

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

IMPORTANT

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

KNOWLEDGE-DRIVEN EVOLUTIONARY
ALGORITHMS FOR MULTIOBJECTIVE
OPTIMIZATION PROBLEMS

ZHENZHONG WANG

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University
Department of Computing

Knowledge-Driven Evolutionary Algorithms for Multiobjective
Optimization Problems

Zhenzhong WANG

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

June 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Zhenzhong WANG

Abstract

Multiobjective optimization is an indispensable task in a plethora of real-world applications such as logistics, manufacturing, and finance. In recent years, real-world applications with big data keep arising in various fields, leading to more complex multiobjective optimization problems of large-scale, dynamic, and multitasking nature. On the other hand, evolutionary algorithms, inspired by biological evolution mechanisms, have been widely used in many real-world applications to solve complex optimization problems, as they can find a set of approximately Pareto optimal solutions but without requiring domain-specific information such as continuous, differentiable, and unimodal functions. Despite the great success enjoyed by evolutionary algorithms, it is worth noting that traditional evolutionary algorithms often conduct the search at a “ground zero” knowledge state, leading to unnecessary computational costs and inefficient search. In fact, the optimization knowledge and search experience during the search process could be harnessed for effective problem-solving. Therefore, to effectively and efficiently cope with the more complex multiobjective optimization problems, creating knowledge-driven evolutionary algorithms is deemed desirable and much expected by learning useful traits of problem-solving experiences. Keeping these in mind, this thesis aims to conduct research on knowledge-driven evolutionary algorithms for solving multiobjective optimization problems such as large-scale multiobjective optimization problems, dynamic multiobjective optimization problems, dynamic constrained multiobjective optimization problems, and multitask multiobjective optimization problems.

Firstly, to solve large-scale multiobjective optimization problems, traditional evolutionary algorithms fail to learn from high-quality solutions or useful traits, leading to inefficient exploration of the large-scale search space. The reproduced offspring are often near their parents, suffering from poor diversity and local optima. To address this issue, the thesis presents a knowledge-driven offspring generation operator that learns knowledge from non-dominated solutions. Specifically, the thesis points out that Pareto optimal solutions follow a low-dimensional manifold. By learning from the manifold, high-quality offspring solutions can be generated via a dedicated generative adversarial network, thereby improving the convergence and diversity of the offspring solutions. Experimental results demonstrate that significant improvements have been achieved by the proposed knowledge-driven offspring generation method.

Secondly, to solve dynamic multiobjective optimization problems, existing evolutionary algorithms fall short of leveraging the environmental knowledge to adaptively adjust search strategies according to the changing environments, thus making the computational resources wasted on the suboptimal subpopulations and improper evolutionary operators. To exploit the computational resources, this thesis presents a knowledge-driven Monte Carlo tree-assisted multi-population algorithm. The Monte Carlo tree activates promising subpopulations to search for promising search regions. On the other hand, the evolutionary operator of the activated subpopulation can be adaptively switched according to the changing environments. Experimental results on test functions have demonstrated the efficacy of the proposed method in solving dynamic optimization problems.

Furthermore, to solve dynamic constrained multiobjective optimization problems, existing techniques could force infeasible solutions toward partially feasible regions, leading to the loss of diversity. They did not utilize knowledge in constrained/unconstrained search space or dynamic environments to accelerate the search for feasible solutions. To address the issue, this thesis proposes a spatial-temporal knowledge-based dynamic response strategy. Specifically, two synergistic tasks are established: one task is focused on the unconstrained search space to

maintain diversity, while the other involves searching the constrained search space to focus on feasibility. Particularly, two knowledge transfer modules, i.e., the spatial knowledge transfer module and temporal knowledge transfer module, are designed to leverage the knowledge of the historical dynamic environment and constrained/unconstrained search space. Additionally, to advance the test suite toward real-world cases, the thesis designs 14 dynamic constrained multiobjective optimization test problems with various properties. Experiments conducted on the proposed test problems and a real-world problem have demonstrated the efficacy of the proposed algorithm.

Next, to solve multitask optimization problems, a plethora of evolutionary multitasking algorithms have been developed to enhance positive transfer among optimization tasks. Nevertheless, most of these methods conduct knowledge transfer by transferring the best solutions from a source task to the target task while ignoring the proper use of knowledge from the target task in solution selection. As a result, the transferred solutions could not adapt well to the target task, thus limiting the effectiveness of knowledge transfer across tasks. To address this issue, this thesis crafts a knowledge-driven solution selection method for evolutionary multitasking. By learning the solution distribution of both source and target tasks, a number of high-quality solutions can be selected and transferred to enhance positive transfer for the target task. Experimental results on multitasking benchmarks and a real-world application have demonstrated the effectiveness of the proposed method.

Lastly, this thesis develops a dedicated knowledge-driven multiobjective optimization algorithm for fair graph learning. This work aims to generate diagnostic and actionable explanations to address the problem of “what subgraph patterns cause the biased behavior of graph neural networks and what actions practitioners could take to rectify the bias.” By formulating the explanation generation problem as a multiobjective combinatorial optimization problem, a multiobjective evolutionary algorithm is presented to ensure classification performance, explainability, and fairness of graph neural networks in one go. In particular, an influenced nodes-based

gradient approximation is developed to boost the computation efficiency of the evolutionary algorithm. Extensive experiments have been conducted to demonstrate the proposed method’s superiority in classification performance, fairness, and explainability for fair graph learning.

List of Publications

1. Zhenzhong Wang, Dejun Xu, Min Jiang, and Kay Chen Tan, “Spatial-Temporal Knowledge Transfer for Dynamic Constrained Multiobjective Optimization”, in *IEEE Transactions on Evolutionary Computation* (2024).
2. Zhenzhong Wang, Lulu Cao, Liang Feng, Min Jiang, and Kay Chen Tan, “Evolutionary Multitask Optimization with Lower Confidence Bound-Based Solution Selection Strategy”, in *IEEE Transactions on Evolutionary Computation* (2024).
3. Zhenzhong Wang, Qingyuan Zeng, Wanyu Lin, Min Jiang, and Kay Chen Tan, “Generating Diagnostic and Actionable Explanations for Fair Graph Neural Networks”, in *38th AAAI Conference on Artificial Intelligence* (2024).
4. Zhenzhong Wang, Lulu Cao, Wanyu Lin, Min Jiang, and Kay Chen Tan, “Robust Graph Meta-Learning via Manifold Calibration with Proxy Subgraphs”, in *37th AAAI Conference on Artificial Intelligence* (2023).
5. Zhenzhong Wang, Qingyuan Zeng, Wanyu Lin, Min Jiang, and Kay Chen Tan, “Multi-View Subgraph Neural Networks: Self-Supervised Learning with Scarce Labeled Data”, in *IEEE Transactions on Neural Networks and Learning Systems* (2024).
6. Zhenzhong Wang, Haokai Hong, Kai Ye, Guang-En Zhang, Min Jiang, and Kay Chen Tan, “Manifold Interpolation for Large-Scale Multiobjective Optimization via Generative

- Adversarial Networks”, in *IEEE Transactions on Neural Networks and Learning Systems* (2023).
7. Min Jiang, Zhenzhong Wang, Haokai Hong, and Gary G Yen, “Knee Point-Based Imbalanced Transfer Learning for Dynamic Multiobjective Optimization”, in *IEEE Transactions on Evolutionary Computation* (2020).
 8. Min Jiang, Zhenzhong Wang, Liming Qiu, Shihui Guo, Xing Gao, and Kay Chen Tan, “A Fast Dynamic Evolutionary Multiobjective Algorithm via Manifold Transfer Learning”, in *IEEE Transactions on Cybernetics* (2020).
 9. Min Jiang, Zhenzhong Wang, Shihui Guo, Xing Gao, and Kay Chen Tan, “Individual-Based Transfer Learning for Dynamic Multiobjective Optimization”, in *IEEE Transactions on Cybernetics* (2020).
 10. Zhenzhong Wang, Zehui Lin, Wanyu Lin, Ming Yang, Minggang Zeng, and Kay Chen Tan, “Explainable Molecular Property Prediction: Aligning Chemical Concepts with Predictions via Language Models”, *Arxiv* (2024).
 11. Zhenzhong Wang, Haowei Hua, Wanyu Lin, Ming Yang, and Kay Chen Tan, “Crystalline Material Discovery in the Era of Artificial Intelligence”, *Arxiv* (2024).

Acknowledgments

Looking back on my path of doctoral journey, I am overwhelmed with a deep sense of gratitude toward the many individuals who have profoundly impacted my academic and personal growth.

First and foremost, I extend my sincerest gratitude to my supervisors, Prof. Kay Chen Tan and Dr. Wanyu Lin. It is my honor to learn from you and become your student. Your unwavering guidance and profound wisdom have been fundamental to this endeavor. Your patience, mentorship, and faithful support have not only shaped this thesis but also profoundly influenced my growth as a researcher. You have acted as a role model, and I will continue to learn from you in my future scientific research and daily life.

I would also like to express my utmost gratitude to Prof. Min Jiang, who guided me throughout my Master's program. It was under your mentorship that I delved into the realm of research and developed a passion for scientific inquiry. In addition, I am profoundly thankful to Prof. Liang Feng for your valuable insights and constructive critiques, which have enhanced my comprehension of this research field. I would also like to extend my deepest gratitude to Dr. Jibin Wu, your input and guidance have pushed me and shaped me into a better researcher. I deeply value your devoted support towards my academic endeavors.

I am also kindly thankful for the support from my collaborators at MindLab: Dr. Yujie Wu, Dr. Zhi-An Huang, Dr. Songbai Liu, Dr. Xun Zhou, Dr. Rui Liu, Dr. Yinglan Feng, Dr. Xingyu Wu, Dr. Yuxiao Huang, Dr. Shenghao Wu, Mr. Xiaoming Xue, Mr. Yao Hu, Mr. Wu Lin,

Ms. Xinyi Chen, Mr. Chenxiang Ma, Mr. Haokai Hong, Mr. Xiang Hao, Mr. Ziyuan Ye, Mr. Beichen Huang, Mr. Yinsong Yan, Mr. Yu Zhou, Ms. Yajie Zhang, Mr. Shimin Zhang, Ms. Mengqi Xu, and many others who are not listed herein. I am deeply thankful for the moments of joy that have made this journey intellectually stimulating and emotionally fulfilling. Your support and shared passion for research have been a continuous source of inspiration.

Lastly, thanks must go to my family. Especially my parents and my beloved wife, Yun. I am unable to adequately express the depth of my gratitude for your unwavering love, encouragement, and belief in me. Your enduring support has been the foundation that enabled me to pursue my dreams and overcome challenges along the way. Your unwavering dedication and companionship have been a wellspring of strength. I am forever grateful to all of you for bringing me to this significant moment.

In conclusion, this thesis represents years of hard work. It stands as a testament to the invaluable support, guidance, and encouragement of the many individuals who have accompanied me on this journey. To all of you, I offer my deepest and most heartfelt gratitude.

Table of Contents

Abstract	i
List of Publications	v
Acknowledgments	vii
List of Figures	xv
List of Tables	xxii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Contributions	4
1.3 Thesis Organization	7
2 Preliminaries on Multiobjective Optimization Problems	9
2.1 Large-Scale Multiobjective Optimization	9
2.2 Dynamic Multiobjective Optimization	12

2.3	Dynamic Constrained Multiobjective Optimization	16
2.4	Multitask Multiobjective Optimization	21
3	Knowledge-Driven Offspring Generation Operator for Large-Scale Optimization	24
3.1	Motivation	24
3.2	Proposed Framework	28
3.2.1	Central Solutions Identification	29
3.2.2	Manifold Interpolation	30
3.2.3	Manifold Selection Mechanism	33
3.2.4	Complexity Analysis	34
3.3	Experiments	36
3.3.1	Experimental Settings	36
3.3.2	Comparisons with Baselines	37
3.3.3	Ablation Studies	39
3.3.4	Performance of Different Interpolation Strategies	42
3.3.5	Sensitivity Studies	43
3.3.6	Discussion	45
3.4	Conclusion	47
4	Knowledge-Driven Adaptive Resource Allocation for Dynamic Multiobjective Optimization	49

4.1	Motivation	49
4.2	Proposed Framework	52
4.2.1	Overview	53
4.2.2	Cluster-Based Subpopulation Partition	54
4.2.3	Monte Carlo Tree-Assisted Computational Resource Allocation	56
4.2.4	Complexity Analysis	61
4.3	Experiments	61
4.3.1	Experimental Settings	61
4.3.2	Comparisons with Baselines	64
4.3.3	Sensitivity Studies	69
4.3.4	Runtime Performance	74
4.4	Conclusion	76
5	Knowledge-Driven Spatial-Temporal Knowledge Transfer for Dynamic Constrained Multiobjective Optimization	77
5.1	Motivation	77
5.2	Proposed Framework	82
5.2.1	Overview	82
5.2.2	Spatial Knowledge Transfer	83
5.2.3	Temporal Knowledge Transfer	86
5.2.4	Complexity Analysis	89

5.3	Proposed Benchmark	89
5.4	Experiments	103
5.4.1	Experimental Settings	103
5.4.2	Comparisons with Baselines	106
5.4.3	Sensitivity Studies	110
5.4.4	Real-World Applications	111
5.5	Conclusion	116
6	Knowledge-Driven Solution Selection Strategy for Evolutionary Multitask Optimization	117
6.1	Motivation	117
6.2	Preliminaries	121
6.3	Proposed Framework	123
6.3.1	Overview	123
6.3.2	Training Process of Transfer Gaussian Process	123
6.3.3	Solution Selection Module	126
6.3.4	Incorporation with Existing Mutitask Optimization Algorithms	129
6.4	Experiments	131
6.4.1	Experimental Settings	131
6.4.2	Comparisons with Baselines	132
6.4.3	Ablation Studies	140

6.4.4	Generality Analysis	144
6.4.5	Real-World Applications	145
6.5	Conclusion	150
7	Knowledge-Driven Multiobjective Combinatorial Optimization for Explainable Fair Graph Neural Networks	151
7.1	Motivation	151
7.2	Preliminaries	154
7.3	Proposed Framework	157
7.3.1	Identification of Biased Subgraphs	158
7.3.2	Influenced Nodes-Based Gradient Approximation	159
7.3.3	Generation of Counterfactual Debiased Subgraphs	162
7.4	Theoretical Analysis	163
7.5	Experiments	168
7.5.1	Experimental Settings	168
7.5.2	Comparisons with Baselines	169
7.5.3	Ablation Studies	173
7.5.4	Sensitivity Studies	174
7.6	Conclusion	177
8	Summary and Future Work	178
8.1	Summary	178

8.2 Future Work 180

References **182**

List of Figures

3.1	The probability distribution of β of simulated binary crossover [1].	25
3.2	Solutions interpolation on the manifold.	27
3.3	The framework of the proposed GAN-LMEF. Step 1. Algorithm 2: Map non-dominated solutions $\mathcal{F}$ into the $m - 1$ -dimensional manifold, cluster them into several clusters, and identify the central solution C_i for each cluster. Step 2. Algorithm 3: A GAN is utilized to interpolate solutions between the central solutions C_i . Step 3. Algorithm 4: Select the excellent solutions from the original population and interpolated solutions to form a new population to guide future searches.	28
3.4	The flowchart of the manifold interpolation model. Step 1. Train the GAN with non-dominated solutions $\mathcal{F}$ and noise variables z . Step 2. Save the well-trained <i>generator</i> : G . Step 3. Apply interpolation strategies to z and input these interpolating noise variables into the <i>generator</i> : G . Step 4. Generate interpolating solutions Q	32
3.5	The proposed manifold selection mechanism: For solutions A and C , C_1 and C_2 are the two nearest central solutions, and for B , C_2 and C_3 are the two nearest central solutions. If two solutions should be selected, A and B are selected because they have the two minimum manifold distances.	33

3.6	The selection process. Step 1. Select solutions from Q as Q' by the manifold selection mechanism. Step 2. Evaluate the solutions in Q' . Step 3. Replace solutions in P with those in Q'	34
3.7	Convergence profiles of IGD values obtained by GAN-NSGA2 and NSGA2 on LSMOP problems with 500 decision variables.	42
3.8	The number of selected non-dominated solutions that are interpolated by different strategies on LSMOP problems with 500 decision variables.	43
3.9	The influence of the proportion of interpolated solutions σ and the first evaluations for interpolating $\epsilon \cdot e$	45
3.10	The relationship between the number of decision variables and the wall-clock time of executing Line 4-9 in Algorithm 1 once.	47
4.1	Limitations of existing DMOEAs. (a) When the environment changes from $t = 0$ to $t = 1$, keeping the subpopulation 1 activated could lead to the local optimum. (b) The search strategies of the activated subpopulation should be adaptively changed according to the changing environment. Otherwise, the population could be trapped in local optimum or slow convergence.	51
4.2	The whole population is first divided into several subpopulations. Then, MCTS determines which subpopulation should be activated and the specific evolutionary operator of the corresponding activated subpopulation.	54

4.3	Monte Carlo Tree-assisted computational resource allocation. (1) Selection: Recursively, a node is selected according to the UCB value until the selected node is a leaf node. (2) Expansion: The selected leaf node reproduces two children nodes by using SBX and DE operators. (3) Simulation: An offspring node is selected for simulation with a minimum batch of computational resources. (4) Backpropagation: UCB values of nodes along the selection path are updated. . .	57
4.4	Convergence traces of average MIGD values obtained by competing algorithms under test problems DF1-DF7 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$). .	67
4.5	Convergence traces of average MIGD values obtained by competing algorithms under test problems DF8-DF14 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$). .	68
4.6	Convergence traces of average MIGD values obtained by MCTS-DMOEA, MCTS-DMOEA ₁ and MCTS-DMOEA ₂ under test problems DF1-DF7 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).	69
4.7	Convergence traces of average MIGD values obtained by MCTS-DMOEA, MCTS-DMOEA ₁ and MCTS-DMOEA ₂ under test problems DF8-DF14 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).	70
4.8	The proportion of allocation of computing resources to SBX and DE operators. .	74
5.1	Limitations of existing methods in solving DCMOPs. (a) Adaptive penalty mechanisms in existing DCMOAs could lead to the loss of diversity when dealing with DCMOPs with narrow or disconnected feasible areas. (b) An evolutionary multitasking-based static constrained multiobjective optimization algorithm (MTCMO) [2] that is equipped with the dynamic response strategies in DNSGA2 [3] suffers from slow convergence when dealing with DCMOPs. . . .	79

5.2	The framework of the proposed DC-MFEA. (1) Spatial knowledge transfer: An auxiliary task is derived from the main task (the original DCMOP), and then the two tasks are encoded and optimized in the unified search space, where the spatial knowledge is implicitly transferred by genetic operators. (2) Temporal knowledge transfer: When the environment changes, solutions from the constrained search space and solutions from the unconstrained search space are archived in archives A^C and A^U , respectively. According to the estimated similarities between two consecutive environments, solutions in the two archives are reused for the new environment.	83
5.3	The objective space of (a) DCF1 and (b) DCF2.	92
5.4	The objective space of (a) DCF3 and (b) DCF4.	94
5.5	The objective space of (a) DCF5 and (b) DCF6.	95
5.6	The objective space of (a) DCF7 and (b) DCF9.	97
5.7	The objective space of (a) DCF8 and (b) DCF10.	98
5.8	The objective space of (a) DCF11 and (b) DCF12.	101
5.9	The objective space of (a) DCF13 and (b) DCF14.	102
5.10	Convergence traces of average IGD values for all environmental steps obtained by compared algorithms with the dynamic test setting ($n_t = 10, \tau_t = 20$). . . .	109
5.11	Convergence traces of average HV values obtained by compared algorithms with the dynamic test setting ($n_t = 10, \tau_t = 20$)	110

6.1	EMT for minimizing the single-objective function of two tasks. When an ideal source-target mapping is learned, only transferring the top three best solutions (x_2 , x_3 , and x_4) while falling short of exploiting the solution distribution in the target task could lead to local optima. On the contrary, by leveraging the solution distribution in the target task, transferring x_1 exhibits the potential of escaping from local optima as the region nearby x_1 is less explored than other regions.	119
6.2	The framework of the proposed solution selection approach. With the training data from the two tasks, a TGP model respective to the target task is trained, where the inter-task similarity is derived simultaneously. Solution Selection Module: The LCB values of the solutions are synthesized using the TGP model and inter-task similarity. According to the LCB values, several solutions with lower LCB values are selected and transferred to the target task.	124
6.3	Convergence traces of average objective values obtained by MFEA, EMT-GFK, EMT-EA, GMFEA, AT-MFEA, MFEA-KAE, and LCB-EMT on the F series single-objective multitasking instances (F1~F4).	135
6.4	Convergence traces of average objective values obtained by MFEA, EMT-GFK, EMT-EA, GMFEA, AT-MFEA, MFEA-KAE, and LCB-EMT on the F series single-objective multitasking instances (F5-F8).	136
6.5	Convergence traces of IGD values obtained by MFEA, LCB-MFEA, MFEA-II, LCB-MFEA-II, DAE-MFEA, and LCB-DAE-MFEA on task 1 of multiobjective multitasking instances.	137
6.6	Convergence traces of IGD values obtained by MFEA, LCB-MFEA, MFEA-II, LCB-MFEA-II, DAE-MFEA, and LCB-DAE-MFEA on task 2 of multiobjective multitasking instances.	139

6.7	Inter-task similarity λ estimated by the proposed LCB-EMT. λ_{gen}^1 and λ_{gen}^2 denote the estimated similarity on the first task and the second task under the generation index gen , respectively, and mean $\lambda = \frac{\sum_{gen}(\lambda_{gen}^1 + \lambda_{gen}^2)}{2 \sum_{gen} 1}$	142
6.8	The average number of exploratory solutions and exploitatory solutions on F1~F4 instances when the knowledge transfer is triggered.	143
7.1	(a) An example of diagnostic explanations and actionable explanations on graphs. (b) Biased subgraphs as diagnostic explanations are identified by selecting a subset of edges and nodes to locate the root cause of the biased behavior of a GNN model. (c) By repairing the biased subgraphs (both features and structures), counterfactual debiased subgraphs as actionable explanations are generated to help the system developers and machine learning practitioners to reduce the data bias, thus rectifying discriminatory behavior.	153
7.2	Two stages in the proposed <i>Geb</i> : The top branch (stage one) aims to identify biased subgraphs as diagnostic explanations for locating the root of data bias. The bottom branch (stage two) aims to generate counterfactual debiased subgraphs as actionable explanations for mitigating the bias.	157
7.3	The illustrative examples of calculating the influence scores. The red line represents the newly selected edge in the subgraph, while the dashed line represents the edge that is removed from the subgraph. The number below the node denotes the influence score.	161
7.4	The obtained Pareto front on NBA, Pokec-z, and Pokec-n datasets, where the lower-right corner is desirable.	169
7.5	Visualization of fairness performance on NBA, Pokec-z, and Pokec-n datasets. The lower left corner, the fairer of the GNN model.	169

7.6	Explanation visualization on NBA dataset, where different node's color denotes the nodes with different sensitive attributes. The solid black line, dashed line, and solid red line denote the raw edge, the selected edge in the biased subgraph, and the added edge in the counterfactual debiased subgraph, respectively. (a) The identified biased subgraph of the first 20 nodes. (b) The counterfactual debiased subgraph is generated based on the biased subgraph in (a). (c) The identified biased subgraph of the 70-th~ 90-th nodes. (d) The counterfactual debiased subgraph is generated based on the biased subgraph in (c). The explanation visualization indicates that nodes with the same sensitive attributes could amplify biases and breaking such connections can suppress the amplification of bias.	171
7.7	The ratio of each type of edge to all edges, where the actionable explanations produced by <i>Geb</i> -GCN is used to mitigate bias.	172
7.8	The change of the fairness performance and the AUC scores obtained by GCN when randomly flipping positive edges to negative edges with varied ratios. . .	172
7.9	The impact of different crossover rates and mutation rates of <i>Geb</i> -GCN on AUC performance under NBA, Pokec-z, and Pokec-n datasets.	175
7.10	The impact of different crossover rates and mutation rates of <i>Geb</i> -GCN on Δ_{sp} values under NBA, Pokec-z, and Pokec-n datasets. A mutation rate that is too high and a crossover rate that is too low will degrade the ability to search for counterfactual debiased subgraphs.	176
7.11	The convergence traces of the proposed EA on NBA, Pokec-z, and Pokec-n datasets.	176
7.12	Analysis of varying population sizes of <i>Geb</i> -GCN on NBA, Pokec-z, and Pokec-n datasets, which indicates a small population size can lead to the failure of searching the promising solution traits.	177

List of Tables

3.1	Mean and variance of IGD values obtained by compared algorithms over LSMOPs.	38
3.2	Mean and variance of SP values obtained by compared algorithms over LSMOPs.	39
3.3	Mean and variance of IGD values obtained by different basic solvers.	40
3.4	Mean and variance of IGD values obtained by compared algorithms over LSMOPs when the number of decision variables is 500.	41
3.5	Mean and variance of IGD values obtained by different ϵ and k values of GAN- NSGA2.	44
3.6	Mean and variance of IGD values obtained by compared algorithms over LSMOPs with various numbers of objective functions.	46
4.1	Mean and variance of MIGD values obtained by compared algorithms over DF test functions.	65
4.2	Mean and variance of MHV values obtained by compared algorithms over DF test functions	66
4.3	Mean and variance of MIGD values obtained by different ablation algorithms over DF test functions.	72

4.4	Mean and variance of MHV values obtained by different ablation algorithms over DF test functions.	73
4.5	Mean of runtime obtained by compared algorithms at $\tau_t=10$ and $\tau_t=5$. (in seconds)	75
5.1	Characteristics of the proposed DCF test functions.	91
5.2	Mean and standard deviation of MIGD values obtained by compared algorithms over DCF test functions.	107
5.3	Mean and standard deviation of MHV values obtained by compared algorithms over DCF test functions.	108
5.4	Mean and standard deviation of MIGD and MHV values obtained by ablation algorithms over DCF test functions.	112
5.5	Five conditions for raw ore allocation of mineral processing.	114
5.6	Raw ore properties.	115
5.7	Mean and standard deviation of HV values obtained by competing algorithms on the ROA problem.	115
6.1	Mean objective values and standard deviation obtained by compared algorithms on the classical single-objective multitask problems.	133
6.2	Mean objective values and standard deviation obtained by compared algorithms on the F series single-objective multitask problems.	134
6.3	Mean IGD values and standard deviation obtained by ablation algorithms on the multiobjective multitask problems.	138
6.4	Mean objective values and standard deviation obtained by ablation algorithms on the F series single-objective multitask problems.	140

6.5	Mean objective values and standard deviation obtained by ablation algorithms on the classical single-objective multitask problems.	141
6.6	Mean objective values and standard deviation obtained by ablation algorithms on the F series single-objective multitask problems.	146
6.7	Mean objective values and standard deviation obtained by ablation algorithms on the classical single-objective multitask problems.	147
6.8	Five conditions for raw ore allocation of mineral processing.	148
6.9	Mean HV values and standard deviation obtained by different algorithms on the ROA problems.	149
7.1	The comparisons of our proposed <i>Geb</i> with the baselines (%).	168
7.2	The compactness values of the explanations generated by <i>Geb</i> , GEAR, and NIFTY, respectively.	170
7.3	The comparisons of our proposed <i>Geb</i> with its ablation versions (%)	173
7.4	The running time of the compared algorithms on NBA, Pokec-z, and Pokec-n datasets (seconds).	174

Chapter 1

Introduction

1.1 Background and Motivation

Multiobjective optimization problems (MOPs) are optimization problems with conflicting objective functions, which are widely present in a rich body of real-world applications, such as vehicle routing [4, 5], social media analysis [6, 7], beneficiation process [8, 9], physical field simulation [10], etc. Due to the conflicting nature of the objectives, solving an MOP typically yields a set of solutions that represent the optimal trade-offs among the conflicting objectives. These solutions are known as the Pareto optimal set (POS) of an MOP, while the mapping of POS in the objective space is referred to as the Pareto optimal front (POF) of an MOP.

