Dynamic Constrained Multi-objective Evolutionary Algorithm Based on Graph-Temporal Neural Network

1st Chenwen Ding
Shantou University
Guangdong, China
23cwding@stu.edu.cn

2nd Jiaping Hu
Shantou University
Guangdong, China
23jphu@stu.edu.cn

3rd Wenji Li*
Shantou University
Guangdong, China
liwj@stu.edu.cn

4th Hanyuan Zhang
Shantou University
Guangdong, China
20hyzhang@stu.edu.cn

5th Jiachun Huang
Shantou University
Guangdong, China
22jchuang1@stu.edu.cn

6th Zhaojun Wang*
Shantou University
Guangdong, China
17zjwang@stu.edu.cn

7th Biao Xu
Shantou University
Guangdong, China
xubiao@stu.edu.cn

8th Yun Li
University of Electronic Science and Technology of China
Sichuan, China
Yun.li@ieee.org

9th Zhun Fan*
University of Electronic Science and Technology of China
Sichuan, China
fanzhun@uestc.edu.cn

***Abstract*—Dynamic constrained multi-objective optimization problems (DCMOPs) are common in engineering applications. In these problems, objectives and constraints change over time, requiring algorithms to adapt quickly and continuously track the dynamic constrained Pareto front. However, existing DCMOEAs mainly focus on predicting or perturbing individual solutions or the centroid of the Pareto set, failing to exploit the overall spatial structure of the population. Moreover, they insufficiently utilize historical information, making it difficult to capture long-term change patterns of the dynamic environment. To address these issues, this paper proposes a dynamic constrained multi-objective evolutionary algorithm based on graph-temporal neural networks. Specifically, the algorithm constructs a population topology by partitioning subspaces and uses graph convolutional networks (GCNs) to extract topological features. Subsequently, it employs gated recurrent units (GRUs) to learn the migration trends of the Pareto set across historical environments, enabling accurate prediction of its distribution in new environments. Additionally, the algorithm integrates memory-based and diversity-based strategies to generate initial populations that balance convergence, feasibility, and diversity. Experimental results on the DCP test suite show that the proposed algorithm outperforms five representative DCMOEAs in most test scenarios, validating its effectiveness in solving DCMOPs.**

***Index Terms*—Evolutionary computation, Dynamic constrained multi-objective optimization, Graph convolutional networks.**

I. Introduction

Dynamic constrained multi-objective optimization problems (DCMOPs) have attracted increasing attention in recent years [1]. These problems are characterized by multiple objectives and constraints that change over time during the optimization process [2]. Such problems are prevalent in real-world applications, such as cognitive radio networks [3] and resource scheduling [4].

Generally, a DCMOP can be defined as follows:

$$\min \mathbf{F}(\mathbf{x},t) = (f_1(\mathbf{x},t),\ldots,f_M(\mathbf{x},t))^T \quad \text{s.t.} \begin{cases} \mathbf{x} = (x_1,x_2,\ldots,x_D)^T \in \mathbb{R}^D \\ g_i(\mathbf{x},t) \le 0, \quad i = 1,\ldots,p \\ h_j(\mathbf{x},t) = 0, \quad j = 1,\ldots,q \end{cases} \tag{1}$$

where $\mathbf{x}$ denotes the D-dimensional decision variable in the decision space $\mathbb{R}^D$, and $\mathbf{F}(\mathbf{x},t)$ represents the objective functions. $g_i(\mathbf{x},t)$ and $h_j(\mathbf{x},t)$ denote the i-th inequality and j-th equality constraints, respectively. A solution satisfying all the constraints is considered feasible. The discrete time step t is defined as follows:

$$t = \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \frac{1}{n_t} \tag{2}$$

where τ is the generation counter, and τ_t and n_t denote the frequency and severity of environmental changes, respectively.

