Adaptive Epsilon-Penalty Switching for Push-Pull Search from Vicinity to Feasibility

1st Jiaping Hu†
Shantou University
Guangdong, China
23jphu@stu.edu.cn

2nd Jiachun Huang†
Shantou University
Guangdong, China
22jchuang1@stu.edu.cn

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

4th Zhaojun Wang*
South China University of Technology
Jinpeng Electronic Information Machine
Guangdong, China
wangzj@gzjp.cn

5th Wulin Cai
Shantou University
Guangdong, China
24wlcai@stu.edu.cn

6th Chenwen Ding
Shantou University
Guangdong, China
23cwding@stu.edu.cn

7th Hanyuan Zhang
Shantou University
Guangdong, China
20hyzhang@stu.edu.cn

8th Biao Xu
Shantou University
Guangdong, China
xubiao@stu.edu.cn

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

***Abstract*—The ϵ-constraint handling mechanism is a widely used method for dealing with constraints. However, when the value of ϵ is reduced to a very small magnitude, it becomes equivalent to the constrained dominance principle. Under these circumstances, maintaining the diversity of the algorithm becomes challenging. To overcome this challenge, we introduce AEP-PPS, an adaptive epsilon-penalty switching for push-pull search algorithm with the adaptive stage transition mechanism. AEP-PPS employs a two-stage search strategy. In the push stage, constraints are temporarily ignored to drive the population toward the UPF. Based on the symmetric cross-generational distance indicator, the algorithm then adaptively switches to the pull stage, in which an ϵ-constraint handling mechanism combined with a penalty function is adopted to progressively enhance feasibility pressure and guide the population toward the CPF. In addition, an archive updating strategy based on θ-domination is introduced to preserve population diversity. Experimental results on the LIR-CMOP and CF test suites demonstrate that AEP-PPS outperforms nine state-of-the-art algorithms in terms of IGD and HV metrics on most of the test problems. These results confirm its effectiveness in handling complex infeasible regions while achieving a good balance between convergence and diversity.**

***Index Terms*—constrained multi-objective optimization; constraint handling mechanism; push–pull search**

I. Introduction

Many real-world problems can be formulated as constrained multi-objective optimization problems (CMOPs) [1] [2]. The primary goal of CMOPs is to approximate the Pareto optimal solution set for multiple conflicting objectives while simultaneously satisfying all prescribed constraints. Accordingly, various constrained multi-objective evolutionary algorithms (CMOEAs) have been developed, including dominance-based [3], decomposition-based [4], and indicator-based [5] algorithms. However, most existing algorithms primarily rely on single-population or single-stage optimization strategy. For instance, Gu et al. [6] enhanced constraint-handling capability at the cost of increased computational complexity. Yang et al. [7] alleviated the dependence on conventional constraint handling mechanisms by introducing an additional objective function, but this led to higher computational overhead. Yuan et al. [8] improved efficiency through indicator-based optimization, yet tended to suffer from premature convergence in complex objective spaces.

To address these problems, several multi-stage CMOEAs have been proposed, such as AT-CMOEA developed by Bao et al. [9] and CMOEA-TSRA proposed by Xia et al. [10]. These algorithms enhance global exploration by expanding the diversity of infeasible solutions in the first stage, and then shift their focus to optimizing feasible solutions in the second stage. However, their stage transitions rely on fixed parameters or empirical settings, which may cause diversity degradation or insufficient convergence, thereby degrading overall performance.

To address the above problems, an adaptive epsilon-penalty switching for push-pull search algorithm (AEP-PPS) is proposed. AEP-PPS incorporates the symmetric cross-generational distance (SCGD) indicator into the PPS framework to monitor population convergence in real time, enabling adaptive switching between the push stage and the pull stage. The main contributions are summarized as follows:

1) An adaptive stage switching strategy based on SCGD is proposed to overcome the fixed parameter switching

† These two authors contribute equally to this work.
∗ Corresponding authors.

limitation in the existing PPS framework, thereby improving the stability and efficiency of the optimization process.
2) A hybrid constraint handling strategy is developed to enhance convergence stability in complex constrained environments, which integrates the ϵ-constraint handling mechanism with a penalty function.
3) The novel AEP-PPS is proposed and experimentally validated, demonstrating significantly improved performance on the LIR-CMOP and CF test suites.