Although a wide variety of mathematical optimization techniques have been developed to solve MOPs, they typically reformulated the objective functions as a single-objective problem. Instead of POS, only one single solution could be found in one algorithmic execution. In addition, they often require domain-specific information or mathematical properties of the problems, such as continuous, differentiable, etc. These limitations have motivated the use of a class of heuristic algorithms — evolutionary algorithms (EAs). Inspired by biological evolution mechanisms, EAs

iterate through a population of candidate solutions, using mechanisms like selection, mutation, crossover, and evaluation to navigate the search space and improve solutions over the evolution process. Through iteratively evolving populations, EAs can find a set of approximately Pareto optimal solutions in one go. The evolution process can be briefly summarized as the following steps:

1. Initialization: Creating an initial population of candidate solutions to start the optimization process.
2. Reproduction: Creating new candidate solutions by combining genetic information via crossover and mutation from the parent solutions to explore the search space.
3. Evaluation: Evaluating the quality of each candidate solution by calculating the objective functions.
4. Selection: Choosing candidate solutions from the current population based on their objective values to move on to the next generation.
5. Termination: Determining when to stop the algorithm, usually based on reaching a specified number of generations or achieving a desired solution quality.

Despite the great success enjoyed by EAs in various fields, more and more complex MOPs of large-scale, dynamic, and multitasking nature pose a series of challenges to EAs. Specifically, according to the characteristics of different kinds of complex MOPs, traditional EAs face the following challenges.

Large-scale MOPs (LSMOPs): LSMOPs typically involve a large number of decision variables, leading to a vast search space. The problem's dimensionality increases exponentially with the number of variables, making the exploration of solutions more challenging. In particular, due to the inability to learn knowledge from the solution space, traditional EAs struggle to ef-

fectively search this high-dimensional space. Consequently, traditional EAs could prematurely converge to a suboptimal region without fully exploring the potential search space.

Dynamic MOPs (DMOPs): DMOPs are optimization problems with time-varying, conflicting objective functions. The difficulty in solving DMOPs lies in efficiently tracking the varying POS and POF. Traditional EAs fall short of leveraging the knowledge of dynamic environments to accelerate the search, thus leading to inefficient search.

Dynamic Constrained MOPs (DCMOPs): DCMOPs are characterized as conflicting optimization objectives and constraints that change throughout the optimization. Due to the simultaneous presence of constraints and dynamism, not only should Pareto optimal solutions be rapidly tracked, but the solutions found must also be feasible. Traditional EAs with constraint-handling techniques could force the population to search towards some feasible areas, damaging diversity and leading to slow convergence.

Multitask MOPs: As MOPs seldom exist in isolation in real-world applications, multitask MOPs require concurrently conducting the evolutionary search on multiple optimization tasks. Traditional EAs can only solve tasks one by one instead of learning the synergetic knowledge among multiple optimization tasks, resulting in low solving efficiency.

In short, traditional EAs typically start the search from a “ground zero” knowledge state, resulting in inefficient search when solving the above complex MOPs [11]. In fact, optimization knowledge and search experience during the evolution process could enhance problem-solving performance. Therefore, to tackle increasingly complex MOPs effectively and efficiently, leveraging traits of problem-solving experiences to develop knowledge-driven EAs is deemed desirable and much expected. Built upon this intuition, new evolution paradigms, such as evolutionary transfer optimization (ETO) [11] and learnable EAs [12], have attracted extensive attention in the field of evolutionary computation. A general guideline for designing knowledge-driven EAs is to reuse the optimization experiences or knowledge, e.g., solutions, for efficient problem-solving [11, 12]. Following this guideline, this thesis conducts research on knowledge-driven

EAs for solving complex MOPs such as LSMOPs, DMOPs, DCMOPs, and multitask MOPs. Specifically, the research objectives of the thesis are as follows.

1. Designing a knowledge-driven offspring generation operator to generate high-quality offspring solutions for LSMOPs.
2. Devising a knowledge-driven computational resource allocation strategy to exploit the limited computational resource in dynamic environments for DMOPs.
3. Proposing a knowledge-driven spatial-temporal knowledge transfer-based dynamic response strategy to accelerate the search for feasible solutions for DCMOPs.
4. Crafting a knowledge-driven solution selection method to promote positive knowledge transfer for multitask MOPs.
5. Solving multiobjective optimization problems in fair graph learning and leveraging domain-specific knowledge to design knowledge-driven EAs to ensure graph neural networks (GNNs)' explainability, fidelity, and fairness.

1.2 Contributions

As mentioned above, traditional EAs face many challenges for complex MOPs, such as LSMOPs, DMOPs, DCMOPs, and multitask MOPs. The emerging evolutionary search paradigms, including ETO and learnable EAs, have shown great potential in addressing these complex MOPs. To mitigate the challenges posed by these complex MOPs and further promote the research progress of ETO and learnable EAs, this thesis conducts research on knowledge-driven EAs for solving complex optimization problems such as LSMOPs, DMOPs, DCMOPs, and multitask MOPs. The main contributions of this thesis are as follows:

- In Chapter 3, to solve LSMOPs, the thesis presents a knowledge-driven offspring generation operator by learning the knowledge from high-quality solutions.
 1. Specifically, the thesis points out that Pareto optimal solutions follow a low-dimensional manifold. By learning from the manifold, high-quality offspring solutions distributed on the manifold can be generated by a dedicated generative adversarial network, thereby improving the convergence and diversity of the offspring solutions.
 2. Three interpolation strategies are proposed to manipulate the low-dimensional latent variables to generate more diverse offspring, driving the offspring to escape from the local optima.
 3. A manifold selection mechanism is presented to predict the quality of a generated solution, which reduces computational costs during the evaluation process.
- In Chapter 4, to solve DMOPs, a knowledge-driven computational resource allocation strategy is proposed to exploit the limited computational resources for efficient search in dynamic environments.
 1. Specifically, the thesis argues that existing EAs fall short of leveraging the environmental knowledge to adaptively adjust search strategies in dynamic environments, thus making the computational resources wasted on the suboptimal search regions and improper evolutionary operators.
 2. To reduce the waste of computational resources on suboptimal search regions, an upper confidence bound metric is designed to estimate the potential search benefits of different search regions to maintain a delicate exploration-exploitation trade-off.
 3. To reduce the computational waste on improper evolutionary operators, the evolutionary operator of the activated population can be flexibly changed by the tree policy of Monte Carlo tree search, enabling the evolutionary strategies to adapt to the characteristics of the changing environment.

- In Chapter 5, to solve DCMOPs, the thesis presents a spatial-temporal knowledge transfer-based dynamic response strategy to accelerate the search for feasible solutions in dynamic environments.
 1. Firstly, the thesis shows that existing constraint-handling techniques could force infeasible solutions toward partially feasible regions, leading to the loss of diversity in dynamic environments.
 2. To maintain diversity while keeping feasibility, this thesis establishes two synergistic tasks: one task is focused on the unconstrained search space to maintain diversity, while the other involves searching the constrained search space to focus on feasibility. Particularly, two knowledge transfer modules, i.e., the spatial knowledge transfer module and temporal knowledge transfer module, are designed to reuse the knowledge of the dynamic environment and constrained/unconstrained search space, thus accelerating the search for the two synergistic tasks.
 3. To advance DCMOPs toward real-world cases, a test suite containing 14 test problems is designed with various characteristics, including time-varying feasible and infeasible regions, time-varying geometry of the Pareto front, and time-varying connectivity of feasible areas, etc.
- In Chapter 6, to solve multitask optimization problems, the thesis crafts a knowledge-driven solution selection method for evolutionary multitasking to transfer high-quality solutions for the target optimization problems.
 1. The thesis first claims that existing methods often conduct knowledge transfer by transferring the best solutions from a source task to the target task, while ignoring the proper use of knowledge from the target task in solution selection. As a result, the transferred solutions could not adapt well to the target task, thus limiting the effectiveness of knowledge transfer across tasks.

2. To address the issue, the transfer Gaussian process is employed to model the solution distribution of both the source task and target task, mitigating the negative transfer caused by the lack of exploiting the solution distribution.
 3. An inter-task similarity-based lower confidence bound metric is designed to estimate the potential improvement of the transferred solutions on the target task, enabling the algorithm to identify high-quality solutions to the target task.
- In Chapter 7, towards complex MOPs in fair graph learning, the thesis formulates the fair graph learning problem as a multiobjective optimization problem. A dedicated multiobjective EA is developed to ensure GNNs’ explainability, fidelity, and fairness in high-stake applications.
 1. The thesis attempts to produce compact, diagnostic, and actionable subgraph patterns as explanations that are responsible for the GNN’s biased behavior, and a dedicated multiobjective EA is designed to search such subgraph patterns while maintaining fairness and fidelity.
 2. The thesis proposes an influenced nodes-based gradient approximation for GNNs. The proposed approximation method could avoid retraining the GNNs from scratch during the EA’s evaluation stage.
 3. The thesis provides the theoretical fairness risk bound of the proposed method, indicating that using actionable explanations to rectify the root of bias can theoretically improve fairness.

1.3 Thesis Organization

This thesis comprises eight chapters. Chapter 1 sets the stage by introducing this thesis’s background, motivations, and primary contributions. Chapter 2 offers an overview of the relevant

literature on complex MOPs. Chapter 3 delves into my research on knowledge-driven large-scale multiobjective optimization. Chapter 4 presents my work on knowledge-driven dynamic multiobjective optimization. Chapter 5 showcases my research on knowledge-driven dynamic constrained multiobjective optimization. In Chapter 6, my work on knowledge-driven evolutionary multitask optimization is presented. Chapter 7 provides research on knowledge-driven evolutionary optimization for fair graph learning. Finally, Chapter 8 consolidates the contributions of this thesis and offers future directions.

Chapter 2

Preliminaries on Multiobjective Optimization Problems

In this chapter, the thesis presents fundamental concepts and related work on complex MOPs studied in this thesis, including large-scale multiobjective optimization, dynamic multiobjective optimization, dynamic constrained multiobjective optimization, and multitask optimization.

2.1 Large-Scale Multiobjective Optimization

Concepts. LSMOPs typically involve a large number of decision variables, leading to an extremely large search space. The dimensionality of the problem scales exponentially with the number of variables, making it increasingly challenging to explore and evaluate potential solutions efficiently. The mathematical form of LSMOPs is as follows:

$$\begin{cases} \min F(x) = \langle f_1(x), f_2(x), \dots, f_M(x) \rangle \\ s.t. x \in \Omega. \end{cases} \quad (2.1)$$

where $x = (x_1, x_2, \dots, x_N)$ is the N -dimensional decision vector, and $F = (f_1, f_2, \dots, f_M)$ is the M -dimensional objective vector. It should be noted that the dimension of the decision vector N is greater than 100 [13, 14]. The goal of large-scale multiobjective optimization algorithms is to find the solutions for which all objectives are as small as possible. However, conflicts exist between multiple objectives, and one solution cannot satisfy the minimum of all objectives. Therefore, a trade-off method called Pareto dominance is introduced to compare these solutions. The set of optimal trade-off solutions is called the POS in the decision space and the POF in the objective space.

Definition 1. (*Decision Vector Domination*) A decision vector x_2 Pareto-dominates another vector x_1 denoted by $x_2 \succ x_1$, if and only if

$$\begin{cases} f_i(x_2) \leq f_i(x_1), & \forall i = 1, \dots, M \\ f_i(x_2) < f_i(x_1), & \exists i = 1, \dots, M. \end{cases} \quad (2.2)$$

Definition 2. (*Pareto Optimal Set, POS*) If a decision vector x^* satisfies

$$POS = \{x^* | \nexists x, x \succ x^*\}, \quad (2.3)$$

then all x^* are called Pareto-optimal solutions, and the set of Pareto-optimal solutions is called the POS.

Definition 3. (*Pareto Optimal Front, POF*) POF is the corresponding objective vector of the POS

$$POF = \{y^* | y^* = F(x^*), x^* \in POS\}.$$

Related Work. The existing large-scale multiobjective optimization algorithms (LSMOAs) can be roughly grouped into the following three categories [15], including decision variable grouping-based LSMOAs, decision space reduction-based LSMOAs, and novel search strategy-based LSMOAs.

Decision variable grouping-based LSMOAs divide the decision variables into several groups and then optimize each group of decision variables successively. Antonio *et al.* [16] maintained several independent subpopulations. Each subpopulation is a subset of the equal-length decision variables obtained by variable grouping (e.g., random grouping, linear grouping, ordered grouping, or differential grouping). All of the subpopulations work cooperatively to optimize the LSMOPs in a divide-and-conquer manner.

Zhang *et al.* [17] proposed a decision variable clustering-based large-scale evolutionary algorithm (LMEA). In LMEA, the decision variables are divided into two types using a clustering method. Then, for diversity-related variables and convergence-related variables, two different strategies are implemented to improve convergence and diversity.

Ma *et al.* [18] presented a multiobjective EA based on decision variable analysis (MOEA/DVA). The LSMOP is decomposed into a number of simpler subproblems that are regarded as independent subpopulations to optimize. The disadvantage of decision variable analysis-based LSMOAs is that they may incorrectly identify the linkages between decision variables to update solutions, which may lead to local optima [17]. However, decision variable grouping methods easily lead to excessive computational complexity due to the large-scale decision variables [19], and two related variables may be divided into different groups, which may lead to local optima [20].

Decision space reduction-based LSMOAs aim to convert large-scale problems to small-scale problems by dimensionality reduction techniques, and then conventional EAs solve the transformed problems. He *et al.* [21] introduced a generic framework called the large-scale multi-objective optimization framework (LSMOF). LSMOF reformulates the original LSMOP into a low-dimensional single-objective optimization problem with some direction vectors and weight variables, which aims at guiding the population towards the POS.

Zille *et al.* [22] proposed a weighted optimization framework (WOF) for solving LSMOPs. Decision variables are divided into many groups, and each group is assigned a weight variable. Then, in the same group, the weight variables are regarded as a subproblem of a subspace of the

original decision space. However, due to the information loss of original search space caused by dimensionality reduction, the transformed problem is likely to lose the global optimal solutions.

Novel search strategy-based LSMOAs design reproduction operators and probability models for solving the LSMOPs directly. Tian *et al.* [14] used a competitive swarm optimizer to solve LSMOPs. The proposed LMOCSO suggests a two-stage strategy to update the positions of particles: a competitive mechanism is adopted to determine the particles to be updated, and the proposed updating strategy is used to update each particle.

GMOEA [23] used a GAN to generate offspring for solving multiobjective optimization problems. IM-MOEA [24] employed the Gaussian process model to map the solutions from the objective space to the decision space, and generates offspring. SparseEA [25] designed a new crossover operator and a new mutation operator for better searching the sparse optimal solutions by controlling the sparsity of the offspring. A more comprehensive review of LSMOAs and LSMOPs can be referenced in this reference [15].

2.2 Dynamic Multiobjective Optimization

Concepts. DMOPs refer to optimization problems with multiple conflicting objectives changing over time. Without loss of generality, the mathematical form of a DMOP can be defined as follows:

$$\begin{aligned} \min F(x, t) &= (f_1(x, t), f_2(x, t), \dots, f_M(x, t)), \\ \text{s.t. } x &\in \Omega, \end{aligned} \tag{2.4}$$

where $x = (x_1, x_2, \dots, x_N)$ is a vector of decision variables, N is the dimension of x , Ω is the decision space, $(f_1, f_2, \dots, f_M)$ are objective functions, M is the number of objective functions, and t is a discrete-time or environment step. Generally, the environment step t involved in the DMOPs is defined as,

$$t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor, \tag{2.5}$$

where τ , n_t , and τ_t refer to the maximum number of generations, the severity of the change, and the frequency of change (i.e., the number of generations for each environment step), respectively. A higher τ_t value denotes a slower changing environment, while a higher n_t value means a more dramatically changing environment.

In MOPs, Pareto dominance is used to define the quality of solutions. A similar concept also exists in DMOPs. A set of solutions is defined as a dynamic Pareto optimal set if the decision vector of this solution set cannot be further optimized in any dimension at a given environment t . Similarly, the following concepts have been extended to the field of DMOPs.

Definition 4. (*Dynamic Decision Vector Domination*) For minimizing a DMOP, a decision vector x_2 Pareto dominates another vector x_1 at given environment step t , denoted by $x_2 \succ_t x_1$, if and only if:

$$\begin{cases} f_i(x_2, t) \leq f_i(x_1, t) & \forall i = 1, \dots, M, \\ f_i(x_2, t) < f_i(x_1, t) & \exists i = 1, \dots, M. \end{cases} \quad (2.6)$$

Definition 5. (*Dynamic Pareto Optimal Set, DPOS*) Assuming x and x^* are two decision vectors, and if a decision vector x^* is non-dominated at environment step t , if and only if no other decision vector x exists such that $x \succ_t x^*$ at environment step t . The dynamic Pareto optimal set is the set of all Pareto optimal solutions at time t ,

$$DPOS(t) = \{x^* \mid \nexists x, x \succ_t x^*\}.$$

Definition 6. (*Dynamic Pareto Optimal Front, DPOF*) At environment step t , the dynamic Pareto optimal front is the corresponding vector of objective functions of the DPOS.

$$DPOF(t) = \{F(x, t) \mid x \in DPOS(t)\}.$$

Related Work. In recent years, various dynamic multiobjective optimization evolutionary algorithms (DMOEAs) have been made for rapidly tracking DPOS. Most existing DMOEAs can

be roughly grouped into the following three main categories: diversity enhancement/maintenance methods, memory-based methods, and prediction-based methods.

To avoid the local optima caused by the loss of diversity when the environment changes, diversity enhancement/maintenance methods introduce newly generated solutions to improve the diversity of the population [26, 3, 27, 28]. DNSGA-II [3] is a pioneering paradigm of diversity enhancement and maintenance methods. In DNSGA-II, partial solutions are replaced by some solutions with randomly created solutions or mutated solutions. Yazdani *et al.* [29] used a reaction method in which the diversity is increased at the beginning of each environment. When a change is detected, other solutions are randomized around trackers' positions. Chen *et al.* [30] simultaneously maintained two subpopulations. One subpopulation focuses on providing competitive selection pressure toward the DPOS, while the other subpopulation maintains a promising diversity.

Liang *et al.* [31] designed a multi-strategy to accelerate convergence while maintaining good diversity. Jiang *et al.* [26] designed a strategy based on steady-state and population diversity to maintain the population diversity. Rong *et al.* [32] presented a cluster strategy for identifying representative solutions describing the diversity of a POS at different environment steps.

Memory-based methods enable EAs to explicitly or implicitly record useful information from past consecutive environments to guide the future search direction [33, 34, 35, 36]. Branke *et al.* [33] designed a memory scheme where the best-found solutions are stored in an additional archive. The saving solutions are retrieved from the archive when the change is detected. Cao *et al.* [37] recorded past solutions obtained in sequential time periods and used a support vector regression to predict the future locations of solutions according to the last traces of solutions.

Similarly, MMTL [35] constructed the geodesic flow with the solutions from sequential time periods and predicted the location of the solutions in the new environment. Xu *et al.* [38] proposed a memory-enhanced dynamic multiobjective evolutionary algorithm based on decomposition, in which a subproblem-based memory scheme can reuse solutions from old environments.

Azzouz *et al.* [39] presented an adaptive memory management strategy that can adjust the number of memories according to the severity of the change. Recently, Xu *et al.* [36] proposed a novel memory mechanism that can implicitly save the information of the solutions in past environments, where the historical information is archived in an online support vector model.

In general, the memory-based mechanism is more suitable for periodic optima problems. If the DMOP is nonperiodic, the solutions kept in the additional storage will be out of the vicinity of the new optimal solution.

Inspired by the powerful ability of machine learning, the prediction-based methods typically construct a machine learning model to predict the state of solutions to accommodate the changing environment in advance [40, 41, 42, 43, 44, 45, 46, 47, 48, 49]. For example, Muruganantham *et al.* [50] employed the Kalman filter as a predictive model to generate the initial population. Feng *et al.* [51] proposed to solve DMOPs via an autoencoder that predicts the moving of the Pareto-optimal solutions based on the nondominated solutions obtained before the dynamic occurs. Zou *et al.* [52] proposed a surrogate model based on inverse reinforcement learning, which applies the max-margin method to approximate the reward function and helps accelerate the convergence of the algorithm.

TK-DMOEA [53] used a set of weak classifiers to predict the knee points in the new environment, which enabled the search process to benefit from a limited number of high-quality solutions. Zhao *et al.* [54] proposed a knowledge-learning strategy to react to changes via learning from the historical search process. In particular, the extracted knowledge can accelerate convergence and introduce diversity for the optimization of the future environment.

Owing to the strong representation ability of neural networks, some works proposed the use of neural networks to assist DMOEAs [47, 52, 55]. For example, Wang *et al.* [5] proposed a two-branch neural network to extract the knowledge from the source task and the target task, where the knowledge from the source task is adaptively reused by the neural network to accelerate the search of the target domain.

ETO-based DMOEAs have emerged as a promising branch of prediction-based methods in recent years, owing to their potential to leverage the knowledge from the historical search process [56, 11, 46, 48, 57, 54, 58, 59]. Specifically, the distribution of the solutions varies due to changes in the environment. Such non-independently identically distributed data challenges the performance of traditional prediction-based DMOEAs. To handle non-independently identically distributed data, ETO-based DMOEAs leverage the transfer learning to reuse the knowledge gained from the historical search process and adapt knowledge to the new environment, thus accelerating the search in the new environment.

2.3 Dynamic Constrained Multiobjective Optimization

Concepts. DCMOPs refer to a type of optimization problem with multiple conflicting objectives and constraints changing over time. Without loss of generality, the mathematical formulation of a DCMOP can be defined as follows:

$$\begin{aligned}
 & \min F(x, t) = (f_1(x, t), \dots, f_m(x, t), \dots, f_M(x, t)) \\
 & s.t. \begin{cases} H(x, t) = (h_1(x, t), \dots, h_k(x, t), \dots, h_{N_H}(x, t)) \\ G(x, t) = (g_1(x, t), \dots, g_k(x, t), \dots, g_{N_G}(x, t)) \end{cases} \\
 & h_k(x, t) = 0, k = 1, \dots, N_H \\
 & g_k(x, t) \leq 0, k = 1, \dots, N_G,
 \end{aligned} \tag{2.7}$$

where $x = (x_1, x_2, \dots, x_N) \in R^N$ is a vector of decision variables, N is the dimension of x . $(f_1, f_2, \dots, f_M)$ are objective functions, and M is the number of objective functions. $H(x, t)$ and $G(x, t)$ represent the equality and inequality constraint vectors, respectively, and t is a discrete-time or environment step. If a solution satisfies all constraints, then it is called a feasible solution; otherwise, it is called an infeasible solution.

Definition 7. (*Dynamic Constrained Domination*) At time t , for two feasible solutions

x_1 and x_2 , x_1 Pareto-dominates x_2 denoted by $x_1 \succ_t x_2$, if and only if,

$$\begin{cases} f_m(x_1, t) \leq f_m(x_2, t), & \forall m = 1, \dots, M, \\ f_m(x_1, t) < f_m(x_2, t), & \exists m = 1, \dots, M. \end{cases} \quad (2.8)$$

Based on the existing definitions of dynamic Pareto optimal set and dynamic Pareto optimal front [46], this thesis extends the definitions to the context of DCMOPs.

Definition 8. (Dynamic Constrained Pareto Optimal Set, DCPOS) If a feasible solutions x^* at time t satisfies,

$$DCPOS = \{x^* | \nexists x, x \succ_t x^*\}, \quad (2.9)$$

then all x^* are called dynamic-constrained Pareto optimal solutions, and the set of dynamic-constrained Pareto optimal solutions is called the dynamic-constrained Pareto optimal set, which is distinguished from the dynamic unconstrained Pareto optimal set (DUPOS) without considering the constraints.

Definition 9. (Dynamic Constrained Pareto Optimal Front, DCPOF) DCPOF is the corresponding objective vectors of the DCPOS at time t ,

$$DCPOF = \{y^* | y^* = F(x^*, t), x^* \in DCPOS\},$$

which is distinguished from the dynamic unconstrained Pareto optimal front (DUPOF) without considering the constraints.

Related Work. DCMOPs are at the frontier of DMOPs and static constrained MOPs (CMOPs) research fields. This section will provide a brief overview of the recent literature on CMOPs. Afterward, the section will review the existing works specifically focused on DCMOPs.

To handle constraints in CMOPs, various efforts have been made, and these approaches can be classified as the following techniques: constraint-domination principle (CDP), penalty functions, stochastic ranking, ϵ -constrained methods, constraint transformation methods, and hybrid techniques.

CDP is used during environmental selection for selecting solutions from the population pool [60, 61, 62]. In CDP, feasible solutions are considered better than infeasible solutions. For infeasible solutions, the dominance between infeasible solutions depends on constraint violation rather than objective values. Penalty function-based techniques weigh the constraints as penalty terms for the objective functions. The penalty weight is fixed or can be adaptively changed during the search progress [63, 64, 65, 66]. For example, Ma *et al.* [67] devised an adaptive penalty function, called ShiP, to help promise infeasible solutions to approach the feasible region from different directions. Stochastic ranking compares two solutions according to their constraint violations or objective values. The comparison criterion depends on a user-defined probability [68, 69]. For example, Tian *et al.* [70] determined the dominance relation by the evaluation of objectives and constraints. Based on the ratio of feasible solutions, different fitness evaluation processes are conducted for environmental selection.

The ϵ -constrained method regards the infeasible solutions in the relaxed constraint boundary as feasible solutions. The relaxation of the constraints is controlled by the parameter ϵ . When ϵ reduces to zero, the ϵ -constrained method degenerates to CDP [71, 72]. Recently, Qiao *et al.* [73] designed an improved ϵ -constrained method under the multitasking framework, in which diverse high-quality infeasible solutions are kept to help the solutions of the main task breakthrough infeasible obstacles in the early stage. Constraint transformation methods regard the constraint violation value as an additional objective, and the nondominated sorting method can be used to rank the solutions [74, 75]. The hybrid technique is a mix of the above constraint handling schemes [76, 77, 78].

In recent studies, a handful of evolutionary multitasking-based constrained multiobjective opti-

mization algorithms have emerged with a focus on leveraging the potential synergy of infeasible solutions to improve the convergence of feasible solutions [73, 2]. These algorithms introduce an auxiliary task with simplified constraints, aiming to explore infeasible regions and supplement diversity to the original CMOP. However, their approaches lack elaborated dynamic response strategies to effectively utilize the knowledge across feasible and infeasible regions in consecutive environments. As a result, they may introduce or retain low-quality solutions that impede the convergence in newly generated environments.

Although the above constraint-handling techniques developed for solving CMOPs can be extended to address DCMOPs, they suffer from a selection bias. When constraints are given a higher priority than objectives, it can lead to the degradation of population diversity. Consequently, when the environment changes, the population may become trapped in local regions. Conversely, if a higher priority is given to objectives, it may lead to a decrease in convergence quality. This contradicts the goal of efficiently obtaining Pareto optimal solutions in dynamic environments. Several studies have also confirmed that these constraint-handling techniques exhibit distinct deficiencies when applied to different CMOPs and dynamic constrained optimization problems. Detailed discussions on these techniques for solving CMOPs and dynamic constrained optimization problems can be found in [79] and [80], respectively.

Although there is a great amount of work regarding DMOPs and CMOPs, little attention has been given to solving DCMOPs. Chen *et al.* [81] devised a tribe partitioning approach that divides the whole population into separate tribes and applies distinct strategies to drive them toward the feasible regions. However, the dynamic strategy employed in this method, using a one-dimensional search strategy, requires additional function evaluations. In another study, Chen *et al.* [82] introduced a dynamic constrained multiobjective evolutionary algorithm called SKTEA, which utilizes a subspace-knowledge transfer strategy for solving DCMOPs. However, the effectiveness of the knowledge transfer heavily depends on the availability of feasible subspaces. In cases where DCMOPs have large infeasible regions, SKTEA could produce an initial

population leading to negative transfer.

Azzouz *et al.* [65] employed the dynamic NSGA2 as the basic solver and designed an adaptive penalty function to guide feasible regions. However, this approach faces difficulties when dealing with problems that have some infeasible regions that need to be traveled. To address the above issue, Azzouz *et al.* [64] designed an improved version of the dynamic NSGA2, called DC-MOEA. The proposed DC-MOEA utilizes a self-adaptive penalty function to make use of infeasible solutions with better objective values, and a feasibility-driven strategy is designed to guide the search toward the new feasible regions. Similarly, Chen *et al.* [66] proposed dCMOEA, which also uses an adaptive penalty function to balance feasible and infeasible solutions to avoid premature convergence. To alleviate the loss of diversity caused by the changing environment, the above works utilized memory schemes to reinitialize the partial solutions for the new environment.