In environment t, a feasible solution $\mathbf{x}_1$ Pareto dominates another feasible solution $\mathbf{x}_2$, denoted as $\mathbf{x}_1 \prec_t \mathbf{x}_2$, if and only if both of the following conditions are satisfied:

$$\begin{cases} f_m(\mathbf{x}_1,t) \le f_m(\mathbf{x}_2,t), & \forall m = 1,\ldots,M, \\ f_m(\mathbf{x}_1,t) < f_m(\mathbf{x}_2,t), & \exists m = 1,\ldots,M. \end{cases} \tag{3}$$

* Corresponding authors.

A feasible solution $\mathbf{x}^*$ is termed a dynamic constrained Pareto-optimal solution if it is not dominated by any other feasible solution in the current environment. The set of all such solutions constitutes the dynamic constrained Pareto set (DCPS), defined as:

$$\text{DCPS} = \{\mathbf{x}^* \in \Omega_t \mid \nexists\, \mathbf{x} \in \Omega_t,\, \mathbf{x} \prec_t \mathbf{x}^*\} \tag{4}$$

where Ω_t denotes the feasible region at time t. Correspondingly, the set of objective vectors of the DCPS forms the dynamic constrained Pareto front (DCPF), defined as:

$$\text{DCPF} = \{\mathbf{F}(\mathbf{x}^*, t) \mid \mathbf{x}^* \in DCPS\} \tag{5}$$

Solving DCMOPs requires strategies capable of tracking the DCPS as environments shift while maintaining solution feasibility under current constraints. Although DMOPs and CMOPs have been examined extensively, research on DCMOPs remains limited and presents significant challenges [5].

DCMOPs lie at the intersection of DMOPs and CMOPs. Consequently, designing dynamic constrained multi-objective evolutionary algorithms (DCMOEAs) often integrates environmental change response mechanisms from DMOEAs with constraint-handling techniques (CHTs) from CMOEAs. Common response mechanisms include diversity-based methods that introduce random mutations [6], memory-based approaches that reuse historical high-quality solutions [7], and prediction-based models that learn the DCPS trajectory to forecast distributions in new environments [8]. Regarding CHTs, critical approaches include penalty methods for balancing feasibility and convergence [9], decoupling methods for assessing constraints separately [10], and multi-objective reformulation methods that treat constraints as additional objectives [11]. Furthermore, problem transformation methods recast CMOPs into simpler forms, exemplified by CCMO [12] for coevolution and PPS [13] for two-stage search.

Based on this, recent studies have proposed various DCMOEAs [14]. Azzouz et al. combined an adaptive penalty with a diversity-based environmental change response mechanism to propose DC-NSGAII [15]. Specifically, its variant DC-NSGAII-A introduces random solutions after changes, while DC-NSGAII-B perturbs existing ones. Subsequently, Chen et al. proposed dCMOEA, which employs adaptive penalties for constraint handling [16]. They also designed a hybrid diversity–memory response mechanism that repairs archived high-quality solutions using new information. Consequently, the new initial population comprises half repaired solutions and half random solutions. This mechanism was further coupled with a reference-point based NSGA-III to form DC-NSGA-III for comparative analysis. More recently, Chen et al. developed an enhanced two-stage hybrid diversity–memory mechanism [17]. The first stage introduces random solutions to boost diversity, while the second stage perturbs solutions based on the centroids of historical non-dominated sets. By embedding this mechanism into the CCMO framework, they proposed TDCEA. However, such mechanisms primarily rely on mixing historical solutions with randomness, which may be insufficient for tracking rapidly evolving environments. To address this limitation, Zhang et al. proposed HATC, which combines a prediction-based response mechanism with multi-population coevolutionary constraint handling [17]. This method analyzes historical non-dominated solutions to predict promising regions in the new environment and initializes the population accordingly.

However, current DCMOEAs still face certain limitations. First, most environmental change response mechanisms focus on predicting or perturbing individual solutions or the Pareto centroid to adapt to the new environment, often overlooking the population's overall spatial structure and the topological relationships among individuals. Second, when initializing the population for the next environment, most mechanisms utilize only one or two past generations, thereby failing to capture long-term environmental change trends. Consequently, these limitations hinder the ability of DCMOEAs to learn environmental dynamics and accurately generate high-quality initial populations.