The remainder of this paper is organized as follows. Section II presents the details of the proposed algorithm. Section III describes the experimental setup and analyzes the experimental results. Section IV concludes the paper.

II. PROPOSED METHOD

Fig. 1 illustrates the workflow of the proposed AEP-PPS algorithm, which consists of two stages: the push stage and the pull stage. In the push stage, constraints are temporarily ignored, and a Tchebycheff aggregation function is employed to guide the population toward the Unconstrained Pareto Front (UPF). The transition to the pull stage is automatically triggered by the SCGD indicator. In the pull stage, the ϵ-constraint handling mechanism combined with a penalty function is adopted to drive the population toward the Constrained Pareto Front (CPF).

A. Push Stage

Given the complex structure of infeasible regions in constrained optimization problems, introducing strict constraints at an early evolutionary stage may cause search stagnation. Therefore, in the push stage, constraints are temporarily disregarded, and the population is evaluated using the Tchebycheff aggregation function, which is defined as follows:

$$g^{\text{tch}}(\mathbf{x} \mid \boldsymbol{\lambda}, \mathbf{z}) = \max_{1 \le i \le M} \{\lambda_i \, |f_i(\mathbf{x}) - z_i|\}, \tag{1}$$

where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$ denotes the weight vector, and $\mathbf{z}$ represents the ideal points.

Through this stage, the population is guided to traverse infeasible regions and rapidly approach the UPF by considering only the objective functions. This provides a solid foundation for subsequent optimization toward CPF.

B. Adaptive Stage Switching Strategy Based on SCGD

In the traditional PPS framework [11] [12], the transition from the push stage to the pull stage is often controlled by fixed parameters, which limits the adaptivity of the algorithm. If the parameter is set too large, the push stage may be unnecessarily prolonged, resulting in wasted computational resources. Conversely, an overly small parameter may trigger an early switch before the population sufficiently approximates the UPF. Consequently, this fixed switching mechanism is difficult to generalize across different problems, leading to reduced search efficiency and potential resource waste.

To address this problem, we propose the adaptive stage switching strategy based on SCGD. This strategy dynamically determines stage transitions based on population convergence. It relies on two convergence indicators, namely the forward cross-generational distance (FCGD) and the backward cross-generational distance (BCGD), which are defined as follows:

$$\text{FCGD} = \frac{1}{N} \sum_{i=1}^{N} \min_{1 \le j \le N} \|x_i^{gen} - x_j^{gen-last}\| \tag{2}$$

$$\text{BCGD} = \frac{1}{N} \sum_{j=1}^{N} \min_{1 \le i \le N} \|x_i^{gen} - x_j^{gen-last}\| \tag{3}$$

where N denotes the population size, x_i^{gen} is the i-th solution in the population at generation gen, and $x_j^{gen-last}$ is the j-th solution in the population at generation $gen - last$.

As the average value of FCGD and BCGD approaches zero, the population in the push stage is regarded as stable and converged, and the push stage subsequently switches to the pull stage. This mechanism helps prevent premature switching that may cause the search to be trapped in local optima, ensuring a timely transition after sufficient exploration of the UPF while reducing unnecessary computational cost.

C. Pull Stage

To facilitate the transition from UPF to CPF, the hybrid constraint-handling mechanism that integrates the ϵ-constraint handling mechanism with a penalty function is employed in the pull stage. In the early stage, the ϵ-constraint handling mechanism is applied with a gradually decreasing bound to prevent the population from being trapped in local optima induced by complex constraint landscapes [13]. Once ϵ decreases below a predefined threshold, the ϵ-constraint handling mechanism switches to a penalty function, where the penalty coefficient increases to effectively balance objective optimization and constraint violation. This hybrid strategy avoids diversity loss caused by overly strict constraints in the later stage, thereby ensuring more stable convergence toward the CPF.

D. Archive Updating Strategy Based on θ-Domination

Balancing convergence and diversity in the archive updating strategy based on NSGA-II [14] is a challenging task. To overcome this difficulty, we use the archive updating strategy based on θ-domination [15], which offers a more effective mechanism. Specifically, the objective space is partitioned using a set of reference vectors, and candidate solutions are ranked and selected according to the penalty boundary intersection (PBI) metric under θ-domination. By calculating the orthogonal distance between each solution and its associated reference direction, a scalar PBI function is constructed to control the trade-off between convergence and distribution. During archive updating, solutions with smaller PBI values are preferentially retained, and additional solutions are selected in ascending order of PBI values until the archive capacity is reached. This strategy effectively enhances selection pressure in complex optimization scenarios and improves the quality of the final solution set.