However, the above-designed adaptive penalty mechanisms may lead to the loss of diversity and premature convergence when faced with discontinuous feasible regions. Without loss of generality, the objective of the above-designed adaptive penalty function can be represented as

$$f'(x, t) = f(x, t) + p_f CV(x, t), \quad (2.10)$$

where p_f is the adaptive penalty factor and $CV(x, t)$ is the constraint violation. When most of the solutions in the population are feasible solutions, p_f will be close to zero, and the modified objective functions will degenerate to the original objective functions. In this case, a feasible solution can hardly convert to an infeasible solution to explore other disconnected feasible regions. Such damage to the diversity of infeasible solutions could result in premature convergence.

2.4 Multitask Multiobjective Optimization

Concepts. Multitask multiobjective optimization aims to optimize multiple tasks simultaneously. Each task is supposed to be an MOP $F(x)$, and multitask multiobjective optimization can be formulated as follows:

$$\begin{aligned} & \min(F_1(x), F_2(x), \dots, F_K(x), \\ & s.t. \quad x \in \Omega, \end{aligned} \tag{2.11}$$

where K is the number of tasks that need to be optimized.

Related Work. existing multitask optimization algorithms can be generally classified into two categories [83]: implicit transfer and explicit transfer. Compared with multitask optimization with implicit transfer, multitask optimization with explicit transfer utilizes additional mechanisms to transfer optimized solutions among tasks, thus better suppressing the negative transfer. Existing multitask optimization algorithms with explicit transfer can be categorized into three groups [84]: 1) distribution-based approaches, 2) matching-based approaches, and 3) hybrid approaches.

Distribution-based approaches are designed to leverage explicit distribution information to mitigate negative transfer effects that arise due to differences in optimal locations or population distributions across tasks [84]. GMFEA [85] aims to transfer the best solutions into the uniform searching space in which decision variable translation and decision variable shuffling strategies were introduced to overcome the heterogeneity of distinct problems. EMT-EA [86] introduced two separate solvers with different operators to solve two tasks with distinct search spaces, and a denoising autoencoder is employed to transfer solutions across the two tasks. MFEA-II [87] is an improved MFEA with an adaptive intertask transfer probability, in which an online learning approach is employed for adjusting the factor rmf of assortative mating to alleviate the threat of negative transfer. MTGA [88] transferred the high-quality solution from the source task

to the target task by multiplying the sample mean of the target task. To transfer knowledge among different tasks with unaligned dimensions, BLKT [89] divided individuals into multiple blocks where each block corresponds to several consecutive dimensions, then the knowledge is transferred among blocks with similar dimensions.

Matching-based approaches aim to address negative transfer effects resulting from differences in landscape similarities between tasks. To elaborate, such methods mainly convert the source task into a new task that shares a high-rank correlation with the target task. In this way, transferable solutions across tasks can be generated, thus mitigating negative transfer. LDA-MFEA [90] used a domain adaptation technique for solving heterogeneous multitasking optimization problems, where a linear mapping between tasks is derived by the ordinal correlation of tasks. Chen *et al.* [91] measured the similarity between tasks by the Kullback-Leibler divergence, and nonlinear mapping or linear mapping is automatically selected for transforming solutions according to the similarity. Hashimot *et al.* [92] computed the distance of Gaussian distribution to estimate the inter-task similarity, and then solutions from the task with higher similarity are selected for knowledge transfer. Jiang *et al.* [93] measured two different types of inter-task similarity, i.e., shape similarity and task similarity. Based on the type of inter-task similarity, knowledge transfer strategies are adaptively selected for knowledge transfer. To handle the heterogeneity of search spaces of multiple tasks, Wu *et al.* [94] designed an orthogonal transfer method to infer the similarity of different dimensions, thus transferring the most helpful knowledge to the target task. By leveraging the inherent similarities between constrained Pareto optimal fronts and unconstrained Pareto optimal fronts, Qiao *et al.* [2, 95] decomposed constrained multiobjective optimization problems into multiple tasks with respect to constrained Pareto optimal fronts and unconstrained Pareto optimal fronts, and the knowledge is transferred among the complementary tasks. In addition, a series of studies designed various space alignment methods for minimizing the discrepancies between tasks to transfer solutions [96, 97, 90, 98].

Although matching-based approaches can effectively address negative transfer, they may po-

tentially overlook distribution information. To overcome this issue, hybrid transfer learning approaches have been proposed by embedding the distribution information into the matching-based transfer process. Some works match ranked individuals in different domains by employing subspace alignment [99, 100]. In particular, the sampled data in a task first undergoes dimension reduction by principle component analysis, which can be better than directly matching ranked individuals. Chen *et al.* [101] classified the individuals into different levels according to their performance, and the data distribution is aggregated into a common space to separate the projections from the different tasks, thereby preserving semantic features for each task individually. Furthermore, as many real-world applications suffer noisy or biased information [84, 97, 102, 6, 103], a number of works are proposed for robust multitasking optimization. To prevent the chaotic matching caused by the noisy information, AT-MEFA [104] prevented the degradation of intertask mapping by a rank loss function to minimize the chaotic mapping gap between the landscapes of tasks. Wu *et al.* [97] designed a robust task with a robust fitness function to help the transferred knowledge to escape local optima.

Among these explicit transfer-based approaches, knowledge transfer is typically achieved by transferring optimized solutions across different tasks. However, based on the assumption that transferring the best solutions from the source task will produce the ‘best’ solutions for the target task, existing methods mainly transfer the best or all solutions of the source task to the target task, but without using information from the target task to elaborately select solutions for knowledge transfer [83, 104, 99, 53, 105, 57]. As a result, some transferred solutions cannot adapt well to the target task, thereby limiting the potential for positive transfer.

Chapter 3

Knowledge-Driven Offspring Generation Operator for Large-Scale Optimization

3.1 Motivation

Many real-world optimization problems involve LSMOPs [106, 107, 108, 109]. For example, in the design of telecommunication networks [110], thousands of variables, such as locations of network nodes, transmission capacity between nodes, and power allocations of nodes, are involved. These numerous decision variables determine the multiple objectives, such as energy consumption, applicability, and stability of the network. It is challenging to find Pareto optimal solutions for these problems because the volume of the search space is exponential to the number of decision variables [111].

Traditional genetic operators encounter low search efficiency when solving LSMOPs [112, 14, 23]. Specifically, the offspring reproduced by traditional genetic operations are often located

near their parents, making the population premature. We take commonly used simulated binary crossover [1] as an example: the similarity between the offspring and their parents can be defined by a factor β , the probability distribution of β is shown in Fig. 3.1. $\beta = 1$ indicates the offspring is the same as the parent. When β moves away from 1, the offspring distribution will be more different away from their parents. From Fig. 3.1, it is clear that offspring will be generated near the parents with a great probability. Although β can be tuned to control the offspring’s position, it is difficult to adjust the hyper-parameter adaptively. In particular, due to the more complicated search space of LSMOPs, it is non-trivial to escape local optima using the traditional operators.

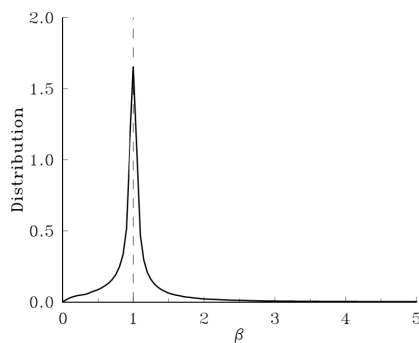


Figure 3.1: The probability distribution of β of simulated binary crossover [1].

To mitigate the issue, research on embedding machine learning techniques into EAs for designing better LSMOAs is attracting increasing attention in the evolutionary computation and machine learning community [48, 113, 114, 115, 116, 117, 118, 119, 120, 121]. For example, Gaussian process [24], covariance matrix [122], and generative adversarial network [23] have been incorporated with LSMOAs for learning the distributions of solutions and generating offspring. In addition, principal component analysis [123] and unsupervised neural networks [124] can be utilized for reducing the dimension of LSMOPs.

Inspired by the ability of machine learning models to learn high-dimensional features, our work utilizes a generative adversarial network (GAN) to learn the characteristics of the decision space of LSMOPs and generate better offspring. To be specific, this work uses GAN to generate

offspring candidates to escape from the local optima and approach the promising search area. There are two reasons for using GAN in this work.

Firstly, POS are distributed on the $m - 1$ -dimensional (m is the number of objectives) manifold [125, 126], and GANs have the ability of learning manifold and modeling manifold. (e.g., Researchers in the field of computer vision use the generative adversarial neural network to learn the manifold of natural images [127, 128]). Therefore, GANs provide a powerful tool for learning high-dimensional solutions and further generating solutions located on the manifold. For example, for a large-scale tri-objective optimization problem, the POS of this problem is distributed on a two-dimensional manifold, and high-dimensional solutions can be generated by mapping two-dimensional variables to high-dimensional search space. In such a way, the POS distribution can be obtained by sampling and mapping a large number of low-dimensional variables, thus reducing the search for high-dimensional decision variables and improving the efficiency of LSMOPs.

Secondly, GANs have the ability to travel the manifold [127, 129, 130]. Many works focused on making the output vary smoothly from one synthesis to another through interpolation method [127, 129, 130]. Compared with randomly sampling low-dimensional variables from the latent space, interpolating the low-dimensional variables from the latent space by linear transformations can create smoother and more meaningful syntheses and better approximate the manifold. By manipulating the low-dimensional latent variables, more solutions can be generated with better uniformity and diversity, helping the population escape from the local optimum. With this ability, we interpolate low-dimensional variables on the manifold to generate solutions on the manifold, and we call this ability manifold interpolation.

This work argues that the integration of GANs into EAs can offer significant benefits for designing better LSMOAs, and a framework called GAN-based large-scale multiobjective evolutionary framework (GAN-LMEF) is proposed. We believe that the key factor for solving LSMOPs is to find an effective way to learn the manifold and generate offspring on this manifold, and

GAN can be utilized as a powerful weapon to solve the issues. The intuitive idea is to learn the manifold and interpolate solutions along the manifold via GANs, as shown in Fig. 3.2. These interpolated offspring solutions fill gaps in the manifold and are uniformly distributed on the manifold, thereby leading the population to draw on the optimal area efficiently. Initially, GAN-LMEF uses non-dominated solutions to construct a series of approximate manifolds. Then, by adopting the proposed GAN-based interpolation method, various solutions are synthesized along manifolds. Next, the population is updated by a manifold selection mechanism. The above procedures are repeated until the termination conditions are met. The contributions of this work are summarized as follows.

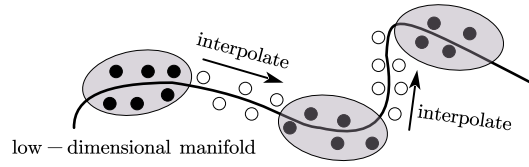


Figure 3.2: Solutions interpolation on the manifold.

1. The proposed framework utilizes a GAN as a powerful weapon to learn the characteristics of high-dimensional solutions and generate solutions along the manifold. These generative solutions are of high quality and can be helpful in the evolutionary process. The framework learns the distribution of high-dimensional solutions by modeling low-dimensional variables, which avoids the search for large-scale space directly. This provides a novel way to solve LSMOPs.
2. Considering that the gaps on the manifold vary with the evolutionary process, three interpolation strategies are proposed to manipulate the low-dimensional latent variables such that more diverse offspring can be generated.
3. In the selection process, this work proposes a manifold selection mechanism to predict the quality of the generated solutions, which can reduce evaluation costs.

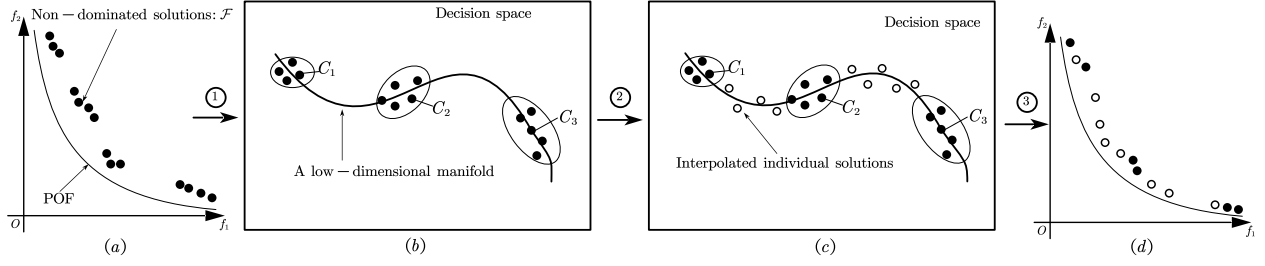


Figure 3.3: The framework of the proposed GAN-LMEF. Step 1. Algorithm 2: Map non-dominated solutions $\mathcal{F}$ into the $m - 1$ -dimensional manifold, cluster them into several clusters, and identify the central solution C_i for each cluster. Step 2. Algorithm 3: A GAN is utilized to interpolate solutions between the central solutions C_i . Step 3. Algorithm 4: Select the excellent solutions from the original population and interpolated solutions to form a new population to guide future searches.

3.2 Proposed Framework

The details of the proposed algorithm, GAN-LMEF, are presented in this section. Briefly, GAN-LMEF consists of three subroutines: Algorithm 2, Algorithm 3, and Algorithm 4. In Fig. 3.3, we give a detailed illustration of the main loop in GAN-LMEF. The main purpose of Algorithm 2 is to map non-dominated solutions into a manifold and find some representative non-dominated solutions on this manifold, called central solutions. In Algorithm 3, a GAN is utilized to interpolate offspring solutions between these central solutions. Algorithm 4 selects promising solutions from the parental population and generative offspring to guide the future search.

The main scheme of the proposed GAN-LMEF is presented in Algorithm 1. The proposed manifold interpolation strategy can only be implemented in the early stage of the search to avoid the premature population. Therefore, We allocate the first $\epsilon \cdot e$ evaluations for the interpolation of solutions, where parameter ϵ controls the number of iterations of interpolation and e is the maximum number of evaluations for solving the problem. When $\epsilon \cdot e$ evaluations

Algorithm 1: GAN-LMEF

Input: N (the population size), σ (the proportion of interpolated solutions), MOA (a multiobjective optimization algorithm), $F(\cdot)$ (the LSMOP), k (the number of clusters), e (the maximum evaluations)

Output: POS (the POS)

```

1 Set parameter  $\epsilon$  for controlling the number of iterations of interpolation;
2 Initialize population  $P$  randomly;
3 while  $Used\ evaluations \leq \epsilon \cdot e$  do
4    $P' = \text{Mating-Selection}(P)$ ;
5    $P = P \cup \text{Variation}(P')$ ;
6    $\mathcal{F} = \text{Fast-Non-Dominated-Sort}(P, F(\cdot))$ ;
7    $C = \text{Computing-Central-Solutions}(\mathcal{F}, k)$ ;
8    $Q = \text{GAN-Manifold-Interpolation}(C, P, k)$ ;
9    $P = \text{Selection}(C, Q, \sigma)$ ;
10 end
11  $POS = \text{MOA}(P, F(\cdot), (1 - \epsilon) \cdot e)$ ;
12 return  $POS$ ;
```

are used, the remaining $(1 - \epsilon) \cdot e$ evaluations are optimized by any population-based MOA. In every iteration of the main loop, the fast non-dominated sort [61] is employed to distinguish the non-dominated solution set $\mathcal{F}$. Next, the non-dominated solution set $\mathcal{F}$ is inputted into Algorithm 2 in which these non-dominated solutions are mapped into the $m - 1$ -dimensional manifold. These mapping solutions are clustered into several clusters, and a central solution C for each cluster is identified. Afterward, in Algorithm 3, a GAN is utilized to interpolate offspring Q between the central solutions C . Finally, Algorithm 4 selects promising solutions P from the parental population and interpolates offspring for the next round.

In the following subsections, this work will introduce the three main components: Algorithm 2, Algorithm 3, and Algorithm 4.

3.2.1 Central Solutions Identification

Central solutions are the representatives on the manifold, and interpolating between these central solutions may yield better solutions. First, to obtain the manifold, the non-dominated

Algorithm 2: Computing Central Solutions

Input: $\mathcal{F}$ (the non-dominated set), k (the number of clusters)**Output:** C (the central solutions)

```
1  $C = \emptyset$ ;  
2 Use PCA with  $\mathcal{F}$  to get the manifold  $\phi(\mathcal{F})$ ;  
3 Cluster  $\phi(\mathcal{F})$  into  $k$  clusters by using  $k$ -means;  
4 for  $i = 1, \dots, k$  do  
5   | Calculate central solution  $C_i$  for each cluster  $\phi(\mathcal{F})_i$  according to Equation (3.1);  
6   |  $C = C \cup C_i$ ;  
7 end  
8 return  $C$ 
```

solutions $\mathcal{F}$ are mapped as the manifold data $\phi(\mathcal{F})$ by principal component analysis (PCA). The mapped data $\phi(\mathcal{F})$ is clustered into k clusters by k -means. Each cluster is denoted as $\phi(\mathcal{F})_i$, $i = 1, \dots, k$, and in each cluster, a central solution is identified to represent the cluster. The central solution C_i for each cluster $\phi(\mathcal{F})_i$ is defined as the solution with the minimum distance from $\phi(C_i)$ to the centroid of the cluster be calculated as

$$C_i = \arg \min_x ||\phi(x) - \overline{\phi(\mathcal{F})_i}||, \quad (3.1)$$

The centroid $\overline{\phi(\mathcal{F})_i}$ of the cluster $\phi(\mathcal{F})_i$ is

$$\overline{\phi(\mathcal{F})_i} = \frac{\sum_{j=1}^{|\phi(\mathcal{F})_i|} \phi(\mathcal{F})_{i,j}}{|\phi(\mathcal{F})_i|}, \quad (3.2)$$

where $\phi(\mathcal{F})_{i,j}$ represents the j -th data in the cluster $\phi(\mathcal{F})_i$.

The central solutions identified and the piecewise manifolds are shown in Fig. 3.3 (b). The details of computing the central solutions are given in Algorithm 2.

3.2.2 Manifold Interpolation

The manifold interpolation will reproduce a number of offspring solutions that are distributed on the manifold and fill gaps in this manifold. These solutions can improve uniformity and diversity and are beneficial to the evolutionary process.

Algorithm 3: GAN Manifold Interpolation**Input:** C (the central solutions), P (the population), $F(\cdot)$ (the LSMOP), k (the number of clusters)**Output:** Q (the interpolation solutions)

```

1  $Q = \emptyset$ ;
2  $\mathcal{F} = \text{Fast-Non-Dominated-Sort}(P, F(\cdot))$ ;
3 Sample noise variables  $\{z\}$  from noise prior  $U(-1, 1)$ ;
4 Train a GAN with  $\mathcal{F}$  and  $\{z\}$ , save its generator:  $G$ ;
5 for  $i = 1, \dots, k$  do
6   | Extract the corresponding variable  $z_{C_i}$  from  $\{z\}$  where  $G(z_{C_i}) = \hat{C}_i$ ;
7 end
8 //Interpolating between clusters;
9 for  $i = 1, \dots, k$  do
10  | for  $j = 1, \dots, k$  and  $j \neq i$  do
11    | | Interpolate the noise variable  $z_{ij}$  according to Equation (3.3);
12    | |  $Q = Q \cup G(z_{ij})$ ;
13  | end
14 end
15 //Interpolating in the cluster;
16 for  $i = 1, \dots, k$  do
17  | for  $x_a, x_b$  in  $k$ -th cluster do
18    | | Interpolate the noise variable  $z_{ab}$  according to Equation (3.4);
19    | |  $Q = Q \cup G(z_{ab})$ ;
20  | end
21 end
22 //Perturbation interpolation;
23 Add disturbance to noise variables  $\{z\}$  to generate variable  $\{z'\}$ ;
24  $Q = Q \cup G(\{z'\})$ ;
25 return  $Q$ 

```

Before interpolating, a standard GAN is trained with non-dominated solutions $\mathcal{F}$ and latent variables $\{z\} \sim U(-1, 1)$ to learn the manifold characteristics of solutions. After that, the well-trained *generator* : G and latent variables $\{z\}$ are saved in the archive for the later interpolation. In our work, three interpolation strategies are used: 1) interpolating between clusters, 2) interpolating in the cluster, and 3) perturbation interpolation.

1) Interpolating between clusters: This method is used to fill the gap between two clusters. Noise variables z_{C_i} are extracted from the archive, where z_{C_i} is the noise variable that generates

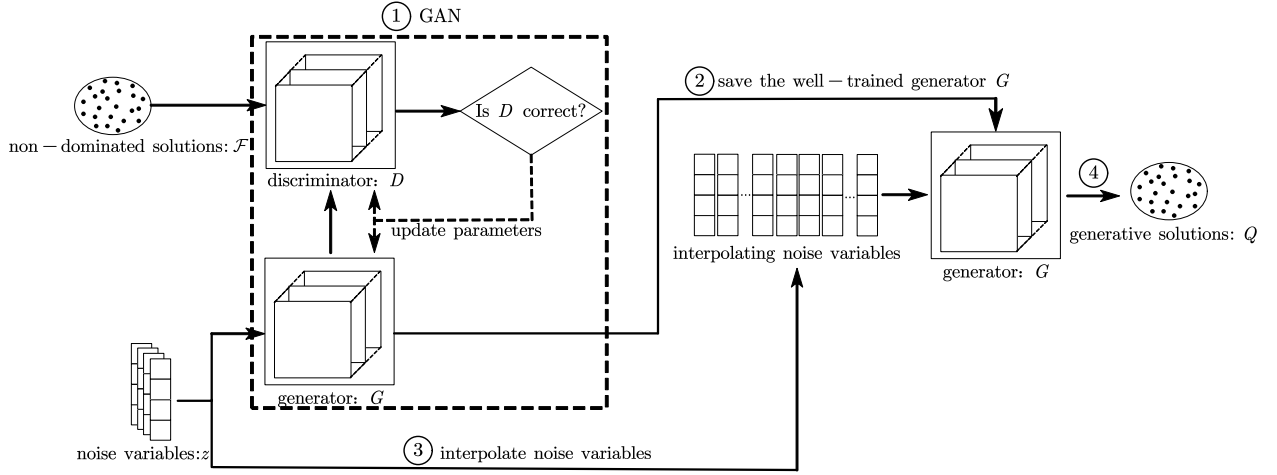


Figure 3.4: The flowchart of the manifold interpolation model. Step 1. Train the GAN with non-dominated solutions $\mathcal{F}$ and noise variables z . Step 2. Save the well-trained *generator* : G . Step 3. Apply interpolation strategies to z and input these interpolating noise variables into the *generator* : G . Step 4. Generate interpolating solutions Q .

the generative fake central solution $\hat{C}_i$ ($G(z_{C_i}) = \hat{C}_i$). For every two noise variables z_{C_i} and z_{C_j} of fake central solutions, interpolating noise variables between them can be described as [131]

$$z_{ij} = (1 - \alpha)z_{C_i} + \alpha z_{C_j}, i, j = 1, \dots, k, i \neq j \quad (3.3)$$

where $\alpha = l/n, l = 0, 1, \dots, n$, and n represents the dimension of the decision variable. Because z_{ij} are gradual between z_{C_i} and z_{C_j} , the generative solutions $G(z_{ij})$ generated by z_{ij} will also be gradual between the two clusters. For k clusters, the GAN will interpolate solutions between different central solutions, so it will interpolate $k * (k - 1)/2$ times.

2) Interpolating in the cluster: This method is used to fill gaps between individuals from the same cluster. The interpolating noise variables are described as

$$z_{ab} = \frac{z_a + z_b}{2}, \quad (3.4)$$

where z_a and z_b are the noise variables that generate fake solutions x_a and x_b , respectively, and x_a and x_b are from the same cluster. Similarly, these noise variables z_{ab} are inputted into the

generator : G . This interpolation method imitates the semantic operation for noise variables [132].

3) Perturbation interpolation: This method is used to apply disturbance on a noise variable to explore the neighbor space of a corresponding solution. Perturbation interpolation adds Gaussian noise to a random dimension of a noise variable. These noise variables are also inputted into the *generator* : G .

The details of the manifold interpolation are given in Algorithm 3, and an illustration of the manifold interpolation is shown in Fig. 3.4.

3.2.3 Manifold Selection Mechanism

To ensure that the generated solutions are of high quality, we propose a manifold selection mechanism to select solutions from the interpolated solutions set Q and use them to update the population.

There is a huge number of generative solutions in Q , and it is a waste of evaluations to calculate their objectives and determine whether a solution is high quality. To settle this issue, a manifold selection mechanism is proposed to select solutions that have the minimum distance to the manifold. The illustration of the manifold selection mechanism is shown in Fig. 3.5.

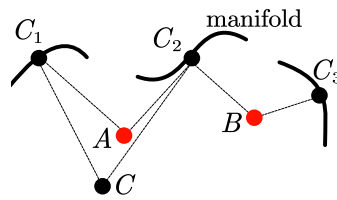


Figure 3.5: The proposed manifold selection mechanism: For solutions A and C , C_1 and C_2 are the two nearest central solutions, and for B , C_2 and C_3 are the two nearest central solutions. If two solutions should be selected, A and B are selected because they have the two minimum manifold distances.

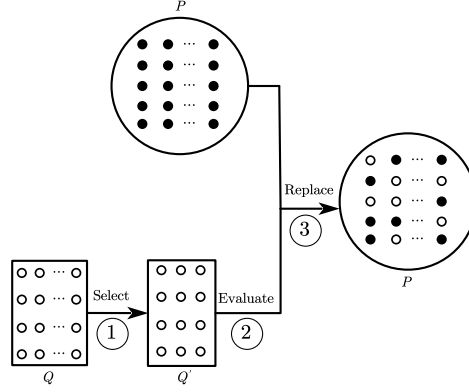


Figure 3.6: The selection process. Step 1. Select solutions from Q as Q' by the manifold selection mechanism. Step 2. Evaluate the solutions in Q' . Step 3. Replace solutions in P with those in Q' .

First, PCA is employed with C and Q to obtain the mapping manifold data $\phi(C)$ and $\phi(Q)$, respectively. Next, the manifold distance (MD) from each mapping solution $x \in \phi(Q)$ to the two nearest mapping central solutions $\phi(C_i)$ and $\phi(C_j)$ ($\phi(C_i), \phi(C_j) \in \phi(C)$) are computed

$$MD(x) = \|x - \phi(C_i)\|_2 + \|x - \phi(C_j)\|_2. \quad (3.5)$$

Then, $\text{top}-(\sigma \cdot |P|)$ solutions that have the minimum MD are selected from Q and denoted as Q' . Solutions in Q' are promising and can improve convergence or diversity because they are close to the existing piecewise manifolds or lie in the gap between piecewise manifolds. Finally, only solutions in Q' are evaluated and solutions $x_i \in P$ are replaced with $x_j \in Q'$, where $x_j \succ x_i$. The new population P is regarded as a parental population for the next round. Fig. 3.6 illustrates the selection process. The details of the selection are given in Algorithm 4.

3.2.4 Complexity Analysis

In this subsection, this work presents the time complexity analysis for each component of GAN-LMEF and gives an overall time complexity.

Algorithm 4: Selection**Input:** C (the central solutions), Q (the interpolated solutions), σ (the proportion of introducing interpolated solutions)**Output:** P (the population)

```

1 Use PCA with  $C$  and  $Q$  to get  $\phi(C)$  and  $\phi(Q)$ , respectively;
2 for  $x \in \phi(Q)$  do
3   for  $\phi(C_i), \phi(C_j) \in \phi(C)$  do
4     Find the two nearest  $\phi(C_i)$  and  $\phi(C_j)$  to  $x$ ;
5      $MD(x) = ||x - \phi(C_i)||_2 + ||x - \phi(C_j)||_2$ ;
6   end
7 end
8 Select top- $(\sigma \cdot |P|)$  minimum  $MD$  solutions from  $Q$  as  $Q'$ ;
9 Evaluate solutions in  $Q'$ ;
10 Replace solutions  $x_i \in P$  with  $x_j \in Q'$  where  $x_j \succ x_i$ ;
11 return  $P$ 

```

For the Algorithm 2, the k -means algorithm requires $O(k \cdot n \cdot N)$, and calculating central solutions for each cluster requires $O(N)$, where k is the number of clusters and n and N are the dimensions of the decision variables and the size of the population, respectively.