In recent years, neural network-based learning methods have gained significant attention in evolutionary optimization; however, their application in DCMOEAs remains limited. In particular, graph convolutional networks (GCNs) and gated recurrent units (GRUs) offer the potential to address the aforementioned limitations. In DCMOPs, dynamic objectives and constraints shape population distributions that exhibit a natural topology in the decision space [18]. Correlations exist among individuals, and an adjacency matrix can be employed to quantify these relationships [19]. GCNs aggregate features from nodes and their neighbors via the adjacency matrix [20], effectively capturing spatial topology and local information. This provides a promising modeling tool for environmental change response mechanisms to predict high-quality populations in new environments. Moreover, dynamic environments in DCMOPs exhibit temporal correlations, thereby supporting prediction-based response mechanisms. Unlike linear predictors, GRUs utilize gating mechanisms to retain and update historical information, allowing them to learn long-term dependencies and more accurately forecast the evolution of the Pareto set. Although prior work has demonstrated the utility of GRUs in DMOPs [21], existing methods typically predict only isolated non-dominated solutions, ignoring the overall population topology, and fail to handle DCMOPs with dynamic constraints. Consequently, effectively utilizing population structure and historical change patterns to enhance dynamic responses remains a key open challenge for DCMOEAs.

Motivated by these considerations, we develop a DCMOEA that integrates GCNs and GRUs. The main contributions are summarized as follows:

1) We propose a graph-temporal neural network framework to predict population distributions in new environments. This model employs GCNs to capture the spatial topology of Pareto solutions in the decision space and utilizes GRUs to learn temporal change patterns from multiple historical environments, thereby enabling accurate

forecasting of the Pareto set's evolution in the next environment.

2) Based on the graph-temporal network, we design a prediction-based environmental change response mechanism. This mechanism is augmented with memory- and diversity-based strategies to generate high-quality initial populations, thereby enhancing performance and robustness across various dynamic constrained scenarios.
3) Extensive experiments on the DCP benchmark suite demonstrate that the proposed algorithm significantly outperforms five representative state-of-the-art algorithms in solving DCMOPs.

II. Proposed Method

This section outlines the proposed Method and the core environmental change response mechanism based on a graph temporal neural network.

A. Method Architecture

Fig.1 shows the overall framework of our dynamic constrained multi-objective optimization algorithm. Similar to typical DCMOEAs, it cycles through environmental change detection, environmental change response, and static optimization. The detection step monitors shifts in objectives and constraints; once a change is found, the response mechanism seeds a population suited to the new environment; during the static phase, the constrained MOEA drives the population toward the current constrained Pareto front. The main novelty lies in the response mechanism, centered on a GCN–GRU graph temporal predictor. It captures the topology of Pareto solutions, learns temporal change patterns from past environments, and predicts high-quality initial populations for the next environment. Two auxiliary response strategies are combined to balance convergence, feasibility, and diversity. Notably, the prediction-based response is modular: it can plug into diverse constrained MOEAs to form DCMOEAs for different dynamic constrained settings.

The algorithm proceeds as outlined in pseudocode Algorithm 1. It first randomizes the N individuals in P and the graph temporal network parameters (line 2). Because training samples are scarce in dynamic optimization, it partitions the objective space with uniform weight vectors $\Lambda = \{\lambda_1, \ldots, \lambda_M\}$ into M subspaces and builds an adjacency matrix A over subspace nodes (line 3), reducing topology from individuals to subspaces. In each generation, the algorithm checks whether the environment has changed (line 5). If a change is detected, the response mechanism triggers: the graph temporal network forecasts the shift of the Pareto population, and memory-based plus diversity-based auxiliaries jointly build an initial population suited to the new environment (line 7), detailed in Section II-B. It then enters static optimization, using a suitable CMOEA to produce the next generation as needed (line 10). After each evolution, it extracts nondominated solutions of P, computes and stores subspace centers $C^t = \{C_1^t, \ldots, C_M^t\}$ over M subspaces (line 11); these centers feed the graph temporal network for training and prediction. This loop repeats until termination, outputting the constrained Pareto set in the final environment.