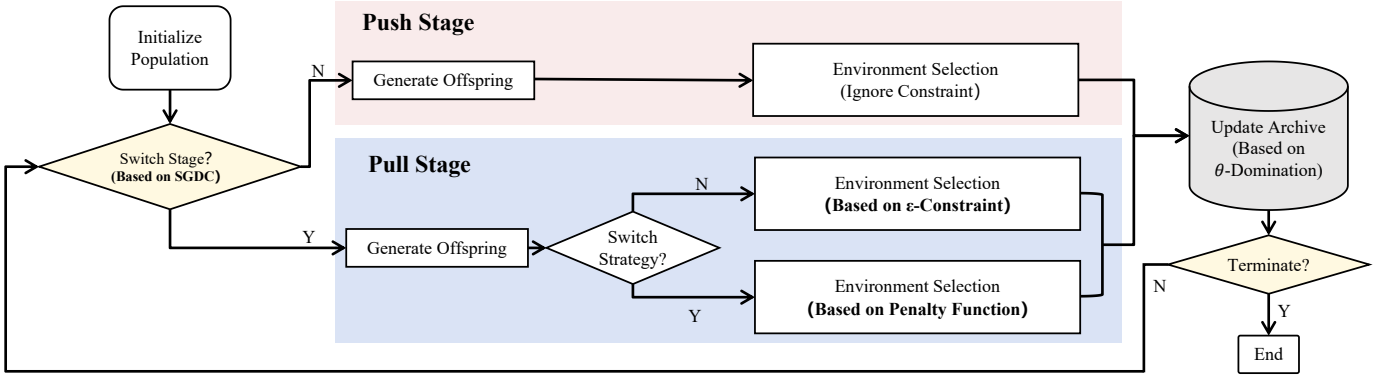


Fig. 1: Procedures of the proposed AEP-PPS.

The coordinated operation of these mechanisms enables AEP-PPS to sustain a dynamic balance between convergence and diversity throughout the optimization process.

E. Computational Complexity

It is noteworthy that the incorporation of the archive updating strategy based on θ-domination, together with the adaptive stage switching strategy based on SCGD, introduces additional computational overhead. This overhead primarily arises from individual comparisons and distance calculations.

F. Pseudocode

Based on the above mechanisms, the overall procedure of AEP-PPS is summarized in Algorithm 1. First, an initial population of size N is generated (line 1). In the push stage, the population convergence is evaluated by computing $Change_{\max}$ based on SCGD, and environmental selection is performed using the Tchebycheff function (lines 5–7). When $Change_{\max}$ falls below a predefined threshold, the push stage switches to the pull stage and initializes the ϵ parameter (lines 8–11). In the pull stage, the ϵ-constraint handling mechanism first guides the population from UPF toward CPF (lines 13–14). Once ϵ decreases below a specified threshold, environmental selection is then conducted using a penalty function (lines 15–20). At the end of each generation, the archive is updated according to the θ-domination principle (line 22).

Algorithm 1: Procedure of AEP-PPS

Input: Population size N, Number of objectives M
Output: Pop_{archive}

1 $Pop \leftarrow \text{Initialize}(N, M)$;
2 $Stage \leftarrow \text{Push}$;
3 **while** *termination condition is not met* **do**
4 $Off \leftarrow$ Generate offspring;
5 **if** $Stage \leftarrow \text{Push}$ **then**
6 Calculate $Change_{\max}$ based on SCGD;
7 Environmental selection based on Tchebycheff;
8 **if** $Change_{max} < Threshold$ **then**
9 $Stage \leftarrow \text{Pull}$;
10 $\epsilon \leftarrow CV_{\max}$;
11 **end**
12 **else**
13 **if** *no penalty* **then**
14 Environmental selection based on ϵ-constrain ;
15 **if** $\epsilon < Threshold$ **then**
16 $k \leftarrow k_0$;
17 **end**
18 **else**
19 Environmental selection based on penalty function;
20 **end**
21 **end**
22 Update Pop_{archive} based on θ-domination;
23 **end**
24 **return** Pop_{archive};