For the Algorithm 3, the generator of the GAN contains a three-layer fully-connected neural network [133], and the discriminator contains a three-layer fully-connected neural network in which a sigmoid unit is utilized as the output layer. It takes $O(\sum_{l=1}^d n_l \cdot n_{l-1} \cdot N^2 n^2)$ to execute the GAN once [134], where d is the depth of the neural networks, and n_l is the number of neurons in the l -th layer. Interpolation of solutions between central solutions of clusters requires the execution of the GAN $k * (k - 1)/2$ times, and the total time complexity of executing the GAN in our work is $O(k^2 \cdot (\sum_{l=1}^d n_l \cdot n_{l-1} \cdot N^2 n^2))$. The time complexity of the Algorithm Algorithm 3 is $O(k \cdot n \cdot N) + O(N) + O(k^2 \cdot (\sum_{l=1}^d n_l \cdot n_{l-1} \cdot N^2 n^2)) = O(k^2 \cdot (\sum_{l=1}^d n_l \cdot n_{l-1} \cdot N^2 n^2))$. The time complexity of the Algorithm 4 is $O(N \cdot k)$. Because k is a user-defined constant, the overall time complexity of GAN-LMEF is simplified as $O(\sum_{l=1}^d n_l \cdot n_{l-1} \cdot N^2 n^2)$.

3.3 Experiments

3.3.1 Experimental Settings

Algorithms in Comparison and Test Problems. The proposed GAN-LMEF is compared against several popular algorithms including WOF [22], LSMOF [21], GMOEA [23], LMOCSO [14], LMEA [17] and RM-MEDA [135]. The parameters of the WOF, LMEA, LSMOF, RM-MEDA, and LMOCSO algorithms are set to defaults according to [136]. For a fair comparison, the training parameters and the structure of the GAN in GMOEA are the same as those in ours. The basic solvers of the proposed GAN-LMEF, WOF [22], and LSMOF [21] are unified as NSGA2 [137], MOEADDE [138] and CMOPSO [139], respectively, and we rename them according to the basic solver, e.g., GAN-NSGA2, WOF-NSGA2, and LSMOF-NSGA2.

The widely used large-scale multiobjective test problem LSMOP1-LSMOP9 [140] is employed in our experiments. For all competitive comparisons, the number of objectives is set to 2 and the number of decision variables is set from 500 to 2000. For the ablation study, the number of objectives varies from 2 to 6, and the number of decision variables is fixed to 500.

Performance Indicators. In this study, the following metrics are utilized to evaluate the performance of compared algorithms.

1) Inverted Generational Distance (IGD) [141]: The IGD is a metric for quantifying the convergence of the solutions obtained by a multiobjective optimization algorithm. When the IGD value is small, the convergence of the solution is improved.

2) Schott's Spacing (SP) [142]: It measures the uniformity of the solutions found by the algorithm. The smaller the SP value is, the more uniform the distribution of the solution obtained by the algorithm is.

Parameter Settings. The parameters are set as follows.

1. Population Size: The population size N is set to 100 for all test instances.
2. Termination Condition: The number of maximum evaluations e is set to 100,000 for all compared LSMOAs.
3. The structure of the GAN is as follows: The number of nodes of the three-layer fully-connected neural network in the generator is set to $m - 1$, $\lceil n * 0.5 \rceil$, and n , respectively. The number of nodes of the fully-connected neural network in the discriminator is set to n , $\lceil n * 0.5 \rceil$, and 1, respectively.
4. Parameters of GAN-LMEF: The number of clusters k is set to 3. The proportion of the introduced interpolated solutions σ is set to 0.4. ϵ is set to 0.1. The learning rate of the GAN is set to 0.001, the epoch is set to 200, and the GAN is trained by full batch learning.

3.3.2 Comparisons with Baselines

The statistical results of the average IGD and SP values over 10 runs can be found in Tables 3.1 and 3.2, respectively. In the tables, (+), (=), and (-) indicate that GAN-NSGA2 is statistically significantly better, indifferent, or significantly worse than the compared algorithms, respectively, by using the Wilcoxon test (0.05 significance level).

As can be observed from Table 3.1, it is obvious that the proposed GAN-NSGA2 exhibits better performance than the five compared algorithms for convergence. GAN-NSGA2 achieved 17 of the 27 best results, LSMOF-NSGA2 achieved 6 of the best results, GMOEA achieved 2 of the best results, and WOF-NSGA2 achieved 2 of the best results for IGD values. Specifically, GAN-NSGA2 performs better on LSMOP1, LSMOP2, LSMOP4, LSMOP5, and LSMOP9 under all decision variable settings but is slightly worse than LSMOF-NSGA2 on the LSMOP3 and LSMOP6 problems. GMOEA can obtain a set of well-converged solutions for the LMSOP7 problems with 1000 and 2000 decision variables. WOF-NSGA2 achieves better convergence

Table 3.1: Mean and variance of IGD values obtained by compared algorithms over LSMOPs.

Problems	Dec.	GAN-NSGA2	WOF-NSGA2	LSMOF-NSGA2	GMOEA	LMOCSSO	LMEA	RM-MEDA
LSMOP1	500	2.47e-1(1.79e-3)	7.01e-1(5.87e-4)+	5.11e-1(4.37e-4)+	3.34e-1(1.62e-2)+	2.05e+0(1.05e+0)+	1.08e+1(1.45e-1)+	7.57e-1(1.98e-3)+
	1000	3.09e-1(1.59e-3)	7.02e-1(5.61e-4)+	4.83e-1(3.07e-4)+	5.10e-1(2.52e-3)+	1.36e+0(1.94e-1)+	1.11e+1(1.42e-1)+	1.23e+0(1.86e-3)+
	2000	3.36e-1(6.12e-3)	7.02e-1(7.06e-4)+	6.50e-1(3.08e-4)+	7.00e-1(9.85e-4)+	2.23e+0(7.92e-2)+	1.18e+1(1.87e-1)+	1.81e+0(5.31e-4)+
LSMOP2	500	1.59e-2(6.87e-6)	3.32e-2(1.49e-6)+	1.74e-2(4.53e-7)+	1.64e-2(4.25e-6)+	4.18e-2(8.12e-5)+	7.30e-2(7.49e-3)+	4.83e-2(2.01e-7)+
	1000	1.15e-2(2.43e-6)	1.90e-2(4.32e-7)+	1.53e-2(1.46e-7)+	1.38e-2(3.93e-6)+	2.49e-2(2.45e-5)+	4.05e-2(1.96e-3)+	3.10e-2(8.51e-6)+
	2000	8.58e-3(2.31e-7)	1.26e-2(1.89e-7)+	9.66e-3(9.64e-8)+	9.56e-3(3.20e-6)+	1.40e-2(6.84e-5)+	2.27e-2(6.14e-3)+	1.99e-2(2.39e-6)+
LSMOP3	500	1.91e+0(1.03e-2)	1.57e+0(2.99e-3)-	1.56e+0(3.31e-3)-	1.57e+0(1.87e-6)-	1.67e+0(1.21e+1)-	5.83e+1(5.06e+0)+	1.93e+1(8.07e-5)+
	1000	2.00e+0(1.22e-3)	1.58e+0(3.13e-3)-	1.57e+0(2.95e-3)-	1.58e+0(2.25e-6)-	4.49e+0(7.28e+1)+	1.48e+3(3.10e+3)+	2.86e+1(8.56e-5)+
	2000	2.18e+0(6.61e-4)	1.58e+0(3.22e-3)-	1.57e+0(2.96e-3)-	1.59e+0(7.76e-6)-	8.17e+0(1.23e+1)+	7.61e+2(6.77e+2)+	2.81e+1(2.71e-2)+
LSMOP4	500	3.88e-2(2.43e-5)	6.89e-2(5.93e-6)+	4.46e-2(2.52e-6)+	4.18e-2(5.15e-5)+	8.34e-2(1.05e-5)+	1.33e-1(2.13e-3)+	8.10e-2(3.40e-6)+
	1000	2.77e-2(6.74e-6)	6.46e-2(5.87e-6)+	2.98e-2(8.98e-7)+	2.93e-2(1.71e-5)+	5.24e-2(2.53e-6)+	7.75e-2(9.36e-3)+	5.57e-2(7.34e-6)+
	2000	1.64e-2(2.97e-6)	3.42e-2(1.41e-6)+	1.76e-2(2.75e-7)+	1.75e-2(1.00e-5)+	3.01e-2(9.23e-7)+	4.42e-2(2.64e-3)+	3.46e-2(4.19e-6)+
LSMOP5	500	6.67e-1(4.43e-4)	7.42e-1(3.67e-4)+	7.42e-1(7.23e-4)+	7.42e-1(3.34e-4)+	1.43e+0(3.54e+0)+	2.30e+1(5.86e-1)+	1.58e+0(9.01e-3)+
	1000	6.51e-1(2.96e-4)	7.41e-1(6.95e-4)+	7.42e-1(7.06e-4)+	7.42e-1(9.95e-4)+	3.49e+0(1.58e+0)+	2.34e+1(7.34e-1)+	3.19e+0(8.24e-3)+
	2000	6.71e-1(3.74e-5)	7.42e-1(7.36e-4)+	7.42e-1(6.45e-4)+	7.42e-1(1.19e-4)+	5.28e+0(5.32e-1)+	2.47e+1(8.29e-1)+	4.27e+0(6.16e-3)+
LSMOP6	500	8.64e-1(2.04e-1)	4.28e-1(2.41e-4)-	3.48e-1(1.81e-4)-	6.06e-1(1.86e-3)-	7.84e-1(1.02e-5)-	2.27e+3(7.25e+3)+	7.73e-1(5.58e-3)-
	1000	8.56e-1(3.10e-3)	4.57e-1(3.00e-4)-	3.40e-1(1.53e-4)-	6.74e-1(4.59e-3)-	7.66e-1(7.10e-4)-	4.65e+3(3.21e+3)+	5.03e-1(1.99e-3)-
	2000	1.21e+0(2.23e-2)	7.57e-1(6.02e-4)-	3.13e-1(1.15e-4)-	6.73e-1(3.03e-5)-	7.55e-1(9.81e-4)-	1.35e+3(2.84e+3)+	7.51e-1(1.36e-3)-
LSMOP7	500	1.54e+0(1.28e-1)	1.51e+0(2.54e-3)-	1.51e+0(3.05e-3)-	1.51e+0(5.14e-5)-	1.41e+2(1.77e+3)+	7.49e+4(7.89e+3)+	1.03e+2(2.51e+1)+
	1000	2.12e+0(4.01e-2)	1.51e+0(3.00e-3)-	1.51e+0(3.08e-3)-	1.51e+0(1.30e-5)-	2.00e+3(1.03e+3)+	8.41e+4(1.09e+3)+	6.87e+2(6.20e+1)+
	2000	2.48e+0(1.63e-2)	1.51e+0(2.54e-3)-	1.51e+0(3.41e-3)-	1.51e+0(1.51e-5)-	5.46e+3(9.28e+3)+	8.88e+4(9.86e+3)+	1.92e+3(7.65e+2)+
LSMOP8	500	6.56e-1(4.93e-5)	4.56e-1(2.48e-4)-	7.42e-1(6.08e-4)+	7.42e-1(3.45e-4)+	1.21e+0(1.45e+0)+	1.89e+1(3.80e-1)+	1.28e+0(1.02e-2)+
	1000	6.51e-1(3.11e-4)	7.42e-1(6.68e-4)+	7.42e-1(6.33e-4)+	7.42e-1(1.14e-4)+	2.74e+0(6.94e-1)+	2.02e+1(5.10e-1)+	1.94e+0(1.48e-2)+
	2000	6.48e-1(3.53e-4)	7.42e-1(6.32e-4)+	7.42e-1(6.28e-4)+	7.42e-1(5.26e-4)+	3.34e+0(4.04e-1)+	2.07e+1(6.73e-1)+	2.96e+0(5.94e-3)+
LSMOP9	500	5.34e-1(3.26e-3)	8.10e-1(9.09e-4)+	8.07e-1(8.61e-4)+	6.75e-1(9.22e-4)+	5.99e-1(5.59e-1)+	5.68e+1(3.45e+0)+	9.62e-1(2.06e-3)+
	1000	5.30e-1(3.17e-3)	8.09e-1(7.02e-4)+	7.97e-1(7.58e-4)+	6.59e-1(8.08e-3)+	1.08e+0(9.88e+1)+	5.91e+1(4.09e+0)+	1.76e+0(7.34e-3)+
	2000	6.40e-1(1.56e-5)	8.52e-1(3.02e-4)+	8.09e-1(7.66e-4)+	8.08e-1(2.39e-2)+	2.50e+0(1.87e+1)+	6.21e+1(5.20e+0)+	3.44e+0(1.85e-2)+
+/-/-			17/0/10	18/0/9	18/0/9	23/0/4	27/0/0	24/0/3

over POF on LSMOP7 and LSMOP8 with 500 decision variables. The statistical results of SP values are presented in Table 3.2. GAN-NSGA2 performs the best on 16 out of 27 test instances, followed by RM-MEDA with 6 best results, LMOCSSO with 4 best results, and WOF-NSGA2 with 1 best result.

The above experimental results suggest that GAN-NSGA2 can obtain a set of solutions with good convergence and diversity for most test problems. However, GAN-NSGA2 is worse for IGD values on LSMOP3 and LSMOP6. This may be attributed to the fitness landscapes of LSMOP3 being the adoption of Rosenbrock’s function and Rastrigin’s function. LSMOP6 mixes Rosenbrock’s function and Ackley’s function [140]. The mix of separable and nonseparable functions leads to a complicated fitness landscape and increases the difficulty of learning characteristics, thus degenerating the performance of the interpolated solutions.

Table 3.2: Mean and variance of SP values obtained by compared algorithms over LSMOPs.

Problems	Dec.	GAN-NSGA2	WOF-NSGA2	LSMOF-NSGA2	GMOEA	LMOCSO	LMEA	RM-MEDA
LSMOP1	500	5.14e-1(4.73e-2)	1.75e+0(3.73e-3)+	1.08e+0(2.00e-3)+	6.80e-1(1.49e-1)+	1.15e+0(6.47e-2)+	9.28e-1(9.48e-4)+	5.96e-1(3.54e-3)+
	1000	5.02e-1(6.72e-2)	9.99e-1(1.17e-3)+	1.04e+0(1.62e-3)+	1.61e+0(1.02e-1)+	1.23e+0(1.23e-2)+	9.73e-1(1.28e-3)+	5.90e-1(7.72e-4)+
	2000	5.09e-1(1.52e-1)	1.69e+0(3.09e-3)+	9.91e-1(1.19e-1)+	1.02e+0(1.08e-1)+	1.18e+0(4.12e-3)+	9.45e-1(1.39e-3)+	5.91e-1(3.57e-4)+
LSMOP2	500	1.13e-1(4.17e-3)	4.65e-1(3.40e-4)+	3.11e-1(9.98e-5)+	3.05e-1(1.53e-3)+	1.17e-1(3.17e-3)=	7.56e-1(7.51e-4)+	1.72e-1(1.43e-4)+
	1000	2.21e-1(1.42e-3)	4.18e-1(2.29e-4)+	3.66e-1(1.61e-4)+	3.00e-1(1.59e-3)+	7.36e-2(2.69e-3)-	6.70e-1(6.05e-4)+	1.48e-1(8.86e-5)-
	2000	2.33e-1(1.30e-3)	4.85e-1(3.37e-4)+	3.85e-1(1.99e-4)+	3.80e-1(1.70e-3)+	3.97e-2(3.28e-3)-	7.30e-1(6.11e-4)+	1.24e-1(1.37e-4)-
LSMOP3	500	8.30e-1(4.93e-4)	1.00e+0(1.21e-3)+	1.80e+0(4.72e-3)+	1.02e+0(1.35e-1)+	1.00e+0(1.33e-3)+	1.28e+0(1.92e-3)+	8.34e-1(2.58e-3)=
	1000	8.46e-1(1.90e-4)	1.00e+0(1.31e-3)+	1.81e+0(3.69e-3)+	1.05e+0(8.63e-2)+	2.28e+0(5.14e-1)+	9.93e-1(1.02e-3)+	8.10e-1(1.87e-3)-
	2000	8.63e-1(2.30e-2)	1.00e+0(1.35e-3)+	1.60e+0(5.21e-2)+	1.00e+0(1.71e-1)+	1.00e+0(1.11e-1)+	1.22e+0(2.32e-3)+	7.66e-1(5.60e-4)-
LSMOP4	500	2.63e-1(1.62e-3)	6.85e-1(5.49e-4)+	3.03e-1(1.18e-4)+	3.10e-1(2.55e-3)-	2.79e-1(1.13e-3)+	5.42e-1(3.81e-4)+	2.94e-1(2.42e-4)+
	1000	2.52e-1(1.35e-3)	5.08e-1(3.30e-4)+	3.14e-1(1.30e-4)+	3.07e-1(1.54e-3)+	1.79e-1(1.43e-3)-	6.90e-1(6.29e-4)+	2.60e-1(1.59e-4)=
	2000	2.55e-1(1.41e-3)	5.44e-1(4.00e-4)+	3.26e-1(1.35e-4)+	3.22e-1(1.59e-3)+	1.09e-1(1.75e-3)-	7.41e-1(7.39e-4)+	2.59e-1(1.22e-4)=
LSMOP5	500	7.93e-1(7.42e-2)	2.08e+0(5.83e-3)+	1.17e+0(1.81e-3)+	1.01e+0(1.05e-1)+	8.63e-1(1.93e-2)+	8.80e-1(1.05e-3)+	8.40e-1(1.58e-3)+
	1000	8.16e-1(2.01e-2)	1.44e+0(2.23e-3)+	1.04e+0(1.69e-3)+	1.01e+0(1.94e-1)+	1.13e+0(4.21e-3)+	1.01e+0(1.23e-3)+	8.45e-1(1.31e-3)+
	2000	9.01e-1(7.30e-3)	9.16e-1(1.18e-3)=	1.12e+0(1.56e-3)+	1.01e+0(1.11e-1)+	1.24e+0(8.34e-3)+	9.24e-1(1.10e-3)+	8.29e-1(8.12e-4)-
LSMOP6	500	9.70e-1(4.32e-1)	6.57e-1(5.70e-4)-	9.38e-1(1.00e-3)-	1.09e+0(1.05e-2)+	1.00e+0(1.29e-1)+	1.08e+0(1.36e-3)+	7.68e-1(2.46e-3)+
	1000	1.00e+0(4.20e-1)	1.03e+0(1.65e-3)=	9.55e-1(1.09e-3)-	9.96e-1(1.02e-2)-	1.00e+0(3.33e-1)=	9.42e-1(1.02e-3)-	5.58e-1(9.06e-3)-
	2000	9.31e-1(4.51e-1)	1.70e+0(3.88e-3)+	5.30e-1(3.51e-4)+	9.99e-1(2.44e-2)+	1.00e+0(1.90e-1)+	8.59e-1(1.00e-3)+	3.81e-1(2.04e-3)-
LSMOP7	500	5.92e-1(4.69e-1)	1.00e+0(1.25e-3)+	1.81e+0(4.21e-3)+	2.44e+0(3.47e-1)+	2.25e+0(1.35e-2)+	1.16e+0(2.09e-3)+	6.36e-1(4.64e-3)+
	1000	7.35e-1(3.19e-1)	1.00e+0(1.55e-3)=	2.03e+0(6.47e-3)+	2.40e+0(1.56e-1)+	8.96e-1(7.80e-3)-	7.68e-1(6.49e-4)+	8.83e-1(1.83e-1)+
	2000	9.97e-1(9.19e-3)	4.00e+0(1.77e-2)+	2.04e+0(4.65e-3)+	9.98e-1(4.51e-1)=	1.23e+0(1.84e-1)+	1.18e+0(2.07e-3)+	9.97e-1(3.08e-3)=
LSMOP8	500	6.07e-1(2.61e-1)	1.00e+0(1.11e-3)+	1.09e+0(1.57e-3)-	1.88e+0(1.90e-1)-	8.78e-1(4.13e-3)+	6.71e-1(4.97e-4)+	6.86e-1(6.56e-3)+
	1000	8.05e-1(2.61e-2)	1.89e+0(5.30e-3)+	1.04e+0(1.29e-3)+	1.10e+0(1.51e-1)+	8.45e-1(8.62e-4)+	8.08e-1(6.92e-4)=	7.94e-1(7.22e-4)=
	2000	8.18e-1(1.60e-2)	1.90e+0(5.16e-3)+	5.24e-1(3.28e-4)-	1.86e+0(1.63e-1)+	8.36e-1(1.14e-3)+	8.21e-1(9.83e-4)=	5.21e-1(5.13e-4)-
LSMOP9	500	8.54e-1(3.93e-4)	1.00e+0(1.14e-3)+	9.99e-1(1.49e-3)+	9.50e-1(8.61e-5)+	1.53e+0(3.52e-2)+	9.63e-1(1.43e-3)+	8.95e-1(4.21e-5)+
	1000	8.47e-1(1.97e-4)	1.00e+0(1.24e-3)+	9.95e-1(1.41e-3)+	9.05e-1(2.81e-3)+	1.67e+0(7.13e-2)+	8.59e-1(8.29e-4)+	9.01e-1(1.41e-3)+
	2000	8.08e-1(3.82e-4)	1.00e+0(1.35e-3)+	9.98e-1(1.49e-3)+	9.99e-1(3.70e-2)+	1.66e+0(4.91e-2)+	8.37e-1(7.56e-4)+	8.87e-1(1.01e-3)+
+/-			23/3/1	23/0/4	23/1/3	20/2/5	24/2/1	14/5/8

3.3.3 Ablation Studies

In this subsection, we perform ablation experiments to demonstrate the effectiveness of applying the proposed framework to NSGA2, MOEADDE, and CMOPSO. In addition, we embed basic solvers into two popular frameworks, WOF and LSMOF, for solving LSMOP. Please note that NSGA2, MOEADDE, and CMOPSO are not suitable for solving LSMOPs. The statistics of IGD values can be found in Table 3.3.

As seen from Table 3.3, GAN-NSGA2 (GAN-MOEADDE and GAN-CMOPSO) achieves significantly better IGD values than WOF-NSGA2 (WOF-MOEADDE and WOF-CMOPSO) and LSMOF-NSGA2 (LSMOF-MOEADDE and LSMOF-CMOPSO). Specifically, GAN-NSGA2 performs significantly better than WOF-NSGA2 in 17 out of 27 test functions and LSMOF-NSGA2

Table 3.3: Mean and variance of IGD values obtained by different basic solvers.

Problems	Dec.	GAN-NSGA2	WOF-NSGA2	LSMOF-NSGA2	GAN-MOEADDE	WOF-MOEADDE	LSMOF-MOEADDE	GAN-CMOPSO	WOF-CMOPSO	LSMOF-CMOPSO
LSMOP1	500	2.47e-1(1.79e-3)	7.01e-1(5.87e-4)+	5.11e-1(4.37e-4)+	1.61e-1(1.59e-2)	4.34e-1(8.39e-2)+	5.90e-1(4.22e-2)+	2.58e-1(1.77e-2)	4.34e-1(8.39e-2)+	5.92e-1(4.22e-2)+
	1000	3.09e-1(1.59e-3)	7.02e-1(5.61e-4)+	4.83e-1(3.07e-4)+	3.11e-1(6.87e-3)	5.15e-1(1.17e-1)+	6.20e-1(2.78e-2)+	5.98e-1(9.85e-9)	5.35e-1(1.37e-4)-	6.23e-1(2.78e-2)+
	2000	3.36e-1(6.12e-3)	7.02e-1(7.06e-4)+	6.50e-1(3.08e-4)+	3.43e-1(2.57e-3)	4.97e-1(8.38e-2)+	6.45e-1(1.58e-2)+	6.93e-1(1.20e-7)	6.77e-1(2.24e-7)+	6.47e-1(1.58e-2)+
LSMOP2	500	1.59e-2(6.87e-6)	3.32e-2(1.49e-6)+	1.74e-2(4.53e-7)+	2.34e-2(2.62e-6)	2.67e-2(2.36e-3)+	2.50e-2(1.31e-3)+	3.00e-2(5.37e-8)	2.67e-2(2.36e-3)-	3.18e-2(2.78e-8)+
	1000	1.15e-2(2.43e-6)	1.90e-2(4.32e-7)+	1.53e-2(1.46e-7)+	1.40e-2(4.30e-7)	1.81e-2(5.03e-4)+	1.81e-2(4.96e-4)+	1.71e-2(1.60e-8)	1.81e-2(5.03e-4)+	1.81e-2(4.96e-4)+
	2000	8.58e-3(2.31e-7)	1.26e-2(1.89e-7)+	9.66e-3(9.64e-8)+	8.48e-3(3.10e-7)	1.23e-2(3.14e-4)+	1.30e-2(3.66e-4)+	1.02e-2(2.31e-8)	1.27e-2(3.14e-4)+	1.33e-2(3.66e-4)+
LSMOP3	500	1.91e+0(1.03e-2)	1.57e+0(2.99e-3)-	1.56e+0(3.31e-3)-	1.46e+0(2.92e-2)	1.50e+0(4.14e-2)+	1.56e+0(6.81e-4)+	1.88e+0(9.46e-3)	1.54e+0(4.14e-2)-	1.56e+0(6.81e-4)-
	1000	2.00e+0(1.22e-3)	1.58e+0(3.13e-3)-	1.57e+0(2.95e-3)-	1.91e+0(5.17e-3)	1.54e+0(7.62e-2)-	1.55e+0(5.82e-4)-	2.07e+0(6.57e-4)	1.55e+0(7.62e-2)-	1.55e+0(5.82e-4)-
	2000	2.18e+0(6.61e-4)	1.58e+0(3.22e-3)-	1.57e+0(2.96e-3)-	1.85e+0(1.09e-2)	1.46e+0(1.07e-1)-	1.51e+0(3.24e-4)-	2.12e+0(3.99e-4)	1.46e+0(1.07e-1)-	1.57e+0(3.24e-4)-
LSMOP4	500	3.88e-2(2.43e-5)	6.89e-2(5.93e-6)+	4.46e-2(2.52e-6)+	4.02e-2(3.83e-5)	5.66e-2(4.60e-3)+	5.16e-2(1.39e-3)+	7.22e-2(2.41e-5)	5.66e-2(4.60e-3)-	5.17e-2(1.39e-3)-
	1000	2.77e-2(6.74e-6)	6.46e-2(5.87e-6)+	2.98e-2(8.98e-7)+	2.67e-2(1.66e-5)	3.64e-2(2.63e-3)+	3.29e-2(9.26e-4)+	5.06e-2(3.25e-6)	3.61e-2(2.63e-3)-	3.22e-2(9.26e-4)-
	2000	1.64e-2(2.97e-6)	3.42e-2(1.41e-6)+	1.76e-2(2.75e-7)+	1.45e-2(8.90e-6)	2.70e-2(1.97e-3)+	2.36e-2(1.28e-3)+	3.04e-2(4.20e-7)	2.76e-2(1.97e-3)-	2.36e-2(1.28e-3)-
LSMOP5	500	6.67e-1(4.43e-4)	7.42e-1(3.67e-4)+	7.42e-1(7.23e-4)+	6.37e-1(4.58e-5)	7.13e-1(7.12e-3)+	7.42e-1(1.19e-6)+	6.52e-1(1.48e-3)	7.44e-1(7.12e-3)+	7.42e-1(1.12e-3)+
	1000	6.51e-1(2.96e-4)	7.41e-1(6.95e-4)+	7.42e-1(7.06e-4)+	6.47e-1(4.00e-5)	7.47e-1(5.74e-3)+	7.42e-1(1.14e-6)+	6.56e-1(6.54e-6)	7.47e-1(5.74e-3)+	7.42e-1(1.46e-3)+
	2000	6.71e-1(3.74e-5)	7.42e-1(7.36e-4)+	7.42e-1(6.45e-4)+	6.56e-1(2.24e-5)	7.48e-2(4.32e-3)+	7.42e-1(1.14e-6)+	6.70e-1(2.81e-7)	7.49e-2(4.32e-3)+	7.42e-1(5.64e-3)+
LSMOP6	500	8.64e-1(2.04e-1)	4.28e-1(2.41e-4)-	3.48e-1(1.81e-4)-	3.89e-1(8.71e-4)	5.44e-1(1.43e-1)+	3.20e-1(4.08e-4)-	7.72e-1(3.33e-3)	5.44e-1(1.43e-1)-	3.20e-1(4.08e-4)-
	1000	8.56e-1(3.10e-3)	4.57e-1(3.00e-4)-	3.40e-1(1.53e-4)-	6.74e-1(7.84e-5)	5.69e-1(1.30e-1)-	3.35e-1(1.98e-4)-	8.09e-1(1.25e-2)	5.69e-1(1.30e-1)-	3.55e-1(1.98e-4)-
	2000	1.21e+0(2.23e-2)	7.57e-1(6.02e-4)-	3.13e-1(4.15e-4)-	3.34e-1(6.61e-4)	5.93e-1(1.23e-1)+	3.08e-1(5.62e-5)-	7.56e-1(1.73e-8)	5.93e-1(1.23e-1)-	5.57e-1(6.70e-3)-
LSMOP7	500	1.54e+0(1.28e-1)	1.51e+0(2.54e-3)-	1.51e+0(3.05e-3)-	1.50e+0(6.16e-5)	1.54e+0(1.14e-1)+	1.51e+0(1.18e-3)=	1.48e+0(1.25e-1)	1.54e+0(1.14e-1)+	1.50e+0(1.18e-3)=
	1000	2.12e+0(4.01e-2)	1.51e+0(3.00e-3)-	1.51e+0(3.08e-3)-	1.51e+0(3.89e-3)	2.43e+1(1.02e+2)+	1.51e+0(4.61e-4)=	1.56e+0(1.36e-1)	2.43e+1(1.02e+2)+	1.51e+0(4.61e-4)-
	2000	2.48e+0(1.63e-2)	1.51e+0(2.54e-3)-	1.51e+0(3.41e-3)-	2.59e+0(2.94e-1)	1.48e+0(9.01e-2)-	1.52e+0(3.71e-4)-	1.75e+0(1.12e-1)	1.42e+0(9.01e-2)-	1.51e+0(3.71e-4)-
LSMOP8	500	6.56e-1(4.93e-5)	4.56e-1(2.48e-4)-	7.42e-1(6.08e-4)+	6.45e-1(1.14e-5)	6.56e-1(1.23e-1)+	7.42e-1(1.14e-6)+	5.86e-1(8.34e-5)	6.56e-1(1.23e-1)+	7.42e-1(1.14e-6)+
	1000	6.51e-1(3.11e-4)	7.42e-1(6.68e-4)+	7.42e-1(6.33e-4)+	6.44e-1(2.89e-5)	7.45e-1(2.98e-3)+	7.42e-1(1.14e-6)+	6.55e-1(2.45e-6)	7.45e-1(2.98e-3)+	7.42e-1(1.14e-6)+
	2000	6.48e-1(3.53e-4)	7.42e-1(6.32e-4)+	7.42e-1(6.28e-4)+	6.58e-1(4.05e-6)	7.43e-1(1.71e-3)+	7.42e-1(1.14e-6)+	6.61e-1(2.01e-7)	7.43e-1(1.71e-3)+	7.42e-1(1.14e-6)+
LSMOP9	500	5.34e-1(3.26e-3)	8.10e-1(9.09e-4)+	8.07e-1(8.61e-4)+	6.05e-1(1.62e-6)	8.10e-1(2.30e-3)+	8.09e-1(6.05e-4)+	6.87e-1(2.36e-7)	8.10e-1(2.30e-3)+	8.09e-1(6.05e-4)+
	1000	5.30e-1(3.17e-3)	8.09e-1(7.02e-4)+	7.97e-1(7.58e-4)+	6.11e-1(1.30e-4)	8.11e-1(3.02e-3)+	8.01e-1(1.05e-3)+	6.67e-1(1.49e-6)	8.13e-1(3.02e-3)+	8.08e-1(1.05e-3)+
	2000	6.40e-1(1.56e-5)	8.52e-1(3.02e-4)+	8.09e-1(7.66e-4)+	6.74e-1(4.30e-6)	8.11e-1(5.16e-3)+	8.09e-1(1.53e-3)+	6.57e-1(8.62e-7)	8.11e-1(5.16e-3)+	8.05e-1(1.53e-3)+
+/-			17/0/10	18/0/9	23/0/4	19/2/6	15/0/12		15/1/11	