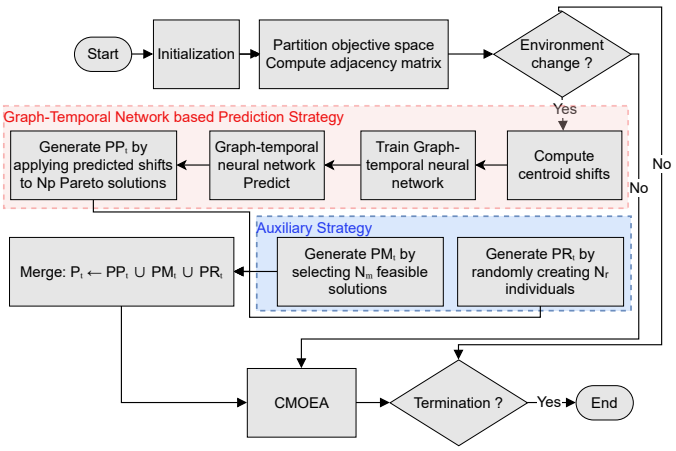

Fig. 1. Framework of the proposed algorithm.

Algorithm 1: Graph-Temporal Neural Network Based DCMOEA Framework

Input: the DCMOP, population size N, number of subspaces M, prediction subpopulation size N_p, memory subpopulation size N_m, random subpopulation size N_r
Output: the $DCPOS_t$ at the final environment step t

1 set the environment step $t \leftarrow 0$;
2 initialize population P_0 with size N and graph-temporal neural network;
3 partition objective space into M subspaces;
4 compute adjacency matrix A;
5 **while** *termination condition is not met* **do**
  // Static optimization phase
6 $P_t \leftarrow \text{CMOEA}(P_t)$;
7 compute and store the centroids of non-dominated solutions C_t;
8 environment detection $\leftarrow \text{ChangeDetection}(P_t)$;
9 **if** *environmental change is detected* **then**
10 $t \leftarrow t + 1$;
11 EnvironmentResponseStrategy ; // Algorithm 2
12 **end**
13 **end**
14 **return** $DCPOS_t$;

B. Environmental change response mechanism based on a graph temporal neural network

The core of the proposed environmental change response mechanism is a graph temporal neural network that cascades a GCN and a GRU. To realize this mechanism, the algorithm first builds the population topology. Specifically, it partitions the objective space into M subspaces using uniformly distributed weight vectors. Each subspace is one node of the population topology. Pareto-optimal solutions are assigned to

the nearest subspace by distance to its weight vector. Edges are built according to subspace affinity. The edge weight between subspace nodes i and j, A_{ij}, depends on the normalized Euclidean distance between w_i and w_j:

$$A_{ij} = 1 - \frac{\|w_i - w_j\|_2}{\max_{i,j}\|w_i - w_j\|_2} \tag{6}$$

Thus, closer subspaces have larger edge weights, indicating stronger affinity. In the population topology, each node feature records the offsets of that subspace's Pareto cluster centers across three consecutive past environments. Concretely, it uses $[\Delta C_m^{t-2}, \Delta C_m^{t-1}, \Delta C_m^t]$ to capture the movement direction and magnitude between adjacent environments, which is computed as:

$$\Delta C_m^t = C_m^t - C_m^{t-1} \tag{7}$$

Here, C_m^t is the cluster center of subspace m in environment t. Compared with raw centers, these offsets show stronger continuity across dynamic environments. Their variation amplitude is also smaller, which helps reduce learning difficulty and improve prediction accuracy.