III. EXPERIMENTAL STUDY

A. Experimental Setup

To evaluate the performance of the proposed AEP-PPS algorithm, comparative experiments are conducted on the LIR-CMOP test suite [16] and CF test suite [17] against nine representative CMOEAs, including BiCo [18], EMCMO [19], CCMO [20], cDPEA [21], ToP [22], PPS [11], IMCMOEAD [23], LCMEA [24] and APSEA [25]. The experimental settings are as follows:

- For LIR-CMOP1–14, the number of decision variables is set to 30, and the population size is set to 300.
- For CF1–10, the number of decision variables is set to 10, and the population size is set to 300.
- Each algorithm is independently executed 30 times for each test problem, with 300,000 function evaluations per run.
- The key parameters of the compared algorithms are configured based on the recommendations provided in their original studies.

B. Experimental Results

On the LIR-CMOP1–14 benchmark set and CF1-10 benchmark set, AEP-PPS and the nine compared algorithms are independently run 30 times, with the IGD and HV results summarized in Table I, where the best values are highlighted in bold. In the Wilcoxon rank-sum test, '+', '−', and '=' indicate whether the compared algorithm performs statistically significantly better than, worse than, or is comparable to AEP-PPS. Overall, AEP-PPS outperforms the other nine algorithms on most test problems.

To comprehensively evaluate the performance of the proposed algorithm, two representative benchmark problems, LIR-CMOP1 and LIR-CMOP8 were selected. LIR-CMOP1 is characterized by a narrow feasible region and disjoint CPF and UPF. This complex landscape frequently traps the evolving population in local feasible regions, thereby impeding convergence toward the true CPF (see Fig. 2 for comparative results). Conversely, LIR-CMOP8 features expansive infeasible regions that act as substantial barriers, severely hindering the population from approximating the CPF, as illustrated in Fig. 3. To further elucidate the advantages of AEP-PPS, the fundamental differences between the proposed approach and existing methods are discussed below from three distinct perspectives.

Firstly, from the perspective of stage transition mechanisms, existing multi-stage algorithms largely rely on empirical parameters or rigid preset rules, which frequently cause transitions to occur prematurely or belatedly. For instance, as observed in Fig. 2(e), ToP triggers the second stage before achieving adequate convergence toward the UPF, resulting in an incomplete CPF. Furthermore, its excessively strong constraint selection pressure (Fig. 3(e)) traps the population near feasible boundaries for extended periods, severely hindering its ability to traverse large infeasible regions. A similar limitation is evident in the classic PPS algorithm (Fig. 2(f) and Fig. 3(f)). Although the original push-pull framework provides a robust optimization foundation, its reliance on fixed rules for stage division lacks the agility to dynamically respond to the population's real-time evolutionary state, thereby restricting its exploration capabilities in complex constrained environments. Likewise, APSEA (Fig. 2(i) and Fig. 3(i)) struggles with premature convergence to local optima; its preset transition rules fail to capture rapid shifts in population distribution, ultimately degrading overall search efficiency. In contrast, AEP-PPS overcomes these bottlenecks by introducing an adaptive stage-switching strategy driven by SCGD. This novel mechanism enables precise, state-aware transitions that are strictly governed by the real-time convergence dynamics of the population.

Secondly, from the perspective of population collaboration mechanisms, existing multi-population strategies often evolve in parallel with limited inter-population synergy. For instance, both EMCMO and CCMO (Fig. 2(b)–(c) and Fig. 3(b)–(c)) fail to achieve uniform coverage of the UPF. This deficiency stems from their constraint-handling mechanisms, which rely heavily on binary feasibility discrimination or isolated stage-dependent strategies. The absence of continuous, dynamic adjustment for constraint effects frequently leads to insufficient exploration and diminished convergence efficiency in complex environments. Furthermore, although BiCo attempts to enhance boundary-crossing capabilities through dual-population co-evolution, its performance remains heavily reliant on specific population structures. Consequently, it struggles to yield a uniformly distributed nondominated solution set within narrow feasible regions (Fig. 2(a)) and exhibits inadequate exploration of expansive infeasible barriers (Fig. 3(a)). Similarly, the feasibility-first strategy of cDPEA (Fig. 2(d) and Fig. 3(d)) drives the population to prematurely concentrate in feasible regions, exacerbating the uneven distribution of solutions. To overcome these disjointed search behaviors, AEP-PPS leverages its refined push-pull architecture to establish deep sequential collaboration. During the push stage, infeasible solutions are fully exploited to comprehensively explore the broader search space. Subsequently, in the pull stage, the population is precisely guided back toward the true CPF.