in 18 out of 27 test functions. GAN-MOEADDE has better IGD values than WOF-MOEADDE in 23 out of 27 test functions and LSMOF-MOEADDE in 19 out of 27 instances. GAN-CMOPSO obtains better IGD values than WOF-CMOPSO in 15 out of 27 instances and LSMOF-CMOPSO in 15 out of 27 functions. These pairwise comparisons have demonstrated the ability of our proposed GAN-LEMF framework for improving the performance of existing MOAs on LSMOPs.

Table 3.4: Mean and variance of IGD values obtained by compared algorithms over LSMOPs when the number of decision variables is 500.

Problems	GAN-NSGA2	Ip-NSGA2	W/O _n -NSGA2	NSGA2
LSMOP1	2.47e-1(1.79e-3)	6.53e-1(1.05e-3)+	3.21e-1(5.20e-3)+	1.80e+0(1.57e+0)+
LSMOP2	1.59e-2(6.87e-6)	4.75e-1(5.41e-4)+	1.80e-2(8.91e-6)+	5.91e-2(1.40e-5)+
LSMOP3	1.91e+0(1.03e-2)	7.45e-1(1.36e-3)-	3.73e+0(2.69e-1)+	1.87e+1(7.11e+1)+
LSMOP4	3.88e-2(2.43e-5)	6.12e-1(8.24e-4)+	4.00e-2(5.86e-5)=	9.15e-2(9.31e-5)+
LSMOP5	6.67e-1(4.43e-4)	1.06e+0(2.31e-3)+	1.54e+0(4.48e-2)+	5.69e+0(6.28e+0)+
LSMOP6	8.64e-1(2.04e-1)	7.61e-1(1.28e-3)-	7.37e-1(4.87e-1)-	8.07e-1(8.27e-3)-
LSMOP7	1.54e+0(1.28e-1)	4.81e-1(4.75e-4)-	1.51e+0(4.15e-1)=	1.50e+1(1.18e+1)+
LSMOP8	6.56e-1(4.93e-5)	8.64e-1(1.83e-3)+	7.59e-1(5.20e-5)+	1.53e+0(8.96e+0)+
LSMOP9	5.34e-1(3.26e-3)	5.69e-1(7.80e-4)+	9.57e-1(2.77e-3)+	1.23e+0(4.46e+1)+
+/-/-		6/0/3	6/2/1	8/0/1

To verify the effectiveness of interpolation for latent noise variables z , this work develops Ip-LMEF that directly uses the piecewise linear interpolation method to interpolate solutions between the decision variables of central solutions. Also, W/O_n-LMEF is designed to generate solutions via GAN but without interpolating noise variables. The experimental results are shown in Table 3.4. As seen from Table 3.4, GAN-NSGA2 achieves 6 of the 9 best IGD values, while Ip-NSGA2 achieves 3 of the best results. this work also draws convergence profiles of IGD values obtained by GAN-NSGA2 and NSGA2 in Fig. 3.7. The convergence rate of GAN-NSGA2 is faster than that of NSGA2. As seen in the experimental results, incorporation of GAN into NSGA2 can greatly improve the quality and convergence speed of solutions in solving LSMOPs. These results confirm the effectiveness of our proposed interpolation method. The GAN learns the mapping relationship from the latent low-dimensional manifold space to the

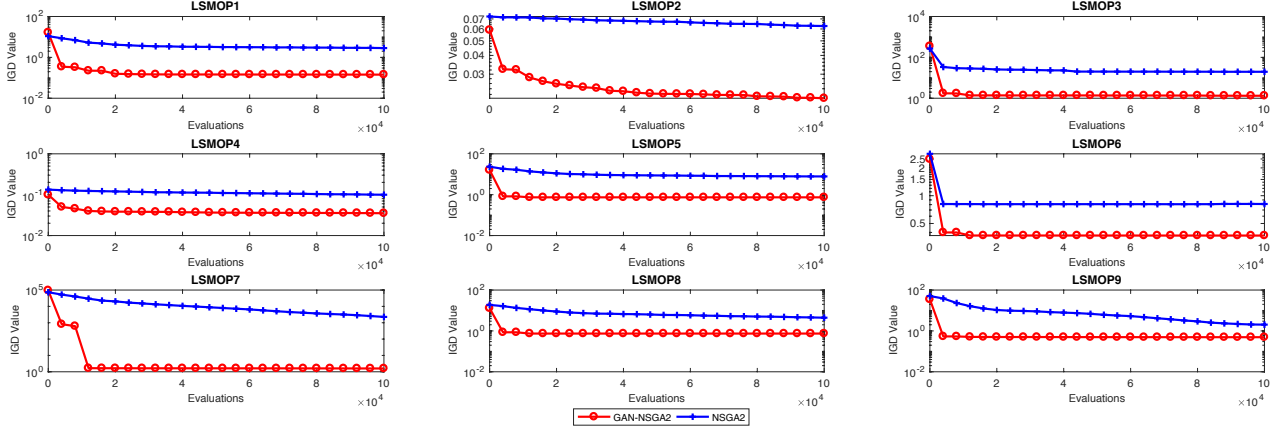


Figure 3.7: Convergence profiles of IGD values obtained by GAN-NSGA2 and NSGA2 on LSMOP problems with 500 decision variables.

high-dimensional solutions, and interpolation of the latent variables z can generate high-quality solutions lying on the manifold.

3.3.4 Performance of Different Interpolation Strategies

To investigate the quality of solutions interpolated via different interpolation strategies, we record the number of different interpolated non-dominated solutions selected by the Algorithm 4 during evolutionary optimization.

As shown in Fig. 3.8, interpolating between central solutions can produce much better individuals than the perturbation interpolation method and interpolating in the cluster method. This indicates that the gaps between clusters are always huge in solving LSMOPs, and the population lacks diversity and convergence during the entire evolutionary optimization. Therefore, interpolating between central solutions can produce more promising solutions.

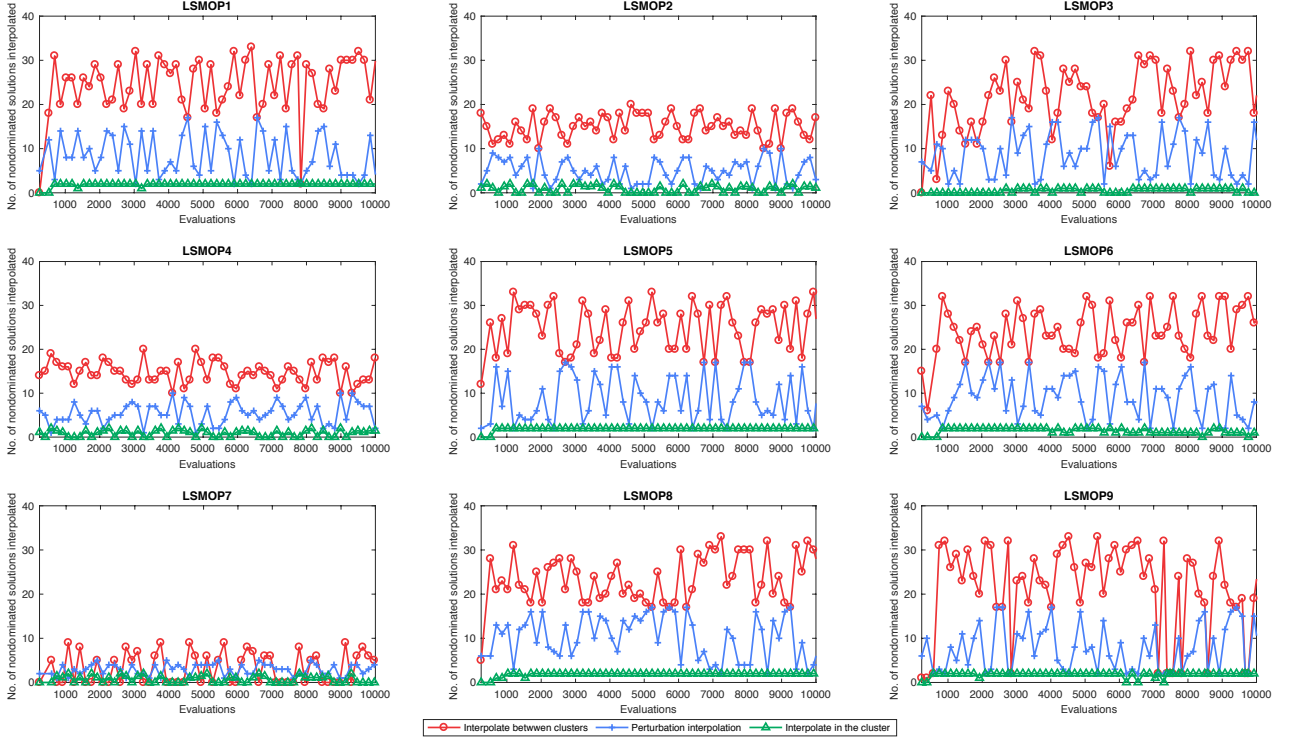


Figure 3.8: The number of selected non-dominated solutions that are interpolated by different strategies on LSMOP problems with 500 decision variables.

3.3.5 Sensitivity Studies

In GAN-LMEF, the parameters k and ϵ decide the number of interpolated solutions and affect the performance. In this sensitivity study, the influence of k and ϵ on the convergence of GAN-NSGA2 is investigated. Table 3.5 shows IGD values obtained by different choices of k and ϵ values for GAN-NSGA2 on the LSMOP benchmark. When k increases from 2 to 3, IGD values improve in 18 out of 27 instances. When ϵ varies from 0.1 to 0.5, half of the instances in terms of IGD values become small, while half of the instances increase. Based on the experimental results, a larger k value can generate more piecewise manifolds, thus interpolating many more solutions in the gaps. While the choice of parameter ϵ depends on the specific problem, it needs to be set carefully.

Table 3.5: Mean and variance of IGD values obtained by different ϵ and k values of GAN-NSGA2.

Problems	ϵ	k=2	k=3
LSMOP1	0.1	2.40e-1(1.45e-3)	2.47e-1(1.79e-3)
	0.2	2.31e-1(4.00e-3)	2.35e-1(3.31e-3)
	0.5	2.43e-1(6.19e-3)	2.38e-1(4.36e-3)
LSMOP2	0.1	1.65e-2(3.72e-6)	1.59e-2(6.87e-6)
	0.2	1.77e-2(5.61e-6)	1.50e-2(4.63e-6)
	0.5	1.69e-2(4.63e-6)	1.52e-2(4.39e-6)
LSMOP3	0.1	1.86e+0(2.16e-2)	1.91e+0(1.03e-2)
	0.2	1.77e+0(4.52e-2)	1.87e+0(2.11e-2)
	0.5	1.89e+0(4.80e-2)	1.80e+0(1.65e-2)
LSMOP4	0.1	3.97e-2(3.47e-5)	3.88e-2(2.43e-5)
	0.2	3.82e-2(4.55e-5)	3.67e-2(3.26e-5)
	0.5	3.59e-2(5.10e-5)	3.20e-2(4.50e-5)
LSMOP5	0.1	6.59e-1(7.10e-4)	6.67e-1(4.43e-4)
	0.2	6.90e-1(7.72e-4)	6.74e-1(7.99e-4)
	0.5	7.06e-1(5.45e-4)	6.86e-1(3.48e-4)
LSMOP6	0.1	8.82e-1(3.16e-1)	8.64e-1(2.04e-1)
	0.2	8.96e-1(5.20e-1)	8.49e-1(2.18e-1)
	0.5	9.90e-1(3.79e-1)	9.74e-1(6.16e-1)
LSMOP7	0.1	1.54e+0(3.38e-1)	1.54e+0(1.28e-1)
	0.2	1.57e+0(2.79e-1)	1.53e+0(2.04e-1)
	0.5	1.80e+0(7.07e-1)	1.78e+0(5.52e-1)
LSMOP8	0.1	7.10e-1(5.13e-5)	6.56e-1(4.93e-5)
	0.2	7.35e-1(4.34e-5)	6.61e-1(6.50e-5)
	0.5	7.42e-1(6.34e-5)	6.72e-1(4.83e-5)
LSMOP9	0.1	5.32e-1(1.18e-3)	5.34e-1(3.26e-3)
	0.2	5.30e-1(7.13e-3)	5.31e-1(2.63e-3)
	0.5	5.21e-1(4.10e-3)	5.25e-1(3.96e-3)

In addition, the influence of the proportion of interpolated solutions σ and the first evaluations for interpolating $\epsilon \cdot e$ are investigated, and the plots are shown in Fig. 3.9. We can observe that interpolating at the early evolution stage achieves better IGD values in most cases. This is because the proposed manifold interpolation model has a powerful ability of exploration. Besides, the parameter σ is not significantly sensitive to the experimental results.

Because POS can be represented in an $m - 1$ -dimensional manifold, the number of objectives m can determine the complexity of the manifold, thus affecting the performance of interpolation. Therefore, we conduct the experiment concerning m , and the experimental results concerning the IGD metric are shown in Table 3.6. GAN-NSGA2 achieves better IGD values in 12 out of 27 instances. However, we can immediately find that GAN-NSGA2 has more difficulty

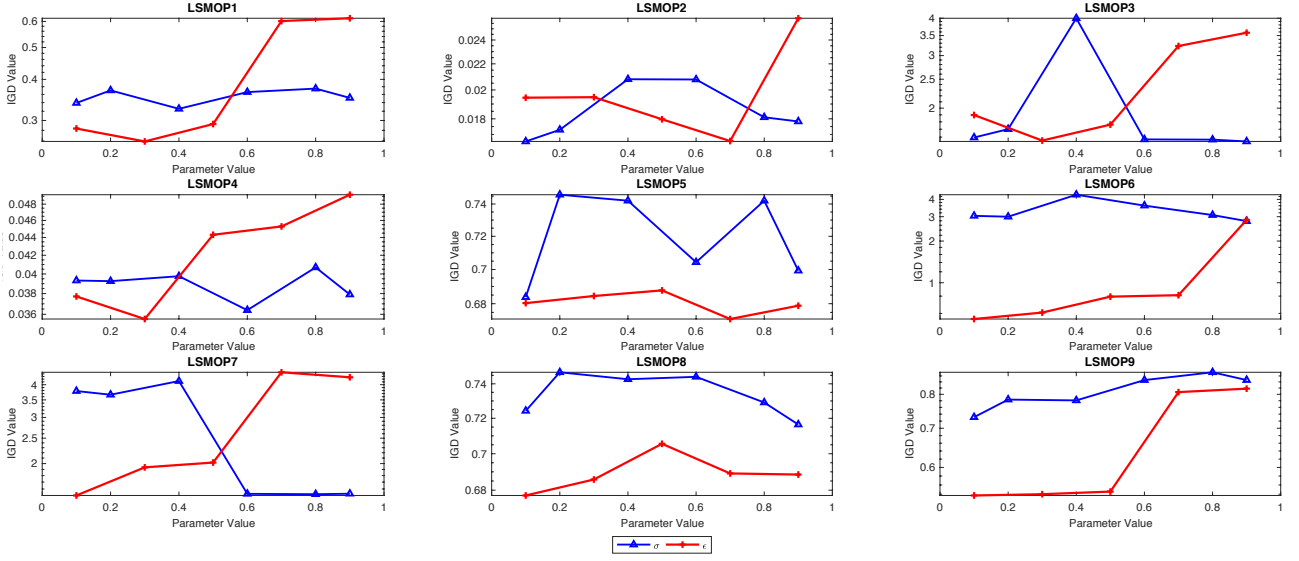


Figure 3.9: The influence of the proportion of interpolated solutions σ and the first evaluations for interpolating $\epsilon \cdot e$.

obtaining well-converged solutions on more objective functions because the manifold becomes more complicated and more difficult to learn.

3.3.6 Discussion

During the past few decades, many crossover operators have been proposed. Among these operators, simulated binary crossover exhibits excellent performance and has been widely used. However, as the fitness landscape becomes more complicated and contains more local optima, the offspring generated by simulated binary crossover is more difficult to escape from the local optimum. Therefore, we design a model-based approach to partially replace genetic operators to reproduce offspring. We notice that POS is distributed on the low-dimensional manifold, and the GAN can be employed to learn the manifold. When the GAN is well-trained, GAN can automatically learn the manifold of data, and the similarity between two generated samples can be approximated as the Euclidean distance between their corresponding latent variables.

Table 3.6: Mean and variance of IGD values obtained by compared algorithms over LSMOPs with various numbers of objective functions.

Problems	m	GAN-NSGA2	WOF-NSGA2	LSMOF-NSGA2	GMOEA	LMOCOS	LMEA	RM-MEDA
LSMOP1	2	2.47e-1(1.79e-3)	7.01e-1(5.87e-4)+	5.11e-1(4.37e-4)+	3.34e-1(1.62e-2)+	2.05e+0(1.05e+0)+	1.08e+1(1.45e-1)+	7.57e-1(1.98e-3)+
	4	4.19e-1(6.71e-3)	8.65e-1(1.22e-4)+	8.77e-1(1.37e-6)+	8.17e-1(4.36e-4)+	6.18e-1(3.56e-5)+	4.35e-1(2.55e-4)+	2.61e+0(2.33e-1)+
	6	8.63e-1(5.50e-3)	9.50e-1(1.36e-3)+	9.50e-1(1.36e-3)+	9.57e-1(1.80e-4)+	8.63e-1(9.21e-5)=	4.45e-1(2.20e-4)-	6.04e+0(1.99e-1)+
LSMOP2	2	1.59e-2(6.87e-6)	3.32e-2(1.49e-6)+	1.74e-2(4.53e-7)+	1.64e-2(4.25e-6)+	4.18e-2(8.12e-5)+	7.30e-2(7.49e-3)+	4.83e-2(2.01e-7)+
	4	1.61e-1(3.59e-5)	1.80e-1(2.10e-5)+	1.48e-1(4.93e-6)-	1.48e-1(2.65e-5)-	1.49e-1(2.86e-4)-	5.11e-1(2.81e-4)+	1.54e-1(1.57e-3)=
	6	1.96e-1(8.03e-5)	2.87e-1(6.33e-5)+	2.58e-1(1.66e-5)+	2.47e-1(5.55e-5)+	2.67e-1(1.03e-4)+	6.31e-1(5.93e-4)+	2.69e-1(2.11e-3)+
LSMOP3	2	1.91e+0(1.03e-2)	1.57e+0(2.99e-3)-	1.56e+0(3.31e-3)-	1.57e+0(1.87e-6)-	1.67e+0(1.21e-1)-	5.83e+1(5.06e+0)+	1.93e+1(8.07e-5)+
	4	4.70e+0(2.66e-2)	1.80e+0(5.42e-3)-	1.79e+0(2.17e-3)-	1.80e+0(1.16e-6)-	1.15e+1(5.50e-1)+	7.11e-1(7.11e-4)-	1.51e+1(2.01e-1)+
	6	4.72e+0(2.10e-1)	1.85e+0(4.88e-3)-	1.17e+0(5.42e-2)-	1.85e+0(1.18e-6)-	2.08e+1(4.62e-1)+	8.25e-1(9.69e-4)-	1.64e+1(4.18e-3)+
LSMOP4	2	3.88e-2(6.43e-5)	6.89e-2(5.93e-6)+	4.46e-2(2.52e-6)+	4.18e-2(5.15e-5)+	8.34e-2(1.05e-5)+	1.33e-1(2.13e-3)+	8.10e-2(3.40e-6)+
	4	1.60e-1(4.80e-5)	2.37e-1(3.27e-6)+	2.15e-1(5.23e-5)+	2.13e-1(6.83e-5)+	2.11e-1(1.46e-3)+	5.57e-1(4.09e-4)+	2.28e-1(2.60e-4)+
	6	2.80e-1(6.09e-5)	4.02e-1(9.15e-5)+	3.48e-1(2.11e-4)+	3.47e-1(2.89e-4)+	3.63e-1(3.41e-4)+	6.27e-1(4.49e-4)+	3.48e-1(1.52e-3)+
LSMOP5	2	6.67e-1(4.43e-4)	7.42e-1(3.67e-4)+	7.42e-1(7.23e-4)+	7.42e-1(3.34e-4)+	1.43e+0(3.54e+0)+	2.30e+1(5.86e-1)+	1.58e+0(9.01e-3)+
	4	6.69e-1(4.08e-4)	4.64e-1(1.81e-6)-	4.80e-1(2.91e-6)-	4.87e-1(8.61e-3)-	1.84e+0(2.92e+0)+	9.33e-1(1.36e-3)+	5.18e+0(3.70e+0)+
	6	7.31e-1(4.90e-4)	7.40e-1(1.63e-3)+	9.56e-1(1.01e-5)+	8.13e-1(6.66e-3)+	2.14e+0(5.08e+0)+	1.18e+0(1.00e-3)+	1.54e+1(1.18e+1)+
LSMOP6	2	8.64e-1(2.04e-1)	4.28e-1(2.41e-4)-	3.48e-1(1.81e-4)-	6.06e-1(1.86e-3)-	7.84e-1(1.02e-5)-	2.27e+3(7.25e+3)+	7.73e-1(5.58e-3)-
	4	9.00e-1(7.04e-3)	9.29e-1(7.21e-7)+	9.45e-1(1.01e-4)+	9.07e-1(1.59e-3)+	1.14e+0(6.66e-2)+	9.15e-1(1.12e-3)+	1.09e+0(7.93e-3)+
	6	1.22e+0(2.51e-1)	1.39e+0(4.92e-5)+	1.20e+0(2.64e-4)=	1.55e+0(8.53e+1)+	1.40e+0(5.42e-1)+	1.20e+0(1.36e-1)=	1.48e+0(1.24e-3)+
LSMOP7	2	1.54e+0(1.28e-1)	1.51e+0(2.54e-3)=	1.51e+0(3.05e-3)=	1.51e+0(5.14e-5)=	1.41e+2(1.77e+3)+	7.49e+4(7.89e+3)+	1.03e+2(2.51e+1)+
	4	1.38e+0(4.81e-1)	1.23e+0(1.43e-1)-	1.34e+0(1.09e-4)=	1.23e+0(1.79e-2)-	2.91e+2(9.60e+2)+	1.44e+0(2.59e-3)+	1.43e+3(1.23e+2)+
	6	1.60e+0(7.22e-1)	1.67e+0(3.73e-5)+	1.86e+0(2.73e-4)+	3.40e+1(5.91e+1)+	1.58e+0(1.68e-1)=	9.74e-1(1.64e-3)-	2.74e+4(5.02e+3)+
LSMOP8	2	6.56e-1(4.93e-5)	4.56e-1(2.48e-4)-	7.42e-1(6.08e-4)+	7.42e-1(3.45e-4)+	1.21e+0(1.45e+0)+	1.89e+1(3.80e-1)+	1.28e+0(1.02e-2)+
	4	6.80e-1(1.39e-5)	4.59e-1(6.46e-7)-	4.39e-1(1.19e-4)-	4.53e-1(5.29e-3)-	2.12e+0(1.64e+0)+	8.11e-1(1.02e-3)+	3.26e+0(1.01e+0)+
	6	6.24e-1(7.37e-5)	6.54e-1(2.13e-5)+	6.76e-1(1.08e-4)+	1.15e+0(1.73e+0)+	1.50e+0(1.74e-0)+	8.02e-1(9.56e-4)+	9.27e+0(1.79e+0)+
LSMOP9	2	5.34e-1(3.26e-3)	8.10e-1(9.09e-4)+	8.07e-1(8.61e-4)+	6.75e-1(9.22e-4)+	5.99e-1(5.59e-1)+	5.68e+1(3.45e+0)+	9.62e-1(2.06e-3)+
	4	8.53e-1(6.92e-3)	1.46e+0(2.30e-8)+	2.25e+0(3.00e-2)+	5.68e-1(9.05e-2)-	6.95e-1(7.52e-1)-	9.51e-1(1.00e-3)+	1.84e+1(6.94e+0)+
	6	2.14e+0(1.30e-1)	7.29e-1(9.46e-5)-	3.71e+0(5.51e-2)+	8.74e-1(3.03e-1)-	2.30e+0(7.69e-1)+	9.91e-1(1.53e-3)-	1.45e+2(1.30e+1)+
+/-			17/1/9	17/3/7	16/1/10	21/2/4	21/1/5	25/1/1

Therefore, interpolating z_1 and z_2 can generate uniform samples located on the manifold. In such a way, individuals with uniformity and diversity are introduced, thus helping the population escape from the local optima and better-handling LSMOPs.