Based on the above topology, the graph temporal network uses two inputs: the three-environments offset series of subspace cluster centers $[\Delta C_{t-2}, \Delta C_{t-1}, \Delta C_t]$, and the adjacency matrix A. It then predicts the next-environment offset ΔC_{t+1} for the Pareto cluster centers. In the network, the GCN layer extracts subspace topology features, with a forward pass:

$$H^{(l+1)} = \sigma(\hat{A}\, H^{(l)} W^{(l)}) \tag{8}$$

Here, $\hat{A}$ is the normalized adjacency, $H^{(l)}$ is the node feature matrix at layer l, the input layer $H^{(0)}$ is the offset matrix of cluster centers. $W^{(l)}$ is a learnable weights, and σ the activation. The GCN uses the adjacency to extract spatial features from each historical offset and captures inter-subspace topology. Then the GRU layer takes the topology feature sequence from each environment. It uses gating to learn temporal migration patterns of the subspaces, with a forward pass:

$$S_t = (1 - u_t) \odot S_{t-1} + u_t \odot \tilde{S}_t \tag{9}$$

Here S_t is the hidden state at environment t, u_t is the update gate, $\tilde{S}_t$ is the candidate state, and $\odot$ denotes element-wise multiplication.

For network training, we adopt an incremental scheme to cope with the few samples available in dynamic optimization. At each environmental change, $[\Delta C_{t-3}, \Delta C_{t-2}, \Delta C_{t-1}]$ serve as train inputs and ΔC_t as the train label; Then we apply several gradient-descent steps to adjust the network weights, so that the network can keeps prior knowledge while adapting the newest pattern. After training, the network takes $[\Delta C_{t-2}, \Delta C_{t-1}, \Delta C_t]$ as input and predicts the next-environment offset ΔC_{t+1}.

Once ΔC_{t+1} is obtained, the algorithm shifts the Pareto solutions accordingly. For the i-th Pareto solution in subspace m, the update is:

$$x_m^i \leftarrow x_m^i + \Delta C_m^{t+1} \tag{10}$$

To keep the population size, the algorithm updates N_p Pareto solutions and forms the predicted subpopulation PP_t. Meanwhile, two auxiliary strategies balance feasibility and diversity: keep N_m feasible solutions in the new environment as the memory subpopulation PM_t to secure feasibility; randomly generate N_r solutions as the random subpopulation PR_t to widen the search and avoid local optima. Finally, these three subpopulations are merged into the initial population P_t for the new environment. Notably, during the first four environmental changes, historical data are insufficient to train the network, so a simplified change response mechanism is used: only the memory and random subpopulations (each $0.5N$) are combined to ensure basic responsiveness early on. Algorithm 2 presents the detailed procedure of the graph temporal network–based environmental change response mechanism.

Algorithm 2: Graph-Temporal Neural Network-based Environment Response Strategy

Input: population P_t, adjacency matrix A, graph-temporal neural network network, historical centroids $C_{t-4} \dots C_t$, prediction subpopulation size N_p, memory subpopulation size N_m, random subpopulation size N_r, number of subspaces M

Output: updated population P_t, updated graph-temporal neural network

if $t \geq 5$ **then**
  compute centroid shifts $\Delta C_{t-3} \dots \Delta C_t$;
  network $\leftarrow$ Train(network, $[\Delta C_{t-3}, \Delta C_{t-2}, \Delta C_{t-1}], A; \Delta C_t$);
  $\Delta C_{t+1} \leftarrow$ Predict(network, $[\Delta C_{t-2}, \Delta C_{t-1}, \Delta C_t], A$);
  for $k = 1$ **to** M **do**
    for N_p *non-dominated individual* x_k^i *in subspace* k **do**
      $x_k^i \leftarrow x_k^i + \Delta C_k^{t+1}$;
      $PP_t \leftarrow PP_t \cup \{x_k^i\}$;
    end
  end
  select N_m feasible solutions from P_t to form memory subpopulation PM_t;
  randomly generate N_r individuals as random subpopulation PR_t;
  $P_t \leftarrow PP_t \cup PM_t \cup PR_t$;
end
else
  select $0.5N$ feasible solutions from P_t to form memory subpopulation PM_t;
  randomly generate $0.5N$ individuals as random subpopulation PR_t;
  $P_t \leftarrow PM_t \cup PR_t$;
end
return P_t, network;

III. Experimenta Study

A. Experimental Setup

To assess effectiveness, we compare the proposed algorithm with five advanced DCMOEAs: DC-NSGAII-A [15], DC-NSGAII-B [15], DC-NSGAIII [16], dCMOEA [16], and TDCEA [17]. For fair comparison, parameter settings for all baselines follow their original papers. In the static optimization phase, we embed CCMO in our framework and use differential evolution and polynomial mutation operators for population evolution.