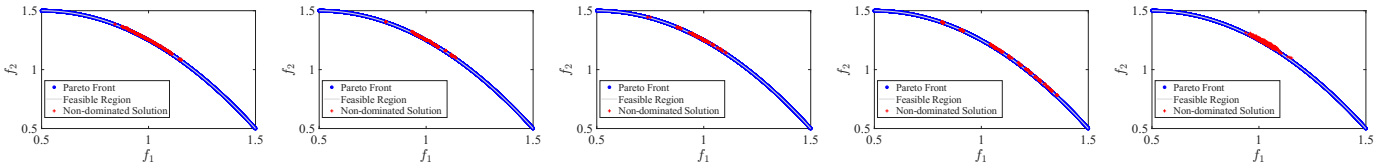
Thirdly, from the perspective of constraint-handling strategies, existing specialized algorithms often exhibit high sensitivity to initial solutions or predefined reference vectors. LCMEA, for instance (Fig. 2(h) and Fig. 3(h)), struggles to identify the complete Pareto front. While it employs a dynamic relaxation mechanism to balance exploration and convergence by progressively tightening constraints, this adjustment relies heavily on rigid, predefined mathematical formulations. Consequently, it lacks the fine-grained, adaptive calibration required to respond to the population's real-time evolutionary state. Furthermore, IMCMOEAD (Fig. 2(g) and Fig. 3(g)) demonstrates a propensity for premature convergence to local optima and struggles to traverse infeasible barriers. Although embedding constraint handling into a decomposition and neighborhood replacement framework enhances optimization efficiency to some degree, its overall performance remains hypersensitive to weight distribution and subproblem coordination. To address the rigidities and sensitivities of existing methods, AEP-PPS adopts a highly generalized ϵ-constraint mechanism augmented by a penalty function, which achieves a better balance between search freedom and strict feasibility guidance.

Furthermore, we construct a comparative algorithm, AEP-PPS-SR, which employs a stochastic ranking strategy. The results presented in Table II indicate that, on the LIR-CMOP test suite, the original AEP-PPS algorithm significantly outperforms AEP-PPS-SR in terms of both HV and IGD indicators for most test problems. These experimental findings provide strong evidence that the ϵ-constraint mechanism based on a penalty function in our study is more effective in maintaining solution diversity and distribution quality.

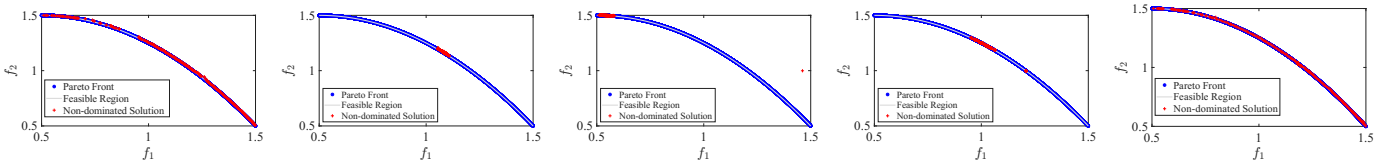
In summary, AEP-PPS is not merely a combination of existing strategies. Rather, it represents a systematic reconstruction. This reconstruction addresses the limitations of prior mechanisms in stage-switching timing, information collaboration, and environmental robustness.

TABLE I: Statistical Comparison of Algorithms on LIRCMOP and CF Test Suites Using IGD and HV Metrics

Test Problem Set	Metrics	BiCo	EMCMO	CCMO	cDPEA	ToP	PPS	IMCMOEAD	LCMEA	APSEA	AEP-PPS
LIRCMOP1-14	IGD	0/14/0	2/12/0	2/12/0	2/12/0	0/14/0	3/10/1	0/14/0	0/14/0	0/14/0	-
	HV	0/14/0	2/12/0	2/12/0	2/12/0	0/14/0	5/8/1	0/14/0	0/14/0	0/14/0	-
CF1-10	IGD	0/8/2	1/8/1	1/6/3	2/5/3	1/9/0	4/4/2	1/8/1	2/7/1	1/7/2	-
	HV	1/6/1	0/7/3	0/8/2	0/7/3	1/7/0	3/6/1	0/6/3	2/8/0	0/8/2	-