It is worth noting that the proposed GAN-LMEF needs extra computational resources to train the model. Fig. 3.10 presents the relationship between the number of decision variables and the wall-clock time of executing Line 4-9 in Algorithm 1 once. The wall-clock time increases linearly with the number of decision variables. When the parameter ε is set to 0.1, σ is 0.4, population size N is 100, and the total evaluations e is set to 100,000, there will be $\frac{e \times \varepsilon}{(1+\sigma) \times N} \approx 70$ times of process of executing Line 4-9 in Algorithm 1. According to Fig. 3.10, we can see that the good performance of GAN-LMEF comes with a small price. We suggest carefully tuning these

parameters according to the specific problem to meet the trade-off between the computation time and solution quality.

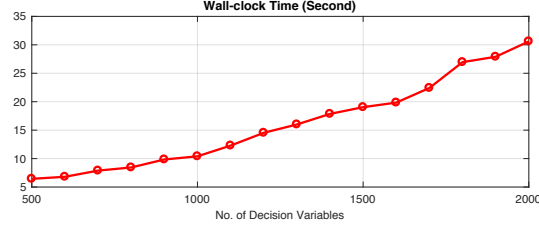


Figure 3.10: The relationship between the number of decision variables and the wall-clock time of executing Line 4-9 in Algorithm 1 once.

3.4 Conclusion

This work has presented a GAN-based evolutionary search framework for solving LSMOPs, called GAN-LMEF. The aim of GAN-LMEF is to interpolate solutions on the manifold via a GAN to maintain the effective manifold. Based on the generative population, higher-quality offspring are reproduced, thereby improving evolutionary performance.

In the proposed GAN-LMEF, a GAN is employed to learn various characteristics from the non-dominated solutions and generate a number of solutions using three different interpolation strategies. Then, a manifold selection mechanism selects promising solutions from the generative solutions for the next round. We integrated the proposed framework with NSGA2, MOEADDE, and CMOPSO and evaluated their performances. The experimental results have demonstrated that the proposed GAN-LMEF is more competitive than several state-of-the-art LSMOAs.

Several possible directions may be taken for future work. For large-scale problems with complicated search spaces, such as constraints and disconnected Pareto-optimal regions, the proposed framework still has much room for improvement. In addition, a rich body of transfer techniques,

such as evolutionary transfer optimization and advanced machine learning, can inspire further innovations in solving real-world applications with a large number of decision variables.

Chapter 4

Knowledge-Driven Adaptive Resource Allocation for Dynamic Multiobjective Optimization

4.1 Motivation

DMOPs are optimization problems with time-varying, conflicting objective functions, which are widely present in a rich body of real-world applications, such as job scheduling [143, 144], vehicle routing [145, 5], etc. For example, in the optimization of vehicle routing for package delivery [5, 146, 147, 148], environmental variables, including the number of vehicles, the time window of customers, and route topology change the objective functions of the problems, e.g., energy consumption and travel time. In such a constantly changing environment, the decision-maker is required to track the changing Pareto-optimal solutions to accommodate the dynamic scenarios.

To track the time-varying Pareto-optimal solutions, a number of DMOEAs have been designed,

and they can be roughly grouped into three categories: diversity enhancement/maintenance-based methods [3, 26, 32], memory-based methods [33, 36, 37], and prediction-based methods [51, 47, 46]. Although a plethora of DMOEAs has been proposed, little attention has been paid to exploiting computational resources for DMOPs. Indeed, to resolve DMOPs well, sufficient computational resources are required to track the Pareto-optimal solutions of each environment. However, due to the successive environmental changes and limited computing resources, it is intractable to provide sufficient computational resources for each environment [149]. The contradiction degrades the performance of DMOEAs and limits their practical applications.

To exploit computational resources, multipopulation-based EAs activate high-performance subpopulations and allocate more resources to the activated subpopulations for efficient problem-solving [150, 151, 152]. However, two issues limit the effectiveness of the management of computational resources. Firstly, existing multipopulation approaches fall short of leveraging characteristics of the varying environment to activate subpopulations, thus allocating resources to suboptimal subpopulations. As presented in Fig. 4.1 (a), due to the changing search space, keeping the subpopulation 1 activated when the time changes from $t = 0$ to $t = 1$ could lead to the local optimum. Secondly, the evolutionary operator with the distinct search bias cannot be adaptively switched according to the changing environment [149], which results in the waste of computational resources on improper evolutionary search strategies. As illustrated in Fig. 4.1 (b), the activated subpopulation should adopt different search strategies according to the changing properties of the search space to escape the local optimum (e.g., the subpopulation 1 at $t = 0$) or strengthen the local search (e.g., the subpopulation 1 at $t = 1$). Otherwise, the computational resources could be wasted on over-exploration and over-exploitation. In a nutshell, one efficient computational resource management strategy should leverage the changing problem properties to activate promising subpopulations with proper evolutionary strategies.

Keeping the above in mind, this work takes Monte Carlo tree search (MCTS) as a computational resource management component due to the following reasons: (1) MCTS uses the upper confi-

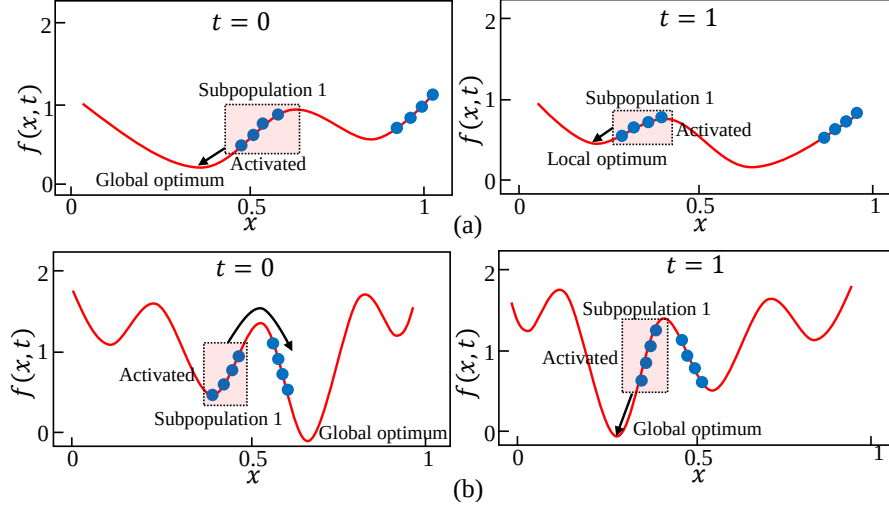


Figure 4.1: Limitations of existing DMOEAs. (a) When the environment changes from $t = 0$ to $t = 1$, keeping the subpopulation 1 activated could lead to the local optimum. (b) The search strategies of the activated subpopulation should be adaptively changed according to the changing environment. Otherwise, the population could be trapped in local optimum or slow convergence.

dence bound (UCB) to quantitatively estimate the benefits of searching towards a region [153]. Taking the characteristics, subpopulations that cover promising search regions can be activated to avoid local optima and slow convergence. (2) MCTS can serve as a decision-making agent with very little domain-specific knowledge [153]. By sampling different evolutionary operators for subpopulations' evolution with limited computational resources, the tree policy of MCTS allows the different evolutionary operators' performance to be estimated in the newly generated environment, thus adaptively adopting proper evolutionary operators to accommodate the changing environment. By taking the two advantages of MCTS, the proposed algorithm first hierarchically partitions whole populations into a number of subpopulations. According to the UCB values of subpopulations, the MCTS-assisted computational resource allocation strategy activates subpopulations covering promising search regions and allocates more computing resources to the subpopulations. Then, by estimating the performance of different evolutionary

operators through tree policy, the activated subpopulations can flexibly select their evolutionary operators to improve the search efficiency of the new environment. The contributions of this work are summarized as follows.

1. To drive subpopulations towards promising search regions, a UCB metric is designed to estimate the potential search benefits of different subpopulations. The designed UCB metric estimates the search benefits by considering a delicate exploration-exploitation trade-off of subpopulations, thus reducing the waste of computational resources on over-exploration and over-exploitation.
2. To reduce the computational waste on improper evolutionary operators, the evolutionary operator of the activated subpopulation can be flexibly changed by the tree policy of MCTS, enabling the evolutionary strategies to adapt to the characteristics of the changing environment.
3. The proposed method improves the utilization of computational resources to solve DMOPs, which advances DMOEAs toward a more practical scenario where computational resources are limited for each environment. The experimental studies have verified the efficacy of the proposed approach.

4.2 Proposed Framework

In this section, we first present the overall design of the MCTS-assisted DMOEA (MCTS-DMOEAs). Then, each component of the proposed computational resource allocation approach in the MCTS-DMOEAs is detailed.

4.2.1 Overview

Briefly, once a change is detected¹, MCTS-DMOEA has two main steps to handle the new environment: Algorithm 6 and Algorithm 7. Algorithm 6 aims to create several subpopulations covering different search regions. After that, Algorithm 7 determines which subpopulation should be activated and the specific evolutionary strategy of the activated subpopulation. In particular, this work uses simulated binary crossover (SBX) and differential evolution (DE) operators as the specific evolutionary strategies of the activated subpopulation in Procedure MCTS RESOURCE ALLOCATION. DE has been widely considered and used as a numerical global optimizer. However, the ability to exploit is relatively insufficient in the late evolution. SBX could produce offspring individuals that are similar to the parents, and thus, it is widely used as a local search operator. However, in the later stage of evolution, the population is relatively aggregated, making the offspring generated very similar and lacking exploration ability. Considering that the SBX and DE operators have distinct search biases, we thus adopt SBX and DE operators for the activated subpopulation. In Fig. 4.2, we briefly illustrate the MCTS-DMOEA.

The main scheme of the proposed MCTS-DMOEA is presented in Algorithm 5. At the first of the beginning, the population Pop is initialized randomly and clustered by Algorithm 6. After that, a set $Subpops$ containing a number of subpopulations is outputted. If the environment does not change, Algorithm 7 allocates computing resources, i.e., evaluations, according to the performance of each subpopulation in the set $Subpops$, and adaptively selects the evolutionary operator for evolving the subpopulation with the given number of evaluations. When the environment changes, all subpopulations in $Subpops$ are merged and re-clustered to search for the next environment.

In the following subsections, we will introduce the two main components: Algorithm 6 and

¹Although identifying dynamic changes is a crucial part of solving DMOPs, in this work, we solely focus on how to exploit the computing resources once the change has been detected.

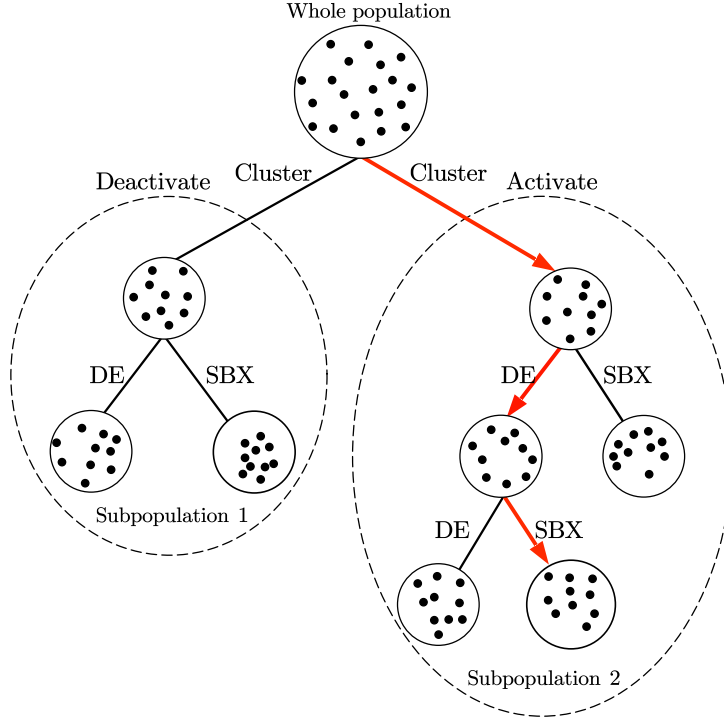


Figure 4.2: The whole population is first divided into several subpopulations. Then, MCTS determines which subpopulation should be activated and the specific evolutionary operator of the corresponding activated subpopulation.

Algorithm 7.

4.2.2 Cluster-Based Subpopulation Partition

In this work, a hierarchical clustering method is designed to partition populations into several subpopulations covering different regions. In particular, according to the characteristics of solutions, the solutions are adaptively clustered, and the number of subpopulations can be automatically determined.

Firstly, given the whole population Pop , each solution in Pop is regarded as one cluster (subpopulation) and these clusters are saved in the set $Subpops$. Then, if there exists a cluster s

Algorithm 5: MCTS-DMOEA**Input:** the DMOP $F_t(\cdot)$ **Output:** the POS_t of the DMOP at the different environments

```

1 Initialize a population  $Pop$  randomly;
2 Set environment step  $t = 0$ ;
3  $SubPops = \text{Clustering}(Pop)$ ;
4 while stopping criterion is not met do
5     if the environment has changed then
6          $t = t + 1$ ;
7         Merge all subpopulations in  $SubPops$  into  $Pop$ ;
8          $SubPops = \text{Clustering}(Pop)$ ;
9     end
10    Set the maximum number of evaluations  $E$  for environment  $t$ ;
11     $POS_t = \text{MCTS Resource Allocation}(SubPops, E)$ ;
12    return  $POS_t$ ;
13 end

```

in $Subpops$, and the number of solutions in the cluster s is less than two, we will find a cluster s_m which is closest to the centroid of s ,

$$s_m = \arg \min_{s_m} \|C(s) - C(s_m)\|_2, \quad (4.1)$$

where $C(s)$ calculates the centroid of a cluster s .

$$C(s) = \frac{\sum_{x \in s} \sum_{i=1}^N x_i}{|s|}, \quad (4.2)$$

where N is the number of decision variables. Next, s is merged with the found s_m , and deleted from $Subpops$. This merge process continues until there is no cluster containing less than two solutions. The clustering strategy enables several subpopulations to be covered in different subareas of the search space, allowing the subsequent resource allocation to specify the evolutionary operators according to the characteristics of the subareas.

It should be noted that, in this process, if the number of clusters is deducted to three and there still exists a cluster s whose number of solutions of s is less than two, we directly merge s with the cluster s_m having the smallest number of solutions in $Subpops$. In this way, we ensure that

each subpopulation has more than one individual. The specific steps are given in the Algorithm 6.

Algorithm 6: Clustering

Input: the whole population Pop

Output: the set of subpopulations $Subpops$

```

1  $Subpops = \emptyset$ ;
2 for each solution  $sol$  in  $Pop$  do
3    $Subpops = Subpops \cup \{sol\}$ ;
4 end
5 while  $\exists s \in Subpops, |s| < 2$  do
6   if  $|Subpops| == 3$  then
7      $s_m = s \cup s_m$ , where  $s_m$  having the smallest number of solutions in  $Subpops \setminus s$ ;
8   end
9    $s_m = s \cup s_m$ , where  $s_m = \arg \min_{s_m} ||C(s) - C(s_m)||_2$ ;
10  Delete  $s$  from  $Subpops$ ;
11 end
12 return  $Subpops$ 

```

4.2.3 Monte Carlo Tree-Assisted Computational Resource Allocation

The designed computational resource allocation uses the tree policy to estimate the performance of the subpopulation by evolving them with different evolutionary operators, and the computational resources are allocated to the most promising subpopulation with specific evolutionary operators. The initial tree of the MCTS consists of two layers: The root node and the leaf nodes. The root of the initial tree is the whole population, and the leaf nodes of the initial tree are the subpopulations clustered by Algorithm 6.

In detail, Monte Carlo tree-assisted Computational Resource Allocation includes four steps: Selection, Expansion, Simulation, and Backpropagation. Fig. 4.3 shows the pipeline of the designed computational resource allocation approach. Firstly, the selection step recursively selects a node based on the node's UCB value until a leaf node is reached. After the selec-

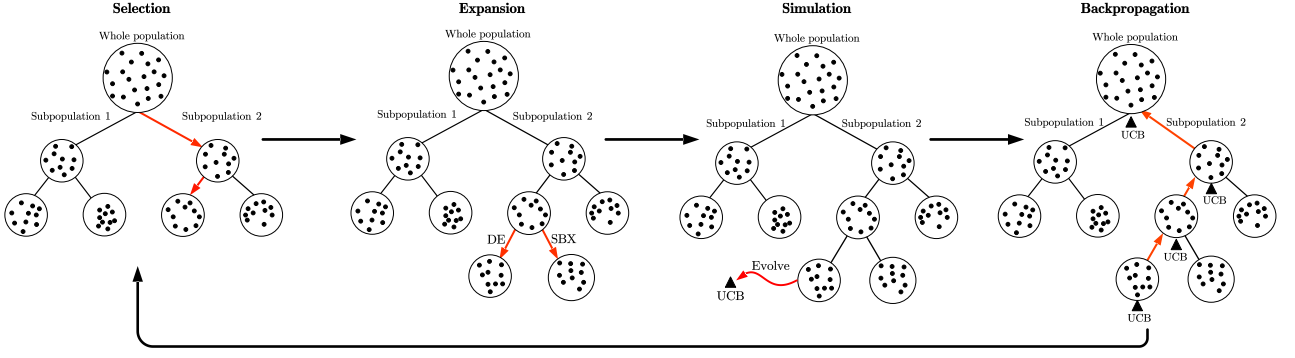


Figure 4.3: Monte Carlo Tree-assisted computational resource allocation. (1) Selection: Recursively, a node is selected according to the UCB value until the selected node is a leaf node. (2) Expansion: The selected leaf node reproduces two children nodes by using SBX and DE operators. (3) Simulation: An offspring node is selected for simulation with a minimum batch of computational resources. (4) Backpropagation: UCB values of nodes along the selection path are updated.

tion process, by using SBX and DE operators, the solutions in the chosen node reproduce two offspring groups with respect to two child nodes. Then, the simulation process aims to estimate the benefit, i.e., UCB value, of the subpopulation (node)’s evolution with the specific evolutionary operator by allocating a limited batch of computational resources. Next, in the backpropagation, the UCB values of nodes along the selection path are updated for the next selection process. The above four steps are iteratively executed until the algorithm terminates. In the following parts, we will introduce the details of the four steps.

(1) Selection: The selection process selects a node recursively according to the UCB value of the node until the selected node is a leaf. Two factors determine the UCB values: the convergence of the node and the spatial size of the node. The convergence of the node $C(node)$ indicates the objective values of the solutions in the node and is calculated as,

$$C(node) = \frac{\sum_{x \in node} \sum_{m=1}^M f_m(x)}{|node|}, \quad (4.3)$$

where $|node|$ means the number of solutions in the node. The spatial size of a node $SP(node)$

represents the diversity of the solutions in this node, and it can be computed as,

$$SP(node) = \max_{x_1, x_2 \in node} ||x_1 - x_2||. \quad (4.4)$$

Subsequently, we respectively normalize SP and C values of the nodes attaching to the same parent,

$$SP(node) = \frac{SP(node)}{SP(node) + \sum_{v \in siblings(node)} SP(v)}, \quad (4.5)$$

$$C(node) = \frac{C(node)}{C(node) + \sum_{v \in siblings(node)} C(v)}, \quad (4.6)$$

where $siblings(node)$ denotes the sibling nodes with the same parent of $node$. The UCB value leveraging convergence value C and diversity value SP is then calculated by,

$$UCB_{CSP}(node) = -C(node) + SP(node). \quad (4.7)$$

According to Eq. (4.7), we can see that one node with lower objective values (i.e., lower $C(node)$) and higher diversity (i.e., higher $SP(node)$) will have larger $UCB(node)$. Compared with conventional UCB, the dedicated UCB_{CSP} leverages convergence and diversity to indicate the potential benefit of solutions, potentially guiding the promising search directions and maintaining a delicate exploration-exploitation of the dynamic environments. The formulation uses diversity to represent uncertain benefits, and there is no need for extensive trials to select nodes to represent the uncertainty.

During the selection strategy, a node with a higher UCB_{CSP} value is more likely to be selected. The probability of the node being selected can be calculated as,

$$P(node) = \frac{e^{UCB_{CSP}(node)}}{e^{UCB_{CSP}(node)} + \sum_{v \in siblings(node)} e^{UCB_{CSP}(v)}}. \quad (4.8)$$

Because the UCB_{CSP} value not only considers the convergence of the node but also the diversity, the selection process can maintain a delicate exploration-exploitation to activate the node (subpopulation).

(2) Expansion: When a leaf node is reached, by performing two evolutionary operators, i.e., SBX and DE, two child nodes are respectively generated to reproduce new offspring solutions. According to the characteristics of the used evolutionary operators, the solutions in the two child nodes have different roles. The offspring node reproduced by the SBX operator aims to exploit the search space, while the offspring node reproduced by the DE operator aims to explore the search space.

(3) Simulation: The simulation process evolves one expanded child node and estimates the UCB_{CSP} value of the node. Firstly, we calculate the spatial size SP for the two expanded child nodes. Afterward, by performing the corresponding evolutionary operator of the expanded child node, one node with higher spatial size values is evolved with a limited batch of computational resources. In this work, the computational resources in this batch (i.e., the number of evaluations) for this evolution are equal to the number of solutions in this node. After evolving, the objective values of the solutions in this node are obtained so that the convergence value C of the node can be calculated by Eq. (4.3). Finally, the UCB value of the expanded node is estimated by Eq. (4.7) to indicate the evolutionary performance of the subpopulation with the specific evolutionary operator.

(4) Backpropagation: After simulation, we update the UCB value of each node on the selection path. Specifically, we directly replace the UCB values with the UCB value of the child node, and the updated UCB value of the node on the path is equal to the UCB value of the expanded node obtained in the simulation process.

The above steps are performed iteratively until the environment changes or the resources of the current environment are exhausted. During the whole tree-policy process, a minimum batch of computational resources is allocated to the promising population and evolutionary operators for searching towards the changing Pareto solutions, and the performance of the evolution is measured by UCB_{CSP} . In such a manner, we can continuously select and evolve the most profitable subpopulations with specific operators by allocating the minimum batch

Chapter 4. Knowledge-Driven Adaptive Resource Allocation for Dynamic Multiobjective Optimization

Algorithm 7: MCTS Resource Allocation

Input: subpopulations *Subpops*, the maximum number of evaluations *E*

Output: the POS at the environment *POS*

```
1 Compute the UCB for each subpopulation in Subpops;
2 Select node with a maximum UCB;
3 // *****Selection*****;
4 while the selected node has children do
5   |  $node = \arg \max_{child \in children(node)} UCB(child)$ ;
6 end
7 // *****Expansion*****;
8 Reproduce child1 by using DE operator to node;
9 Reproduce child2 by using SBX operator to node;
10  $children(node) = child1 \cup child2$ ;
11 // *****Simulation*****;
12 Compute the SP values for child1 and child2;
13 Select child with a maximum SP from child1 and child2;
14 if child1 == child then
15   | Evolve child with DE operator by allocating  $|child|$  evaluations;
16 end
17 else
18   | Evolve child with SBX operator by allocating  $|child|$  evaluations;
19 end
20  $E = E - |child|$ ;
21 Compute the UCB for the select child;
22 // *****Backpropagation*****;
23  $node = parent(child)$ ;
24 while the selected node has parents do
25   | Update the UCB of node by  $UCB(child)$ ;
26   |  $node = parent(child)$ ;
27 end
28 Merge all leaf nodes into Pop;
29  $POS = \text{Non-dominated Sort}(Pop)$ ;
30 if  $E > 0$  or the environment does not change then
31   | Goto Line 1;
32 end
33 else
34   | return POS
35 end
```

of computational resources at one iteration of the tree policy, thus reducing the computational waste on improper subpopulations and evolutionary operators. The details are presented in Algorithm 7.

4.2.4 Complexity Analysis

In this section, we analyze the time complexity and space complexity of the proposed MCTS-assisted resource allocation approach. The selection process and backpropagation process cost $\mathcal{O}(k \cdot \log(I))$ time complexity, where I is the number of iterations for constructing the whole tree and k is the number of branches for the nodes. The expansion process has constant time complexity. As for the simulation, the time complexity is $\mathcal{O}(I \cdot \mathcal{O}(V))$, where $\mathcal{O}(V)$ is the complexity for calculating the UCB values that depend on the specific EAs. Overall, the computational complexity is $\mathcal{O}(k \cdot \log(I) + I \cdot \mathcal{O}(V))$. In each loop of the tree policy, one node with n N -dimensional solutions is generated. Therefore, the space complexity is $\mathcal{O}(n \cdot N \cdot \log(I))$.

4.3 Experiments

In this section, comprehensive numerical experiments have been conducted on a DMOP benchmark containing 14 problems and a real-world application. Moreover, we validated the effectiveness of the proposed Monte Carlo tree-assisted computational in exploiting limited computing resources.

4.3.1 Experimental Settings

Test Problems. All compared algorithms are evaluated based on DF benchmark suite [154]. The DF benchmark suite comprises 14 problems (DF1-DF14). DF1-DF9 are bi-objective problems, and DF10-DF14 are tri-objective problems. The DF benchmark suite has diverse properties nicely representing various real-world scenarios, such as time-dependent POF/POS geometries, irregular POF shapes, disconnectivity, knee points, etc.

Compared Algorithms. We select several representative DMOEAs with respect to the three categories, i.e., diversity-based, memory-based, and prediction-based DMOEAs for comparison.

The algorithms used for comparison in this experiment are as follows:

MDP [32]: A diversity-based DMOEA that clusters the population into representative groups to maintain diversity.

MMTL [35]: A memory-based DMOEA that integrates knowledge transfer, where elite solutions from the historical environment and transferred solutions are both used as the initial population for the new environment.

SVR [37]: SVR stores the locations of the historical solutions, then constructs a time series prediction model to predict the locations of future solutions.

IGP [47]: As a novel inversed prediction model, IGP generates initial solutions by exploiting the information of the objective space rather than the decision space.

TK-DMOEa [53]: As an efficient evolutionary transfer optimization method, TK-DMOEa only uses the knee points for knowledge transfer, then generates the initial knee points.

CT-DMOEa [155]: A clustering-based DMOEA for curbing the negative transfer, where useful knowledge between two similar clusters from the previous environment is transferred to the newly generated environment.

For a fair comparison, most of the parameters of these algorithms are set according to the original references. Other parameters are set as follows:

1. Population size and the number of decision variables: in the comparison, the population size is set to 100 for all problems, respectively. Each benchmark problem has 10 variables. All the problems are run 10 times independently.
2. Dynamic test settings: Considering that the dynamics terms τ and τ_t in Eq. (2.5) are defined by the generations instead of computing resources, e.g., evaluations, we extend the concept of Eq. (2.5) and define τ_t and τ by evaluations for accommodating the scenarios with limited computational resources. That is, τ and τ_t refer to the maximum number

of evaluations and the number of evaluations for each environment step, respectively. In this way, all algorithms consume equivalent evaluations for fair comparison. The modified dynamic setting comprises three pairs of change characteristics, $(n_t = 10, \tau_t = 10 \cdot S)$, $(n_t = 10, \tau_t = 5 \cdot S)$ and $(n_t = 5, \tau_t = 10 \cdot S)$, where S is the size of the population. That is, the maximum number of evaluations E for the environment in Algorithm 5 is τ_t . The total number of evaluations τ is fixed to be $30 \cdot \tau_t$, which ensures there are 30 changes in each run.

3. Evolutionary operators and parameters in the proposed algorithm: SBX Crossover: $p_c = 0.9$, $\eta_c = 2$. Polynomial Mutation: $p_m = 0.1$, $\eta_m = 4$. F and CR in DE: $F = 0.5$, $CR = 0.6$.

It should be noted that we use evaluations instead of generations in the settings of τ_t and τ to ensure that all compared algorithms consume the same computational resources. This is because some algorithms may consume additional evaluations in their designs under the traditional dynamic settings, resulting in unfair comparison.

Performance Indicators. In this study, the following metrics are used to evaluate the performance of the compared algorithms from different perspectives.

- 1) Inverted generational distance (IGD): The definition of the IGD is as follows:

$$\text{IGD}(POF^*, POF) = \frac{1}{n} \sum_{p^* \in POF^*} \min_{p \in POF} \|p^* - p\|^2, \quad (4.9)$$

where POF^* is the true POF of a given multiobjective optimization problem, POF is an approximation set of POF^* obtained by a multiobjective optimization algorithm and n is the number of solutions in the POF^* .

The MIGD metric is a variant of IGD that is defined as the average of the IGD values over

some time steps in a given run:

$$\text{MIGD}(POF^*, POF) = \frac{1}{|T|} \sum_{t \in T} \text{IGD}(POF_t^*, POF_t), \quad (4.10)$$

where T is a set of discrete time points in a run and $|T|$ is the cardinality of T .