In the graph-temporal network, both the GCN and GRU use single layers to keep training simple under scarce dynamic-optimization samples. Specifically, the GCN input size equals the decision dimension D, its hidden size is $1.5D$ with ReLU activation; the GRU hidden state is size D, matching the output. Training uses Adam with a learning rate of 0.002. Each incremental training step applies 30 gradient-descent updates with mean squared error loss. Meanwhile, A Dropout layer follows the GRU to reduce overfitting. Experiments use the DCP suite [17] (DCP1–DCP9) covering diverse dynamic patterns of constraints and objectives. The experiments use a decision dimension of $D = 10$ and a population size of $N = 100$. Environmental change frequency $\tau_t \in \{10, 20, 30\}$ and amplitude $n_t \in \{5, 10, 20\}$, span 81 dynamic constrained scenarios. Each algorithm runs 30 independent times per test, with 30 environmental changes per run. No environmental change occurs in the first 80 generations.

Algorithm performance is evaluated with mean inverted generational distance (MIGD) [22]. Smaller MIGD indicates better performance. Statistical significance is assessed with the Wilcoxon rank-sum test at a 0.05 level. In the result table, "+", "-", and "=" denote significantly better, significantly worse, or no significant difference relative to our method.

B. Experimental Results Analysis

Table I reports MIGD significance results for our method versus five comparators across nine tests. The statistics show our method wins in 81, 81, 81, 81, and 65 scenarios against the five baselines, indicating clear overall gains in convergence, feasibility, and diversity. These gains stem from the graph temporal network capturing both spatial topology and historical change patterns, enabling fast seeding of high-quality populations after an environmental shift. Further analysis shows that DC-NSGAII-A and DC-NSGAII-B perform similarly in most cases because they share the same constrained MOEA in the static phase. By contrast, although both our method and TDCEA use CCMO for static optimization, ours outperforms TDCEA in most scenarios. TDCEA perturbs around historical nondominated centroids, making limited use of full historical population information. Our approach learns whole-population trends in dynamic environments via the graph temporal network, allowing more accurate prediction of Pareto population shifts in new environments.

In some DCP4, DCP5, and DCP9 scenarios, our method does not reach the best populations. These cases have narrow, scattered feasible regions, posing higher demands on dynamic constraint handling. Our graph temporal model still has limits on such special problems. Especially when changes are mild, it cannot clearly outperform other response mechanisms.

To visualize the optimization effect more clearly, Fig.2 shows the final DCPOFs produced by six algorithms on DCP7 (τ_t=10, n_t=5). DCP7 involves time-varying objectives and constraints. Even with mild changes, methods like DC-NSGAII-A with simple responses fail to reach the true DCPOFs. TDCEA covers the DCPOFs overall but shows noticeable deviations in some environments. By contrast, our approach tracks the DCPOFs more accurately and balances convergence, feasibility, and diversity.

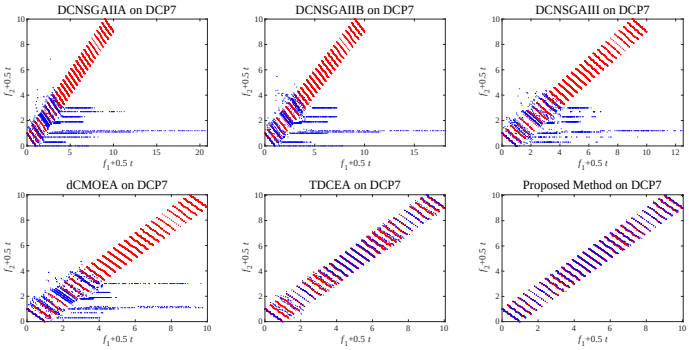

Fig. 2. DCPOFs of environments obtained by the comparative algorithms on DCP7 (n_t = 5, τ_t = 10). Red points are true DCPOFs, and blue ones are DCPOFs obtained by algorithms.

