans. Evol. Comput.* **2009**, *13*, 284–302. [[CrossRef](#)]
27. Qin, S.; Sun, C.; Jin, Y.; Tan, Y.; Fieldsend, J. Large-scale evolutionary multiobjective optimization assisted by directed sampling. *IEEE Trans. Evol. Comput.* **2021**, *25*, 724–738. [[CrossRef](#)]
28. Qian, H.; Yu, Y. Solving high-dimensional multi-objective optimization problems with low effective dimensions. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*; AAAI Press: Washington, DC, USA, 2017; Volume 31.
29. Liu, R.; Ren, R.; Liu, J. A clustering and dimensionality reduction based evolutionary algorithm for large-scale multi-objective problems. *Appl. Soft Comput.* **2020**, *89*, 106120. [[CrossRef](#)]
30. Wang, X.; Zhang, K.; Wang, J.; Jin, Y. An enhanced competitive swarm optimizer with strongly convex sparse operator for large-scale multiobjective optimization. *IEEE Trans. Evol. Comput.* **2022**, *26*, 859–871. [[CrossRef](#)]
31. Zhang, X.; Zheng, X.; Cheng, R.; Qiu, J.; Jin, Y. A competitive mechanism based multi-objective particle swarm optimizer with fast convergence. *Inf. Sci.* **2018**, *427*, 63–76. [[CrossRef](#)]
32. Tian, Y.; Zheng, X.; Zhang, X.; Jin, Y. Efficient large-scale multiobjective optimization based on a competitive swarm optimizer. *IEEE Trans. Cybern.* **2020**, *50*, 3696–3708. [[CrossRef](#)] [[PubMed](#)]
33. Qi, S.; Wang, R.; Zhang, T.; Huang, W.; Qing, F.; Hu, T.; Wang, L. A Thompson sampling-based sparse evolutionary operator for sparse large-scale multiobjective optimization. *IEEE Trans. Evol. Comput.* **2025**, *29*, 2521–2536. [[CrossRef](#)]
34. Mittal, S.; Saxena, D.K.; Deb, K.; Goodman, E.D. A learning-based innovized progress operator for faster convergence in evolutionary multi-objective optimization. *ACM Trans. Evol. Learn. Optim.* **2021**, *2*, 1–29. [[CrossRef](#)]
35. Liu, S.; Li, J.; Lin, Q.; Tian, Y.; Tan, K.C. Learning to accelerate evolutionary search for large-scale multiobjective optimization. *IEEE Trans. Evol. Comput.* **2023**, *27*, 67–81. [[CrossRef](#)]
36. Chugh, T.; Jin, Y.; Miettinen, K.; Hakanen, J.; Sindhya, K. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Trans. Evol. Comput.* **2018**, *22*, 129–142. [[CrossRef](#)]
37. Tao, F.; Zhang, H.; Liu, A.; Nee, A.Y. Digital twin in industry: State-of-the-art. *IEEE Trans. Ind. Inform.* **2019**, *15*, 2405–2415. [[CrossRef](#)]
38. Song, Y.; Wang, K.; Zhan, Z.-H.; Lei, Z.; Gao, S. Learning-assisted search path reconstruction empowers evolution algorithm for optimization. *IEEE Trans. Evol. Comput.* **2026**, *30*, 321–332. [[CrossRef](#)]
39. Yu, J.; Wang, K.; Lei, Z.; Cheng, J.; Gao, S. Serial multilevel-learned differential evolution with adaptive guidance of exploration and exploitation. *Expert Syst. Appl.* **2024**, *255*, 124646. [[CrossRef](#)]
40. Fang, W.; Zhang, H.; Qian, W.; Guo, Y.; Li, S.; Liu, Z.; Liu, C.; Hong, D. An adaptive job shop scheduling mechanism for disturbances by running reinforcement learning in digital twin environment. *J. Comput. Inf. Sci. Eng.* **2023**, *23*, 051013. [[CrossRef](#)]
41. Ouahabi, N.; Chebak, A.; Kamach, O.; Laayati, O.; Zegrari, M. Leveraging digital twin into dynamic production scheduling: A review. *Robot. Comput.-Integr. Manuf.* **2024**, *89*, 102778. [[CrossRef](#)]

42. Pan, L.; Lin, J.; Wang, H.; He, C.; Tan, K.C.; Jin, Y. Computationally expensive high-dimensional multiobjective optimization via surrogate-assisted reformulation and decomposition. *IEEE Trans. Evol. Comput.* **2025**, *29*, 921–935. [[CrossRef](#)]
43. Gu, H.; Wang, H.; He, C.; Yuan, B.; Jin, Y. Large-scale multiobjective evolutionary algorithm guided by low-dimensional surrogates of scalarization functions. *Evol. Comput.* **2025**, *33*, 309–334. [[CrossRef](#)] [[PubMed](#)]
44. Sarangi, P.; Mohapatra, S.; Mohapatra, P. A novel Q-learning-inspired Mountain Gazelle Optimizer for solving global optimization problems. *Int. J. Mach. Learn. Cybern.* **2025**, *16*, 6167–6213. [[CrossRef](#)]
45. Zhang, X.; Chen, D.; Ge, F.; Zou, F.; Cui, L. Large-scale multiobjective competitive swarm optimizer algorithm based on regional multidirectional search. *Complex Intell. Syst.* **2025**, *11*, 51. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.