2) The hypervolume (HV) is used to quantify the convergence and distribution of solutions obtained by the algorithm. The definition of hypervolume is as follows:

$$\text{HV}(POF, ref) = \Lambda\left(\bigcup_{p \in POF} \{p' | ref \succ p' \succ p\}\right), \quad (4.11)$$

where Λ is the Lebesgue measure and $ref \in R^M$ is the reference point for the computation of HV. MHV metric is a variant of HV that is defined as the average of the HV values over some time steps in a given run.

$$\text{MHV}(POF, ref) = \frac{1}{|T|} \sum_{t \in T} \text{HV}(POF_t, ref). \quad (4.12)$$

4.3.2 Comparisons with Baselines

The statistical results of the average MIGD and MHV values over 10 runs can be found in Tables 4.1 and 4.2, respectively. In the tables, (-), (=), and (+) indicate that MCTS-DMOEAs are statistically significantly worse, indifferent, or significantly better than the compared algorithms, respectively, by using the Wilcoxon test. The significance level is set to 0.05 in these experimental studies.

Table 4.1 presents the results of the comparison between MCTS-DMOEAs and other DMOEAs on DF1~DF14 under different configurations. The results indicate that MCTS-DMOEAs outperform the competing algorithms in terms of MIGD values on DF2, DF3, DF4, DF5, DF6, DF9, DF10, and DF12 across all configurations. However, on DF1, DF7, DF8, DF11, DF13, and DF14, MCTS-DMOEAs perform slightly worse than the corresponding best-performing

Table 4.1: Mean and variance of MIGD values obtained by compared algorithms over DF test functions.

Problem	n_t, τ_t	MCTS-DMOEa	MDP	MMTL	SVR	IGP	TK-DMOEa	CT-DMOEa
DF1	10,10	5.78e-01(1.65e-03)	2.70e-01(7.60e-05)-	6.47e-02(2.24e-05)-	1.25e-01(8.16e-06)-	7.49e-02(9.93e-05)-	9.41e-02(1.19e-04)-	2.61e-02(1.04e-06)-
	5,10	5.92e-01(3.72e-03)	2.67e-01(3.03e-04)-	1.01e-01(5.76e-05)-	1.30e-01(5.29e-05)-	6.52e-02(5.45e-05)-	1.09e-01(3.10e-04)-	3.68e-02(1.18e-06)-
	10,5	5.79e-01(2.65e-03)	2.96e-01(3.10e-04)-	1.15e-01(5.22e-04)-	2.66e-01(3.50e-04)-	1.34e-01(6.75e-05)-	1.44e-01(7.11e-05)-	4.96e-02(1.05e-05)-
DF2	10,10	4.20e-02(6.68e-06)	2.11e-01(1.67e-04)+	6.85e-02(7.82e-05)+	7.82e-02(7.03e-06)+	5.81e-02(2.34e-05)+	7.87e-02(2.48e-05)+	6.46e-02(2.22e-05)+
	5,10	4.36e-02(2.77e-05)	2.32e-01(2.10e-04)+	8.22e-02(6.92e-05)+	8.30e-02(7.13e-06)+	5.91e-02(4.68e-05)+	7.89e-02(3.59e-05)+	7.58e-02(3.59e-05)+
	10,5	4.69e-02(7.07e-05)	2.27e-01(5.59e-05)+	1.01e-01(6.43e-05)+	1.70e-01(8.05e-05)+	1.00e-01(7.46e-05)+	1.11e-01(6.08e-05)+	1.11e-01(3.24e-05)+
DF3	10,10	5.29e-02(5.88e-05)	5.39e-01(2.09e-03)+	3.31e-01(3.39e-04)+	4.57e-01(1.52e-04)+	5.12e-01(2.63e-03)+	3.63e-01(2.84e-04)+	1.11e-01(8.79e-05)+
	5,10	5.17e-02(7.09e-05)	5.38e-01(1.56e-03)+	3.85e-01(6.98e-04)+	4.29e-01(3.63e-04)+	5.33e-01(1.80e-03)+	3.89e-01(3.44e-04)+	1.35e-01(8.43e-05)+
	10,5	5.43e-02(5.40e-05)	6.17e-01(1.27e-03)+	3.57e-01(5.00e-04)+	6.31e-01(3.71e-04)+	5.18e-01(1.09e-03)+	3.95e-01(5.30e-04)+	1.52e-01(2.57e-04)+
DF4	10,10	2.30e-01(3.28e-03)	1.57e+00(6.76e-03)+	9.80e-01(3.47e-05)+	4.40e-01(3.17e-04)+	1.25e+00(1.75e-02)+	1.01e+00(1.94e-04)+	2.67e-01(1.59e-04)=
	5,10	2.27e-01(2.98e-03)	1.56e+00(1.18e-02)+	1.00e+00(1.17e-04)+	4.46e-01(5.06e-04)+	1.19e+00(1.45e-03)+	1.00e+00(3.72e-04)+	2.55e-01(8.78e-05)=
	10,5	2.42e-01(2.40e-03)	1.92e+00(4.58e-03)+	9.88e-01(1.79e-04)+	9.30e-01(2.12e-03)+	1.23e+00(1.57e-02)+	1.03e+00(3.77e-04)+	2.68e-01(4.67e-04)=
DF5	10,10	4.10e-02(2.36e-05)	1.53e+00(6.73e-03)+	1.18e+00(3.95e-04)+	9.98e-01(7.25e-04)+	6.42e-01(2.12e-03)+	1.24e+00(1.00e-03)+	4.16e-02(1.74e-06)=
	5,10	4.27e-02(2.82e-05)	1.44e+00(4.46e-03)+	1.11e+00(3.77e-04)+	1.12e+00(3.40e-03)+	6.16e-01(1.84e-03)+	1.13e+00(8.24e-04)+	5.08e-02(5.36e-06)=
	10,5	4.15e-02(1.16e-05)	1.67e+00(5.62e-03)+	1.25e+00(6.20e-04)+	1.57e+00(2.55e-03)+	7.50e-01(4.30e-03)+	1.32e+00(4.13e-03)+	7.21e-02(1.12e-06)+
DF6	10,10	6.54e-02(1.98e-05)	9.63e+00(6.84e-01)+	2.67e+00(6.85e-02)+	7.92e+00(2.28e-01)+	2.26e+00(3.93e-02)+	1.98e+00(2.63e-01)+	3.28e+00(2.55e+00)+
	5,10	6.45e-02(3.45e-05)	9.08e+00(3.09e-01)+	2.17e+00(2.76e-01)+	7.88e+00(1.52e-01)+	1.41e+00(3.53e-02)+	1.53e+00(3.72e-02)+	1.00e+00(9.56e-01)+
	10,5	7.28e-02(2.21e-05)	1.07e+01(8.79e-01)+	3.17e+00(1.25e-01)+	1.06e+01(4.43e-01)+	3.27e+00(1.23e-01)+	2.71e+00(1.79e-01)+	4.28e+00(4.07e+00)+
DF7	10,10	1.02e+00(2.65e-01)	2.54e+00(5.37e-02)+	2.26e+00(1.33e-03)+	6.92e-01(1.56e-03)+	5.07e-01(1.36e-03)-	2.33e+00(1.79e-03)+	3.47e+01(1.93e+02)+
	5,10	1.09e+00(9.27e-03)	4.17e+00(1.13e-01)+	3.62e+00(1.92e-02)+	9.49e-01(2.64e-03)=	6.10e-01(4.51e-03)-	3.89e+00(1.04e-02)+	3.60e+01(5.96e+01)+
	10,5	1.01e+00(2.70e-01)	4.22e+00(6.27e-02)+	2.29e+00(5.81e-04)+	1.40e+00(1.46e-02)+	6.62e-01(3.11e-04)-	2.45e+00(1.17e-02)+	4.08e+01(2.10e+02)+
DF8	10,10	5.47e-01(4.54e-03)	4.21e-01(3.75e-04)-	9.81e-01(2.27e-04)+	5.61e-01(2.31e-04)=	1.03e+00(7.22e-04)+	1.06e+00(7.12e-04)+	9.04e-01(1.68e-04)+
	5,10	5.18e-01(3.97e-03)	4.02e-01(2.03e-04)-	9.50e-01(3.73e-04)+	5.75e-01(1.42e-04)=	1.01e+00(4.26e-04)+	1.02e+00(8.92e-04)+	9.04e-01(3.29e-04)+
	10,5	5.58e-01(5.10e-03)	2.34e-01(4.70e-05)-	9.34e-01(1.89e-03)+	4.80e-01(2.07e-04)-	9.90e-01(7.33e-04)+	1.03e+00(5.29e-04)+	8.41e-01(1.03e-04)+
DF9	10,10	6.50e-01(2.88e-03)	1.75e+00(8.76e-03)+	2.08e+00(6.53e-04)+	1.83e+00(3.03e-03)+	1.61e+00(3.26e-03)+	2.13e+00(5.37e-03)+	1.44e+00(3.21e-03)+
	5,10	6.42e-01(2.39e-03)	1.79e+00(1.02e-02)+	2.10e+00(9.32e-03)+	1.84e+00(6.63e-03)+	1.65e+00(2.10e-03)+	2.10e+00(6.16e-03)+	1.25e+00(7.80e-03)+
	10,5	6.61e-01(5.39e-03)	1.67e+00(1.42e-02)+	2.07e+00(9.01e-03)+	2.16e+00(4.57e-03)+	1.69e+00(4.77e-03)+	2.07e+00(7.32e-03)+	1.31e+00(3.62e-03)+
DF10	10,10	1.75e-01(3.26e-05)	3.39e-01(6.62e-05)+	2.04e-01(7.50e-05)+	2.21e-01(2.30e-05)+	2.64e-01(2.31e-04)+	3.26e-01(2.41e-04)+	2.59e-01(3.74e-04)+
	5,10	1.64e-01(6.48e-06)	3.02e-01(8.86e-05)+	1.83e-01(1.37e-05)=	1.92e-01(1.46e-05)+	2.35e-01(3.51e-05)+	2.85e-01(2.86e-04)+	2.37e-01(2.98e-04)+
	10,5	1.64e-01(4.52e-05)	3.72e-01(9.18e-05)+	2.22e-01(8.04e-05)+	2.71e-01(2.25e-05)+	3.03e-01(2.09e-04)+	3.51e-01(3.54e-04)+	2.22e-01(7.08e-04)+
DF11	10,10	6.33e-01(2.42e-03)	2.08e-01(8.76e-04)-	1.47e-01(1.93e-05)-	2.01e-01(3.70e-05)-	1.89e-01(1.22e-04)-	1.61e-01(4.66e-05)-	1.32e-01(9.32e-07)-
	5,10	6.41e-01(5.63e-04)	2.27e-01(1.46e-03)-	1.50e-01(8.53e-06)-	2.57e-01(3.08e-05)-	1.82e-01(1.18e-04)-	1.57e-01(5.15e-05)-	1.35e-01(5.09e-06)-
	10,5	6.60e-01(3.59e-04)	3.48e-01(4.22e-04)-	1.62e-01(2.43e-05)-	3.45e-01(2.79e-05)-	2.34e-01(4.92e-05)-	1.87e-01(1.87e-05)-	1.43e-01(3.20e-06)-
DF12	10,10	2.44e-01(2.20e-03)	3.19e-01(5.54e-05)+	1.10e+00(2.64e-03)+	5.04e-01(5.56e-05)+	1.08e+00(1.03e-03)+	1.15e+00(2.67e-04)+	7.48e-01(2.09e-06)+
	5,10	2.50e-01(1.94e-03)	3.51e-01(1.37e-04)+	1.11e+00(4.53e-04)+	4.31e-01(1.18e-04)+	9.82e-01(1.75e-03)+	1.02e+00(1.51e-03)+	7.53e-01(2.97e-06)+
	10,5	2.47e-01(2.89e-03)	2.81e-01(1.54e-05)+	1.07e+00(1.97e-03)+	3.59e-01(4.07e-05)+	1.07e+00(1.19e-03)+	1.13e+00(2.47e-04)+	7.57e-01(2.55e-06)+
DF13	10,10	1.41e+00(4.47e-03)	2.72e+00(2.50e-02)+	1.68e+00(7.96e-04)+	1.16e+00(3.91e-03)-	1.44e+00(2.32e-02)=	1.80e+00(1.95e-03)+	3.21e-01(2.67e-05)-
	5,10	1.35e+00(2.47e-03)	2.44e+00(1.56e-02)+	1.58e+00(4.18e-04)+	1.26e+00(1.58e-02)=	1.38e+00(1.17e-02)=	1.66e+00(2.64e-03)+	3.26e-01(2.58e-06)-
	10,5	1.41e+00(7.67e-03)	2.96e+00(1.18e-02)+	1.71e+00(2.90e-04)+	2.04e+00(1.15e-02)+	1.50e+00(3.31e-02)=	1.92e+00(2.59e-03)+	2.84e-01(4.02e-05)-
DF14	10,10	5.94e-01(2.68e-03)	1.38e+00(3.60e-03)+	8.12e-01(1.03e-04)+	5.82e-01(5.80e-03)=	7.04e-01(6.52e-02)+	8.38e-01(2.94e-04)+	8.93e-02(1.62e-06)-
	5,10	5.96e-01(5.65e-03)	1.25e+00(2.20e-03)+	7.57e-01(1.36e-04)+	6.56e-01(6.22e-03)+	6.38e-01(1.14e-02)+	7.73e-01(1.86e-04)+	9.41e-02(2.30e-06)-
	10,5	5.81e-01(5.09e-03)	1.27e+00(1.46e-03)+	8.37e-01(1.19e-04)+	9.58e-01(3.13e-03)+	6.84e-01(3.89e-03)+	8.87e-01(9.29e-04)+	9.56e-02(1.47e-06)-
+/-		33/0/9	35/1/6	28/5/9	30/3/9	36/0/6	25/5/12	

Table 4.2: Mean and variance of MHV values obtained by compared algorithms over DF test functions

Problem	n_t, τ_t	MCTS-DMOE	MDP	MMTL	SVR	IGP	TK-DMOE	CT-DMOE
DF1	10,10	1.34e-01(8.77e-05)	3.02e-01(1.26e-04)-	5.37e-01(6.23e-05)-	4.59e-01(1.54e-05)-	5.32e-01(9.05e-05)-	5.05e-01(1.32e-04)-	5.95e-01(2.97e-06)-
	5,10	1.22e-01(9.85e-04)	3.06e-01(2.29e-04)-	4.92e-01(7.18e-05)-	4.53e-01(2.73e-05)-	5.39e-01(4.21e-05)-	4.89e-01(1.92e-04)-	5.79e-01(3.21e-06)-
	10,5	1.19e-01(2.71e-04)	2.76e-01(4.26e-04)-	4.70e-01(5.36e-04)-	3.21e-01(6.82e-05)-	4.45e-01(1.46e-04)-	4.40e-01(2.50e-05)-	5.56e-01(3.15e-05)-
DF2	10,10	8.16e-01(1.15e-05)	5.13e-01(3.61e-04)+	7.63e-01(1.42e-04)+	7.37e-01(9.32e-06)+	7.72e-01(3.24e-05)+	7.46e-01(6.23e-05)+	8.03e-01(1.61e-05)=
	5,10	8.14e-01(6.94e-05)	4.88e-01(3.69e-04)+	7.35e-01(1.23e-04)+	7.30e-01(1.81e-05)+	7.75e-01(5.20e-05)+	8.03e-01(2.92e-05)=	8.03e-01(2.92e-05)=
	10,5	8.08e-01(7.73e-06)	4.96e-01(9.93e-05)+	7.03e-01(1.31e-04)+	6.00e-01(7.46e-05)+	6.92e-01(1.63e-04)+	6.98e-01(1.02e-04)+	7.81e-01(1.10e-05)+
DF3	10,10	5.35e-01(1.03e-04)	1.81e-01(1.88e-04)+	2.79e-01(1.00e-04)+	1.60e-01(3.92e-05)+	1.98e-01(1.20e-04)+	2.46e-01(1.07e-04)+	4.43e-01(6.43e-05)+
	5,10	5.34e-01(1.33e-04)	1.65e-01(4.85e-04)+	2.43e-01(1.59e-04)+	1.78e-01(5.49e-05)+	1.89e-01(1.65e-04)+	2.31e-01(9.40e-05)+	4.25e-01(7.58e-05)+
	10,5	5.35e-01(1.26e-04)	1.29e-01(1.75e-04)+	2.50e-01(1.36e-04)+	1.01e-01(4.36e-05)+	1.84e-01(4.12e-05)+	2.18e-01(1.05e-04)+	3.93e-01(7.40e-05)+
DF4	10,10	5.58e+00(4.56e-02)	1.48e+00(1.42e-01)+	3.20e+00(8.30e-04)+	4.50e+00(3.90e-03)+	2.53e+00(8.07e-02)+	3.10e+00(3.59e-03)+	5.12e+00(6.41e-05)+
	5,10	5.69e+00(2.77e-02)	1.60e+00(4.97e-02)+	3.18e+00(2.31e-03)+	4.72e+00(6.76e-03)+	2.65e+00(2.53e-02)+	3.16e+00(5.70e-03)+	4.97e+00(3.18e-05)+
	10,5	5.56e+00(2.37e-02)	8.60e-01(6.67e-02)+	3.14e+00(3.19e-03)+	2.90e+00(1.94e-02)+	2.54e+00(4.51e-02)+	2.99e+00(2.69e-03)+	5.01e+00(8.04e-04)+
DF5	10,10	6.36e-01(5.40e-05)	2.35e-01(7.56e-05)+	4.13e-01(8.89e-05)+	3.22e-01(2.75e-05)+	1.97e-01(1.04e-04)+	4.06e-01(1.90e-05)+	5.70e-01(8.61e-07)+
	5,10	6.35e-01(6.41e-05)	2.53e-01(6.96e-05)+	4.03e-01(1.74e-04)+	2.98e-01(3.00e-05)+	2.06e-01(7.76e-05)+	4.14e-01(8.50e-05)+	5.64e-01(7.44e-06)+
	10,5	6.35e-01(2.16e-05)	2.18e-01(2.83e-04)+	3.73e-01(7.28e-05)+	1.98e-01(4.46e-05)+	1.46e-01(1.77e-04)+	3.59e-01(1.48e-04)+	5.41e-01(1.54e-06)+
DF6	10,10	9.13e-01(6.77e-06)	3.59e-02(9.46e-04)+	1.24e-01(1.35e-03)+	00(00)+	2.21e-01(1.57e-03)+	2.26e-01(2.13e-03)+	5.65e-01(1.79e-04)+
	5,10	8.91e-01(4.32e-06)	1.56e-02(1.71e-04)+	1.21e-01(1.13e-03)+	1.13e-04(1.01e-07)+	2.12e-01(1.11e-03)+	1.59e-01(4.43e-04)+	5.64e-01(3.17e-03)+
	10,5	9.11e-01(2.44e-06)	2.00e-02(2.69e-04)+	1.26e-01(1.02e-03)+	00(00)+	1.12e-01(1.16e-03)+	1.27e-01(7.63e-04)+	4.88e-01(1.79e-04)+
DF7	10,10	2.90e+00(3.58e+01)	1.92e+00(4.63e-03)+	2.02e+00(3.19e-03)+	2.47e+00(5.77e-04)+	2.21e+00(2.31e-04)+	2.04e+00(1.92e-03)+	4.86e-03(7.38e-06)+
	5,10	5.98e+00(3.25e+01)	2.65e+00(7.53e-03)+	2.60e+00(6.34e-03)+	3.23e+00(3.83e-04)+	2.89e+00(9.91e-04)+	2.48e+00(8.14e-02)+	2.95e-01(2.51e-02)+
	10,5	2.90e+00(3.21e+00)	1.52e+00(4.48e-03)+	1.97e+00(3.45e-03)+	2.15e+00(5.02e-03)+	2.23e+00(1.60e-03)+	1.95e+00(7.76e-03)+	3.21e-03(1.03e-05)+
DF8	10,10	4.33e+01(3.23e+00)	4.61e+01(6.56e-03)=	4.55e+01(3.72e-02)=	4.60e+01(4.28e-02)=	4.59e+01(4.28e-02)=	4.56e+01(4.86e-02)=	4.36e+01(1.38e-03)=
	5,10	4.04e+01(2.25e+00)	4.31e+01(9.69e-04)-	4.24e+01(7.80e-02)=	4.31e+01(1.83e-03)=	4.28e+01(4.70e-02)=	4.20e+01(2.48e+00)=	4.83e+01(4.55e-02)=
	10,5	4.30e+01(3.39e+00)	4.58e+01(3.02e-03)=	4.55e+01(3.35e-02)=	4.56e+01(1.04e-03)=	4.57e+01(7.10e-02)=	4.56e+01(6.15e-01)=	4.37e+01(3.13e-04)=
DF9	10,10	2.58e+01(1.00e-02)	1.69e+01(3.11e-01)+	1.63e+01(3.38e-01)+	1.88e+01(2.67e-02)+	1.85e+01(7.14e-02)+	1.80e+01(4.66e-02)+	2.14e+01(4.86e-02)+
	5,10	2.45e+01(1.90e-02)	1.52e+01(2.86e-01)+	1.52e+01(2.66e-01)+	1.82e+01(1.52e-02)+	1.71e+01(7.77e-02)+	1.65e+01(8.49e-02)+	2.00e+01(3.36e-02)+
	10,5	2.58e+01(1.63e-02)	1.61e+01(6.81e-01)+	1.61e+01(4.24e-01)+	1.75e+01(3.64e-02)+	1.82e+01(1.22e-01)+	1.78e+01(4.65e-02)+	2.13e+01(5.12e-02)+
DF10	10,10	7.50e-01(3.39e-03)	4.60e-01(8.24e-05)+	7.11e-01(7.97e-05)+	5.42e-01(2.49e-05)+	6.04e-01(8.64e-05)+	6.46e-01(5.90e-04)+	7.13e-01(5.25e-04)=
	5,10	7.83e-01(1.24e-03)	5.88e-01(7.04e-05)+	8.05e-01(4.82e-05)=	7.02e-01(2.60e-05)+	7.17e-01(7.19e-05)+	7.62e-01(5.15e-04)=	7.35e-01(4.09e-05)=
	10,5	7.81e-01(3.42e-03)	4.07e-01(2.84e-04)+	6.71e-01(7.36e-05)+	4.79e-01(1.64e-05)+	5.31e-01(1.67e-04)+	5.93e-01(4.18e-04)+	7.32e-01(3.25e-04)=
DF11	10,10	2.54e-01(1.55e-02)	6.64e-01(2.16e-03)-	7.62e-01(8.47e-05)-	5.97e-01(2.53e-04)-	6.12e-01(6.58e-04)-	7.16e-01(1.02e-04)-	8.10e-01(1.86e-05)-
	5,10	3.25e-01(3.86e-02)	6.32e-01(2.26e-03)-	7.59e-01(2.41e-05)-	5.96e-01(2.69e-04)-	6.19e-01(8.00e-04)-	7.15e-01(3.76e-04)-	7.94e-01(2.01e-05)-
	10,5	2.32e-01(2.82e-02)	4.04e-01(4.95e-04)-	7.22e-01(1.29e-04)-	3.36e-01(1.49e-04)-	4.97e-01(2.86e-04)-	6.54e-01(5.39e-04)-	7.74e-01(2.10e-05)-
DF12	10,10	9.80e+00(3.43e-01)	9.25e+00(6.91e-04)+	9.38e+00(00)+	9.33e+00(9.72e-05)+	9.28e+00(3.89e-03)+	9.38e+00(1.85e-04)+	4.19e+00(8.94e-05)+
	5,10	9.27e+00(3.28e-01)	8.55e+00(2.04e-02)+	8.95e+00(00)+	8.51e+00(3.71e-03)+	8.52e+00(1.33e-02)+	8.54e+00(1.44e-02)+	3.88e+00(3.99e-04)+
	10,5	9.89e+00(7.81e-02)	8.97e+00(1.37e-02)+	9.38e+00(6.63e-08)+	9.11e+00(7.10e-04)+	9.24e+00(6.65e-03)+	9.38e+00(4.74e-05)+	4.10e+00(3.43e-04)+
DF13	10,10	1.99e-01(1.71e-02)	9.87e-02(1.37e-05)+	1.26e-01(9.39e-06)+	1.18e-01(9.12e-06)+	1.21e-01(5.73e-06)+	1.23e-01(1.16e-05)+	2.66e+00(1.87e-03)-
	5,10	2.92e-01(5.52e-03)	1.01e-01(1.04e-05)+	1.31e-01(1.08e-05)+	1.13e-01(1.70e-05)+	1.25e-01(1.25e-05)+	1.27e-01(1.25e-05)+	2.67e+00(6.16e-04)-
	10,5	2.75e-01(9.30e-03)	9.05e-02(1.07e-05)+	1.14e-01(2.79e-05)+	8.88e-02(3.18e-06)+	1.03e-01(1.96e-05)+	1.05e-01(1.19e-05)+	2.60e+00(6.58e-04)-
DF14	10,10	2.86e-02(1.87e-04)	9.42e-02(2.89e-05)-	2.24e-01(4.80e-06)-	1.53e-01(7.90e-05)-	2.05e-01(5.50e-05)-	2.19e-01(1.21e-05)-	3.59e-01(2.36e-06)-
	5,10	2.89e-02(1.08e-04)	1.14e-01(8.66e-05)-	2.16e-01(1.12e-05)-	1.29e-01(3.94e-05)-	2.07e-01(1.98e-05)-	2.11e-01(1.98e-05)-	3.72e-01(1.01e-05)-
	10,5	3.30e-02(3.57e-04)	1.18e-01(1.28e-05)-	2.11e-01(1.80e-05)-	9.55e-02(4.05e-05)-	1.79e-01(1.10e-04)-	2.00e-01(3.39e-05)-	3.53e-01(1.59e-06)-
+/-			30/2/10	29/4/9	30/3/9	29/4/9	29/4/9	22/8/12

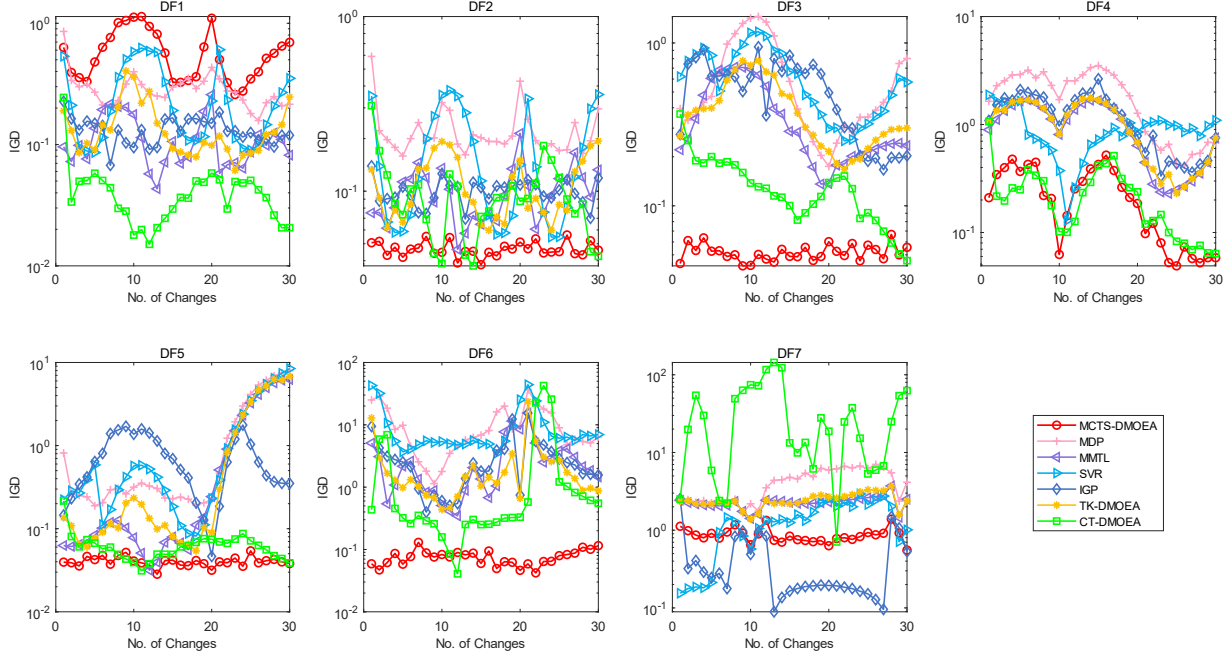


Figure 4.4: Convergence traces of average MIGD values obtained by competing algorithms under test problems DF1-DF7 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).

algorithms. Overall, these results suggest that while MCTS-DMOEa is not always the top-performing algorithm on every test problem, it still exhibits promising performance in many cases and may be a viable option for solving DMOPs.