IV. Conclusion

This paper proposes a graph-temporal neural network-based DCMOEA to address the insufficient utilization of historical population information in existing approaches. The method first partitions the objective space into subspaces to construct a population topology. Subsequently, a GCN captures the topological relationships among subspaces, while a GRU learns the temporal evolutionary patterns of the Pareto set, enabling the accurate prediction of a high-quality population for the next environment. Additionally, memory-based and diversity-based response strategies are integrated to maintain population feasibility and diversity. Experimental results demonstrate that the proposed method outperforms several representative algorithms in overall performance. Future work will focus on: (1) designing more robust feasibility-maintenance strategies for dynamic constrained scenarios with complex feasible regions; (2) refining the model architecture and training scheme by exploring deeper networks and data augmentation to enhance prediction accuracy and generalization; and (3) applying the method to engineering problems to validate its practicality in solving real-world DCMOPs.

Acknowledgment

This research was supported in part by the National Science and Technology Major Project (grant number 2021ZD0111502), the National Natural Science Foundation

TABLE I
SIGNIFICANCE TEST OF MIGD ON ALL TEST INSTANCES

Problem	Proposed Method vs. DCNSGAII-A			Proposed Method vs. DCNSGAII-B			Proposed Method vs. DCNSGAIII			Proposed Method vs. dCMOEA			Proposed Method vs. TDCEA		
	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=
DCP1	0	9	0	0	9	0	0	9	0	0	9	0	1	8	0
DCP2	0	9	0	0	9	0	0	9	0	0	9	0	0	7	2
DCP3	0	9	0	0	9	0	0	9	0	0	9	0	0	7	2
DCP4	0	9	0	0	9	0	0	9	0	0	9	0	4	2	3
DCP5	0	9	0	0	9	0	0	9	0	0	9	0	2	7	0
DCP6	0	9	0	0	9	0	0	9	0	0	9	0	0	7	2
DCP7	0	9	0	0	9	0	0	9	0	0	9	0	0	9	0
DCP8	0	9	0	0	9	0	0	9	0	0	9	0	0	9	0
DCP9	0	9	0	0	9	0	0	9	0	0	9	0	1	7	1
All	0	81	0	0	81	0	0	81	0	0	81	0	8	63	10

of China (grant numbers 62176147, 62476163), the Science and Technology Planning Project of Guangdong Province of China (grant numbers 2022A1515110660, 2021JC06X549), the Science and Technology Special Funds Project of Guangdong Province of China (grant numbers STKJ2021176, STKJ2021019), the Guangdong Basic and Applied Basic Research Foundation(2023B1515120020, 2024A1515012450), and the STU Scientific Research Foundation for Talents (grant numbers NTF21001, NTF22030).

REFERENCES

[1] Z. Wang, K. Ye, M. Jiang, J. Yao, N. N. Xiong, and G. G. Yen, "Solving hybrid charging strategy electric vehicle based dynamic routing problem via evolutionary multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 68, p. 100975, 2022.

[2] I. Kucukoglu, R. Dewil, and D. Cattrysse, "The electric vehicle routing problem and its variations: A literature review," *Computers & Industrial Engineering*, vol. 161, p. 107650, 2021.

[3] C.-L. Chuang, W.-Y. Chiu, and Y.-C. Chuang, "Dynamic multiobjective approach for power and spectrum allocation in cognitive radio networks," *IEEE Systems Journal*, vol. 15, no. 4, pp. 5417–5428, 2021.