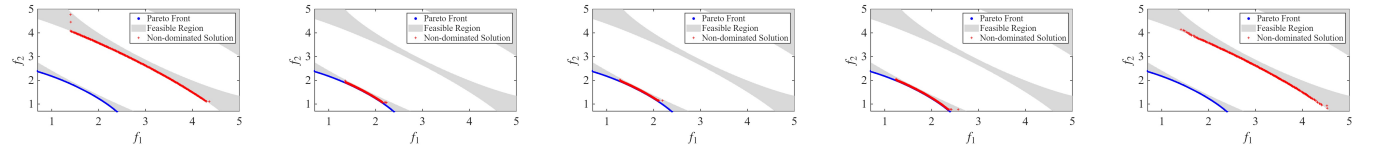


(a) BiCo (b) EMCMO (c) CCMO (d) cDPEA (e) ToP

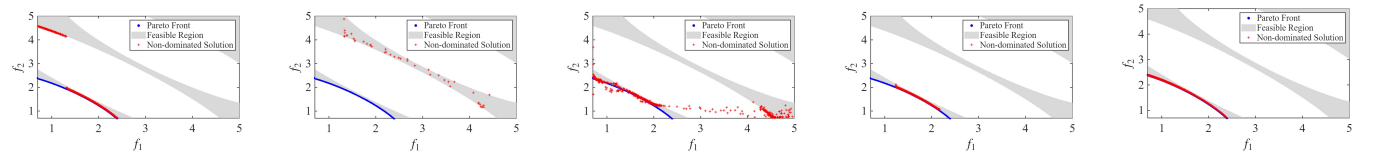


(f) PPS (g) IMCMOEAD (h) LCMEA (i) APSEA (j) AEP-PPS

Fig. 2: Results obtained by each algorithm on LIRCMOP1. The gray area represents the feasible region. The blue lines indicate the CPF and the red points denote the obtained non-dominated solutions.



(a) BiCo (b) EMCMO (c) CCMO (d) cDPEA (e) ToP



(f) PPS (g) IMCMOEAD (h) LCMEA (i) APSEA (j) AEP-PPS

Fig. 3: Results obtained by each algorithm on LIRCMOP8. The gray area represents the feasible region. The blue lines indicate the CPF and the red points denote the obtained non-dominated solutions.

TABLE II: Statistical Comparison of AEP-PPPS-SR and AEP-PPS on LIRCMOP Test Suites Using IGD and HV Metrics

Test Problem Set	Metrics	AEP-PPS-SR	AEP-PPS
LIRCMOP1-14	IGD	2/12/0	-
	HV	2/12/0	-

IV. Conclusion

For problems with complex infeasible regions, excessive emphasis on convergence or insufficient constraint handling can hinder accurate approximation of the CPF. To address this problem, AEP-PPS is proposed. In the push stage, constraints are temporarily ignored to explore the overall structure of the UPF along reference directions. The algorithm then adaptively switches to the pull stage, where the ϵ-constraint and penalty function are applied. Experimental results demonstrate the effectiveness and stability of AEP-PPS in complex constrained scenarios.

AEP-PPS performs exceptionally well under complex constraints but needs improved responsiveness to search dynamics in extreme scenarios. Future research will focus on utilizing convergence, evolution, and distribution data to develop adaptive coordination mechanisms that enhance both robustness and adaptability.

ACKNOWLEDGMENT

This work is supported in part by the National Science and Technology Major Project (grant numbers 2021ZD0111502), the National Natural Science Foundation of China (grant numbers 62406186, 62176147, 62476163, 62441612), the Natural Science Foundation of Guangdong Province (grant numbers 2025A1515010800, 2024A1515012450), Guangdong Basic and Applied Basic Research Foundation (grant numbers 2023B1515120020).

REFERENCES

[1] W. Li, Z. Wang, C. Guan, C. Chen, B. Wang, P. Ren, Y. Qiu, Q. Zhang, H. Wang, D. Wang *et al.*, "Formation control of swarm robotics: A survey from biological inspirations to design automation methods," *Robotics and Autonomous Systems*, p. 105245, 2025.

[2] W. Li, Z. Wang, R. Mai, P. Ren, Q. Zhang, Y. Zhou, N. Xu, J. Zhuang, B. Xin, L. Gao *et al.*, "Modular design automation of the morphologies, controllers, and vision systems for intelligent robots: a survey," *Visual Intelligence*, vol. 1, no. 1, p. 2, 2023.

[3] W. Zhang, J. Liu, J. Liu, Y. Liu, and S. Tan, "A strengthened constrained-dominance based evolutionary algorithm for constrained many-objective optimization," *Applied Soft Computing*, vol. 167, p. 112428, 2024.

[4] Y. Wang, Y. Zhang, C. Xu, W. Bai, K. Zheng, and W. Dong, "Decomposition-based dual-population evolutionary algorithm for constrained multi-objective problem," *Swarm and Evolutionary Computation*, vol. 95, p. 101912, 2025.

[5] X. Zhong, X. Yao, K. Qiao, D. Gong, and Y. Jin, "An indicator-based evolutionary algorithm for large-scale constrained multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 30, pp. 271–285, 2025.

[6] Q. Gu, J. Bai, X. Li, N. Xiong, and C. Lu, "A constrained multi-objective evolutionary algorithm based on decomposition with improved constrained dominance principle," *Swarm and Evolutionary Computation*, vol. 75, p. 101162, 2022.

[7] Y. Yang, P.-Q. Huang, X. Kong, and J. Zhao, "A constrained multi-objective evolutionary algorithm assisted by an additional objective function," *Applied Soft Computing*, vol. 132, p. 109904, 2023.

[8] J. Yuan, H.-L. Liu, Y.-S. Ong, and Z. He, "Indicator-based evolutionary algorithm for solving constrained multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 2, pp. 379–391, 2021.

[9] Q. Bao, M. Wang, G. Dai, X. Chen, Z. Song, and S. Li, "An archive-based two-stage evolutionary algorithm for constrained multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 75, p. 101161, 2022.

[10] M. Xia, Q. Chong, and M. Dong, "A constrained multi-objective evolutionary algorithm with two-stage resources allocation," *Swarm and Evolutionary Computation*, vol. 79, p. 101313, 2023.

[11] 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.

[12] Z. Fan, Z. Wang, W. Li, Y. Yuan, Y. You, Z. Yang, F. Sun, and J. Ruan, "Push and pull search embedded in an M2M framework for solving constrained multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 54, p. 100651, 2020.

[13] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Difficulty adjustable and scalable constrained multiobjective test problem toolkit," *Evolutionary Computation*, vol. 28, no. 3, pp. 339–378, 2020.

[14] 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.

[15] Y. Echevarría, L. Sánchez, and C. Blanco, "Genetic fuzzy modelling of Li-Ion batteries through a combination of Theta-DEA and knowledge-based preference ordering," in *Conference of the Spanish Association for Artificial Intelligence*, vol. 9868, 2016, pp. 310–320.

[16] Z. Fan, W. Li, X. Cai, H. Huang, Y. Fang, Y. You, J. Mo, C. Wei, and E. Goodman, "An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions," *Soft Computing*, vol. 23, pp. 12 491–12 510, 2019.

[17] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, S. Tiwari *et al.*, "Multiobjective optimization test instances for the cec 2009 special session and competition," *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, vol. 264, pp. 1–30, 2008.

[18] Z.-Z. Liu, B.-C. Wang, and K. Tang, "Handling constrained multiobjective optimization problems via bidirectional coevolution," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10 163–10 176, 2021.

[19] K. Qiao, K. Yu, B. Qu, J. Liang, H. Song, and C. Yue, "An evolutionary multitasking optimization framework for constrained multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 2, pp. 263–277, 2022.

[20] 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.

[21] M. Ming, A. Trivedi, R. Wang, D. Srinivasan, and T. Zhang, "A dual-population-based evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 739–753, 2021.

[22] Z.-Z. Liu and Y. Wang, "Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 870–884, 2019.

[23] L. R. Farias and A. F. Araújo, "An inverse modeling constrained multi-objective evolutionary algorithm based on decomposition," in *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2024, pp. 3727–3732.

[24] L. Si, X. Zhang, Y. Zhang, S. Yang, and Y. Tian, "An efficient sampling approach to offspring generation for evolutionary large-scale constrained multi-objective optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 9, no. 3, pp. 2080–2092, 2025.

[25] Y. Tian, R. Wang, Y. Zhang, and X. Zhang, "Adaptive population sizing for multi-population based constrained multi-objective optimization," *Neurocomputing*, vol. 621, p. 129296, 2025.