[4] A. Sundaram, "Multiobjective multi verse optimization algorithm to solve dynamic economic emission dispatch problem with transmission loss prediction by an artificial neural network," *Applied Soft Computing*, vol. 124, p. 109021, 2022.

[5] R. Azzouz, S. Bechikh, L. B. Said, and W. Trabelsi, "Handling time-varying constraints and objectives in dynamic evolutionary multi-objective optimization," *Swarm and evolutionary computation*, vol. 39, pp. 222–248, 2018.

[6] V. A. S. Hernández, O. Schütze, H. Wang, A. Deutz, and M. Emmerich, "The set-based hypervolume newton method for bi-objective optimization," *IEEE transactions on cybernetics*, vol. 50, no. 5, pp. 2186–2196, 2018.

[7] Y. Hu, J. Zheng, J. Zou, S. Yang, J. Ou, and R. Wang, "A dynamic multi-objective evolutionary algorithm based on intensity of environmental change," *Information Sciences*, vol. 523, pp. 49–62, 2020.

[8] L. Yan, W. Qi, A. K. Qin, S. Yang, D. Gong, B. Qu, and J. Liang, "Manifold clustering-based prediction for dynamic multiobjective optimization," *Swarm and Evolutionary Computation*, vol. 77, p. 101254, 2023.

[9] F. Vaz, Y. Lavinas, C. Aranha, and M. Ladeira, "Exploring constraint handling techniques in real-world problems on moea/d with limited budget of evaluations," in *Evolutionary Multi-Criterion Optimization: 11th International Conference, EMO 2021, Shenzhen, China, March 28–31, 2021, Proceedings 11*. Springer, 2021, pp. 555–566.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[11] Y. Zhou, M. Zhu, J. Wang, Z. Zhang, Y. Xiang, and J. Zhang, "Tri-goal evolution framework for constrained many-objective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 8, pp. 3086–3099, 2018.

[12] Y. Tian, T. Zhang, J. Xiao, X. Zhang, and Y. Jin, "A coevolutionary framework for constrained multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 102–116, 2020.

[13] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Push and pull search for solving constrained multi-objective optimization problems," *Swarm and evolutionary computation*, vol. 44, pp. 665–679, 2019.

[14] R. Liu, J. Li, C. Mu, L. Jiao *et al.*, "A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization," *European Journal of Operational Research*, vol. 261, no. 3, pp. 1028–1051, 2017.

[15] R. Azzouz, S. Bechikh, and L. Ben Said, "Multi-objective optimization with dynamic constraints and objectives: new challenges for evolutionary algorithms," in *Proceedings of the 2015 annual conference on genetic and evolutionary computation*, 2015, pp. 615–622.

[16] Q. Chen, J. Ding, S. Yang, and T. Chai, "A novel evolutionary algorithm for dynamic constrained multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 4, pp. 792–806, 2019.

[17] G. Chen, Y. Guo, Y. Wang, J. Liang, D. Gong, and S. Yang, "Evolutionary dynamic constrained multiobjective optimization: Test suite and algorithm," *IEEE Transactions on Evolutionary Computation*, 2023.

[18] L. Feng, W. Zhou, W. Liu, Y.-S. Ong, and K. C. Tan, "Solving dynamic multiobjective problem via autoencoding evolutionary search," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 2649–2662, 2020.

[19] M. M. Ali and W. Zhu, "A penalty function-based differential evolution algorithm for constrained global optimization," *Computational Optimization and Applications*, vol. 54, no. 3, pp. 707–739, 2013.

[20] K. Li, R. Chen, G. Fu, and X. Yao, "Two-archive evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 303–315, 2018.

[21] X. Hou, F. Ge, D. Chen, L. Shen, and F. Zou, "Temporal distribution-based prediction strategy for dynamic multi-objective optimization assisted by gru neural network," *Information Sciences*, vol. 649, p. 119627, 2023.

[22] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization," *IEEE transactions on cybernetics*, vol. 44, no. 1, pp. 40–53, 2013.