We plot the MIGD values obtained by various compared algorithms on DF1-DF14 problems in Fig. 4.4 and Fig. 4.5. The convergence traces for MCTS-DMOEa are often located at the bottom of the graph, indicating better performance than other algorithms in most cases. In addition, the curves generated by MCTS-DMOEa exhibit a smoother trend, suggesting improved stability over the course of the optimization process. Taken together, these results demonstrate that MCTS-DMOEa not only outperforms other algorithms but also shows stable optimization behavior.

Compared to the MIGD metric, the MHV metric is considered a more comprehensive metric of

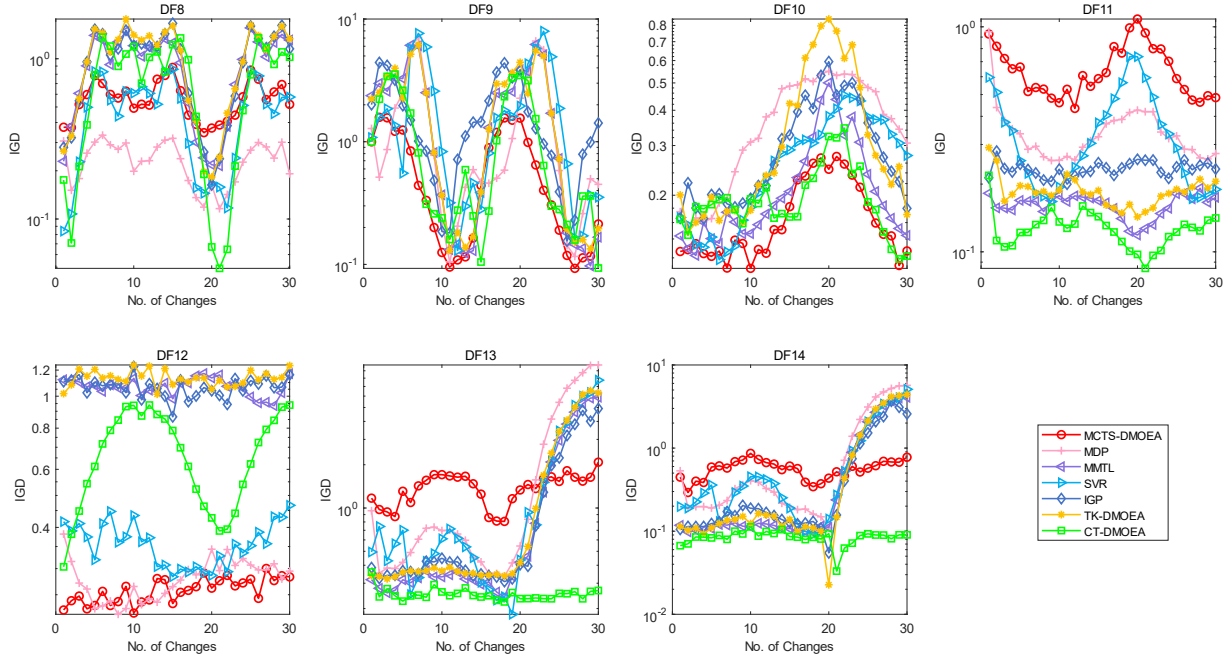


Figure 4.5: Convergence traces of average MIGD values obtained by competing algorithms under test problems DF8-DF14 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).

algorithm performance, as the MHV metric measures both convergence and diversity. Table 4.2 reports the experimental results of the MHV metric. The results indicate that MCTS-DMOEa outperforms the other six compared algorithms in 26 out of 42 test cases, demonstrating its superior performance across a wide range of conditions. Specifically, MCTS-DMOEa excels on DF2, DF3, DF4, DF5, DF6, DF7, DF9 and DF12 under all dynamic test configurations. These results suggest the superiority of the proposed MCTS-DMOEa in achieving both algorithmic convergence and diversity.

The experimental results also reveal that MCTS-DMOEa encounters difficulty in solving DF1, DF7, and DF8 problems. The complexities of the mixed convexity-concavity search spaces in DF1 and DF8 make it challenging for MCTS-DMOEa to balance exploration and exploitation effectively. As for DF7, the presence of local optima leads to some computing resources being allocated to subpopulations trapping into local optima.

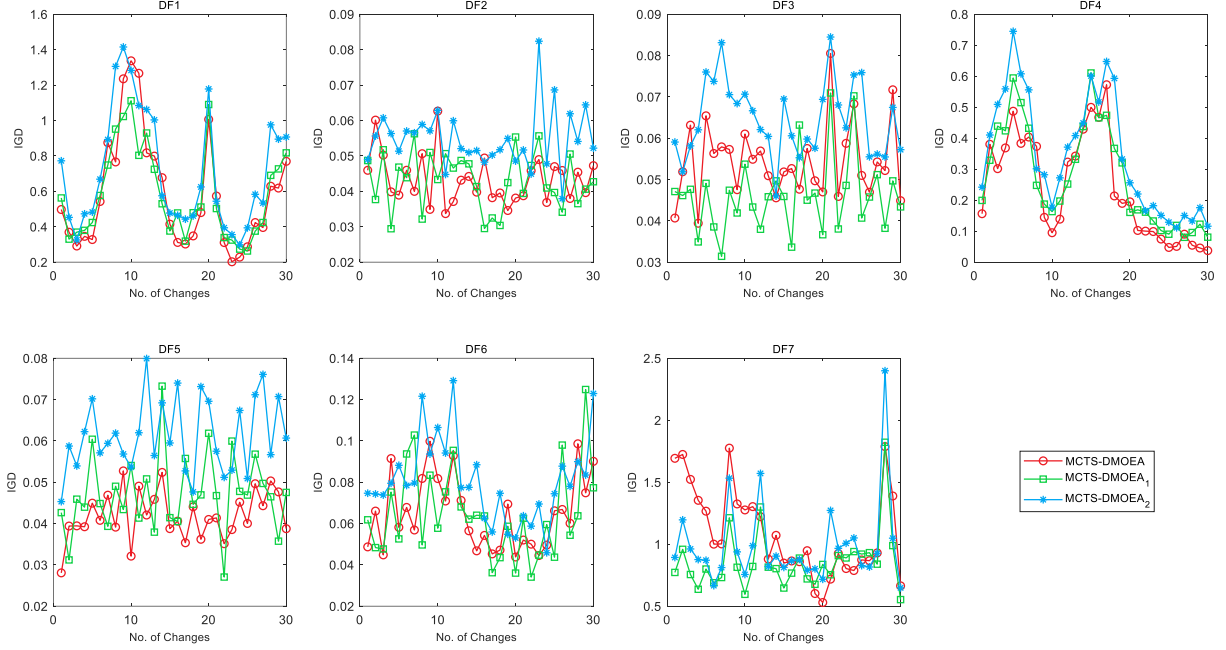


Figure 4.6: Convergence traces of average MIGD values obtained by MCTS-DMOEa, MCTS-DMOEa₁ and MCTS-DMOEa₂ under test problems DF1-DF7 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).

4.3.3 Sensitivity Studies

In this section, we aim to assess the effectiveness of the proposed MCTS-DMOEa from two distinct aspects. Firstly, our goal is to confirm whether the MCTS-DMOEa can leverage the UCB value effectively to select a promising subpopulation to balance the exploration-exploitation trade-off. Secondly, we will investigate whether the MCTS-DMOEa can adjust the evolutionary operator adaptively to facilitate an efficient evolutionary search. The experimental studies conducted will enable us to draw conclusions about the superior ability of the proposed MCTS-DMOEa in exploiting computing resources.

One of the characteristics of MCTS-DMOEa is to adaptively activate subpopulations according to SP value and C value to maintain the exploration-exploitation trade-off. In order to verify

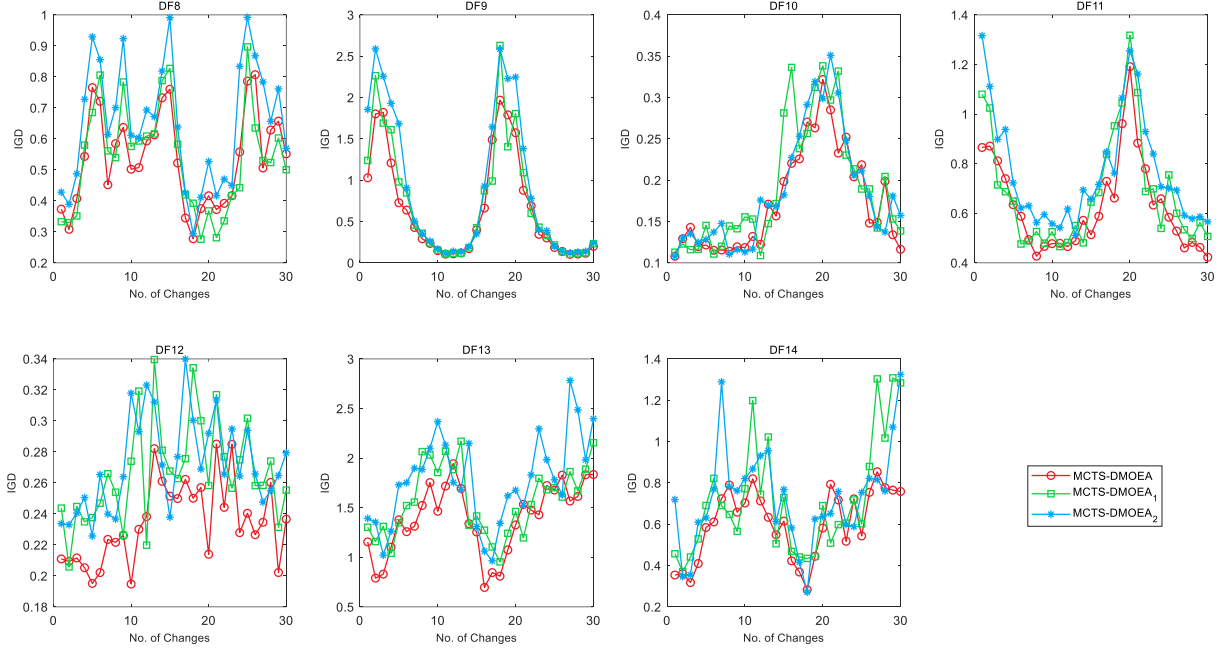


Figure 4.7: Convergence traces of average MIGD values obtained by MCTS-DMOEa, MCTS-DMOEa₁ and MCTS-DMOEa₂ under test problems DF8-DF14 with dynamic test setting ($n_t = 10, \tau_t = 10 \cdot S$).

the effectiveness of each factor in UCB , we ablated the Eq. (4.7) as

$$UCB(node) = -C(node), \quad (4.13)$$

$$UCB(node) = SP(node). \quad (4.14)$$

The ablated version of MCTS-DMOEa algorithms adopted Eq. (4.13) and Eq. (4.14) are renamed as MCTS-DMOEa₁ and MCTS-DMOEa₂, respectively.

In Fig. 4.6 and Fig. 4.7, we compare the MIGD values of MCTS-DMOEa, MCTS-DMOEa₁, and MCTS-DMOEa₂ on DF1-DF14 problems. Although the difference between MCTS-DMOEa and MCTS-DMOEa₁ is insignificant, MCTS-DMOEa produces more favorable convergence profiles. This is because the SP value can maintain diversity and help the population escape from local optima when the environment changes.

Conversely, MCTS-DMOE_A₂ performs poorly compared to the other two algorithms, indicating the importance of the C value for guiding the convergence direction. Overall, these results provide strong evidence for the effectiveness of using both SP and C values to balance exploration and exploitation when searching the solution space.

To evaluate the MCTS-DMOE_A's capability in adjusting the evolutionary operator for efficient evolutionary search, we created three ablation versions of the approach for comparison: MCTS-DMOE_A-SBX, MCTS-DMOE_A-DE, and MCTS-DMOE_A-Rand. The MCTS-DMOE_A-SBX and MCTS-DMOE_A-DE only utilize the SBX operator and DE operator, respectively, to generate child nodes. On the other hand, the MCTS-DMOE_A-Rand selects the SBX or DE operator randomly to generate child nodes. We present the experimental results of the average MIGD and MHV values in Tables 4.3 and 4.4.

The results presented in Tables 4.3 and 4.4 indicate that MCTS-DMOE_A-SBX achieves comparable performance to MCTS-DMOE_A under DF8, DF13, and DF14, while MCTS-DMOE_A-DE can achieve comparable performance under DF2, DF3, DF4, and DF6. This observation confirms that different evolutionary operators with distinct search biases exhibit varying search efficiency for DMOPs with diverse characteristics. Moreover, the adaptability of MCTS-DMOE_A to adjust evolutionary operators enables it to perform only slightly worse than MCTS-DMOE_A-DE and MCTS-DMOE_A-SBX in rare cases. In addition, the proposed MCTS-DMOE_A outperforms MCTS-DMOE_A-Rand in most instances. Taking the above analysis together, these findings confirm the effectiveness of the proposed MCTS-DMOE_A in adaptively adjusting the evolutionary operator for efficient evolutionary search.

Furthermore, to analyze the efficiency of different evolutionary operators in solving DMOPs, we investigate the number of allocated computing resources for different evolutionary operators. Fig. 4.8 displays the percentage allocation of computing resources, i.e., evaluations, to the SBX and DE operators by MCTS-DMOE_A. It can be observed that MCTS-DMOE_A tends to allocate a greater proportion of computing resources to the DE operator when dealing with

Chapter 4. Knowledge-Driven Adaptive Resource Allocation for Dynamic Multiobjective Optimization

Table 4.3: Mean and variance of MIGD values obtained by different ablation algorithms over DF test functions.

Problem	n_t, τ_t	MCTS-DMOEA	MCTS-DMOEA-SBX	MCTS-DMOEA-DE	MCTS-DMOEA-Rand
DF1	10,10	5.78e-01(1.65e-03)	6.27e-01(3.08e-04)=	5.60e-01(4.38e-04)=	5.80e-01(2.61e-03)=
	5,10	5.92e-01(3.72e-03)	6.52e-01(3.81e-03)=	5.77e-01(7.91e-04)=	6.08e-01(4.84e-03)=
	10,5	5.79e-01(2.65e-03)	6.69e-01(1.04e-04)+	5.98e-01(2.28e-03)=	6.08e-01(3.95e-03)=
DF2	10,10	4.20e-02(6.68e-06)	3.67e-01(2.52e-02)+	3.33e-02(2.05e-04)=	1.86e-01(5.26e-02)+
	5,10	4.36e-02(2.77e-05)	3.83e-01(2.85e-02)+	3.37e-02(1.91e-04)=	1.59e-01(3.36e-02)+
	10,5	4.69e-02(7.07e-06)	3.45e-01(2.22e-02)+	3.69e-02(1.75e-04)=	1.73e-01(4.27e-02)+
DF3	10,10	5.29e-02(5.88e-05)	4.43e-01(3.50e-02)+	3.64e-02(4.21e-04)=	2.02e-01(5.97e-02)+
	5,10	5.17e-02(7.09e-05)	4.22e-01(3.20e-02)+	3.51e-02(3.33e-04)=	1.89e-01(5.00e-02)+
	10,5	5.43e-02(5.40e-05)	4.41e-01(3.51e-02)+	3.73e-02(3.58e-04)=	1.93e-01(5.27e-02)+
DF4	10,10	2.30e-01(3.28e-03)	1.75e+00(4.88e-01)+	2.31e-01(6.36e-03)=	8.46e-01(8.54e-01)+
	5,10	2.27e-01(2.98e-03)	1.82e+00(5.56e-01)+	2.13e-01(5.59e-03)=	8.51e-01(8.75e-01)+
	10,5	2.42e-01(2.40e-03)	1.87e+00(5.89e-01)+	2.41e-01(5.51e-03)=	8.65e-01(9.61e-01)+
DF5	10,10	4.10e-02(2.36e-05)	6.15e-01(7.81e-02)+	3.68e-02(3.61e-04)=	2.35e-01(8.96e-02)+
	5,10	4.27e-02(2.82e-05)	6.06e-01(7.28e-02)+	3.58e-02(3.11e-04)=	2.66e-01(1.20e-01)+
	10,5	4.15e-02(1.16e-05)	6.47e-01(8.42e-02)+	3.80e-02(3.74e-04)=	2.67e-01(1.18e-01)+
DF6	10,10	6.54e-02(1.98e-05)	1.02e+01(2.54e+01)+	4.71e-02(1.90e-04)=	4.28e+00(4.29e+01)+
	5,10	6.45e-02(3.45e-05)	1.01e+01(2.71e+01)+	4.53e-02(1.35e-04)-	4.31e+00(4.35e+01)+
	10,5	7.28e-02(2.21e-05)	1.14e+01(3.13e+01)+	5.04e-02(2.53e-04)=	4.84e+00(5.49e+01)+
DF7	10,10	1.02e+00(2.65e-01)	1.44e+01(4.99e+01)+	1.52e+00(1.84e-01)=	6.12e+00(5.17e+01)+
	5,10	1.09e+00(9.27e-03)	1.51e+01(5.17e+01)+	1.24e+00(1.29e-02)=	5.92e+00(5.32e+01)+
	10,5	1.01e+00(2.70e-01)	1.78e+01(6.84e+01)+	1.54e+00(1.82e-01)=	7.96e+00(9.75e+01)+
DF8	10,10	5.47e-01(4.54e-03)	4.65e-01(3.59e-04)=	6.37e-01(6.86e-03)+	4.62e-01(4.30e-04)-
	5,10	5.18e-01(3.97e-03)	4.52e-01(2.87e-04)=	6.29e-01(7.43e-03)+	4.57e-01(3.62e-04)-
	10,5	5.58e-01(5.10e-03)	4.57e-01(3.02e-04)-	6.35e-01(6.50e-03)=	4.80e-01(3.40e-04)-
DF9	10,10	6.50e-01(2.88e-03)	7.79e-01(2.07e-03)+	1.25e+00(2.07e-01)+	7.42e-01(3.91e-04)+
	5,10	6.42e-01(2.39e-03)	7.96e-01(2.24e-03)+	1.25e+00(1.94e-01)+	7.05e-01(3.27e-03)=
	10,5	6.61e-01(5.39e-03)	8.24e-01(2.12e-03)+	1.26e+00(2.09e-01)=	7.17e-01(7.44e-03)=
DF10	10,10	1.75e-01(3.26e-05)	1.84e-01(7.25e-04)=	4.36e-01(2.20e-02)+	2.23e-01(1.98e-03)=
	5,10	1.64e-01(6.48e-06)	1.71e-01(4.07e-04)=	4.24e-01(2.45e-02)+	2.13e-01(2.69e-03)=
	10,5	1.64e-01(4.52e-05)	1.88e-01(6.89e-04)+	4.54e-01(2.79e-02)+	2.28e-01(2.14e-03)+
DF11	10,10	6.33e-01(2.42e-03)	5.46e-01(1.22e-03)-	6.28e-01(9.66e-04)=	5.69e-01(1.41e-03)-
	5,10	6.41e-01(5.63e-04)	5.63e-01(1.89e-03)-	6.68e-01(1.11e-03)=	5.84e-01(2.71e-03)-
	10,5	6.60e-01(3.59e-04)	5.83e-01(6.35e-04)-	6.49e-01(4.97e-04)=	5.89e-01(6.26e-04)-
DF12	10,10	2.44e-01(2.20e-03)	2.53e-01(1.19e-04)=	3.80e-01(3.27e-03)+	2.57e-01(4.86e-04)=
	5,10	2.50e-01(1.94e-03)	2.47e-01(1.48e-04)=	3.74e-01(2.55e-03)+	2.56e-01(4.43e-04)=
	10,5	2.47e-01(2.89e-03)	2.53e-01(3.72e-04)=	3.91e-01(2.83e-03)+	2.73e-01(1.39e-03)=
DF13	10,10	1.41e+00(4.47e-03)	1.16e+00(4.99e-03)-	1.83e+00(1.22e-01)=	1.33e+00(2.08e-02)=
	5,10	1.35e+00(2.47e-03)	1.19e+00(9.32e-03)-	1.70e+00(8.08e-02)=	1.33e+00(1.78e-02)=
	10,5	1.41e+00(7.67e-03)	1.18e+00(1.29e-02)-	1.96e+00(1.72e-01)+	1.30e+00(4.23e-02)=
DF14	10,10	5.94e-01(2.68e-03)	5.53e-01(2.81e-03)=	9.90e-01(8.01e-02)+	6.94e-01(1.79e-02)=
	5,10	5.96e-01(5.65e-03)	5.84e-01(2.87e-03)=	9.56e-01(8.40e-02)=	6.73e-01(8.21e-03)+
	10,5	5.81e-01(5.09e-03)	5.78e-01(9.36e-04)=	9.71e-01(1.05e-01)=	7.53e-01(3.89e-02)+
+/-/-		23/12/7		12/29/1	22/14/6

Table 4.4: Mean and variance of MHV values obtained by different ablation algorithms over DF test functions.

Problem	n_t, τ_t	MCTS-DMOE	MCTS-DMOE-SBX	MCTS-DMOE-DE	MCTS-DMOE-Rand
DF1	10,10	1.34e-01(8.77e-05)	1.66e-01(5.18e-04)-	1.24e-01(1.53e-04)+	1.46e-01(7.40e-04)=
	5,10	1.22e-01(9.85e-04)	1.54e-01(4.22e-04)-	1.26e-01(2.04e-05)=	1.48e-01(6.39e-04)-
	10,5	1.19e-01(2.71e-04)	1.28e-01(7.12e-04)-	1.10e-01(6.34e-04)=	1.28e-01(8.49e-04)-
DF2	10,10	8.16e-01(1.15e-05)	4.51e-01(3.06e-02)+	8.34e-01(3.31e-04)-	6.69e-01(5.57e-02)+
	5,10	8.14e-01(6.94e-05)	4.41e-01(3.30e-02)+	8.34e-01(2.98e-04)-	6.77e-01(4.85e-02)+
	10,5	8.08e-01(7.73e-06)	4.53e-01(3.18e-02)+	8.31e-01(2.73e-04)-	6.68e-01(5.48e-02)+
DF3	10,10	5.35e-01(1.03e-04)	1.57e-01(3.26e-02)+	5.66e-01(6.88e-04)-	3.92e-01(5.75e-02)+
	5,10	5.34e-01(1.33e-04)	1.64e-01(3.08e-02)+	5.63e-01(5.78e-04)-	3.94e-01(5.32e-02)+
	10,5	5.35e-01(1.26e-04)	1.49e-01(3.42e-02)+	5.64e-01(5.29e-04)-	3.96e-01(5.52e-02)+
DF4	10,10	5.58e+00(4.56e-02)	1.49e+00(3.62e+00)+	5.80e+00(5.21e-02)-	3.96e+00(6.92e+00)+
	5,10	5.69e+00(2.77e-02)	1.38e+00(4.16e+00)+	5.92e+00(4.06e-02)-	4.03e+00(7.05e+00)+
	10,5	5.56e+00(2.37e-02)	1.41e+00(4.09e+00)+	5.80e+00(3.03e-02)-	4.09e+00(6.37e+00)+
DF5	10,10	6.36e-01(5.40e-05)	2.17e-01(4.02e-02)+	6.61e-01(6.05e-04)-	4.80e-01(6.73e-02)+
	5,10	6.35e-01(6.41e-05)	2.16e-01(4.11e-02)+	6.64e-01(5.22e-04)-	4.82e-01(6.65e-02)+
	10,5	6.35e-01(2.16e-05)	2.11e-01(4.07e-02)+	6.60e-01(5.89e-04)-	4.76e-01(6.71e-02)+
DF6	10,10	9.13e-01(6.77e-06)	1.53e-01(1.40e-01)+	9.21e-01(3.38e-05)-	6.12e-01(2.25e-01)+
	5,10	8.91e-01(4.32e-06)	1.49e-01(1.32e-01)+	8.98e-01(2.49e-05)-	5.95e-01(2.13e-01)+
	10,5	9.11e-01(2.44e-06)	1.52e-01(1.39e-01)+	9.19e-01(3.41e-05)-	6.11e-01(2.24e-01)+
DF7	10,10	2.90e+00(3.58e+00)	8.36e-01(4.19e+00)+	2.17e+00(1.55e+00)+	3.31e+00(6.63e+00)=
	5,10	5.98e+00(3.25e+01)	2.19e+00(2.16e+01)+	3.06e+00(1.96e+01)+	8.33e+00(4.19e+01)-
	10,5	2.90e+00(3.21e+00)	8.77e-01(4.53e+00)+	2.15e+00(1.67e+00)+	3.37e+00(6.84e+00)=
DF8	10,10	4.33e+01(3.23e+00)	4.57e+01(7.75e-02)-	4.31e+01(1.03e+00)=	4.52e+01(3.23e-01)=
	5,10	4.04e+01(2.25e+00)	4.26e+01(9.56e-02)-	4.02e+01(9.04e-01)=	4.21e+01(2.44e-01)=
	10,5	4.30e+01(3.39e+00)	4.56e+01(1.67e-01)-	4.28e+01(1.04e+00)=	4.49e+01(2.56e-01)=
DF9	10,10	2.58e+01(1.00e-02)	1.85e+01(1.36e+00)+	2.10e+01(2.12e-02)+	2.00e+01(2.28e+00)=
	5,10	2.45e+01(1.90e-02)	1.72e+01(1.64e+00)+	1.97e+01(1.71e-02)+	1.87e+01(2.43e+00)=
	10,5	2.58e+01(1.63e-02)	1.83e+01(1.64e+00)+	2.10e+01(2.36e-02)+	1.99e+01(2.62e+00)=
DF10	10,10	7.50e-01(3.39e-03)	8.19e-01(3.11e-03)-	3.35e-01(9.30e-02)+	6.83e-01(2.25e-02)=
	5,10	7.83e-01(1.24e-03)	8.71e-01(2.42e-03)-	3.95e-01(8.26e-02)+	7.33e-01(1.46e-02)=
	10,5	7.81e-01(3.42e-03)	8.17e-01(8.37e-04)=	3.61e-01(1.14e-01)+	6.67e-01(2.53e-02)+
DF11	10,10	2.54e-01(1.55e-02)	2.06e-01(1.29e-03)=	1.38e-01(1.94e-02)+	1.78e-01(4.45e-04)=
	5,10	3.25e-01(3.86e-02)	1.75e-01(2.14e-03)=	1.43e-01(3.97e-02)+	1.87e-01(3.09e-03)=
	10,5	2.32e-01(2.82e-02)	1.80e-01(2.33e-03)=	1.15e-01(1.20e-02)+	1.60e-01(1.93e-03)+
DF12	10,10	9.80e+00(3.43e-01)	8.56e+00(4.37e-02)=	7.41e+00(1.03e+00)+	8.09e+00(2.70e-01)+
	5,10	9.27e+00(3.28e-01)	7.81e+00(2.16e-02)=	6.98e+00(1.03e+00)+	7.52e+00(7.66e-02)+
	10,5	9.89e+00(7.81e-02)	8.43e+00(6.03e-02)+	7.15e+00(1.13e+00)+	7.79e+00(4.61e-01)+
DF13	10,10	1.99e-01(1.71e-02)	6.37e-01(3.60e-02)-	1.29e-01(4.36e-03)+	5.39e-01(4.20e-02)-
	5,10	2.92e-01(5.52e-03)	7.30e-01(4.76e-02)-	1.64e-01(1.14e-02)+	5.37e-01(7.31e-02)-
	10,5	2.75e-01(9.30e-03)	6.58e-01(2.01e-02)-	1.71e-01(1.53e-02)+	5.67e-01(4.48e-02)-
DF14	10,10	2.86e-02(1.87e-04)	4.15e-02(2.29e-04)-	7.08e-03(8.56e-05)+	2.50e-02(1.50e-04)=
	5,10	2.89e-02(1.08e-04)	4.10e-02(1.11e-04)-	9.48e-03(1.43e-04)+	2.70e-02(2.40e-04)=
	10,5	3.30e-02(3.57e-04)	3.26e-02(1.08e-04)=	9.67e-03(2.63e-04)+	2.38e-02(3.27e-04)+
+/-/-		22/7/13		22/5/15	21/15/6

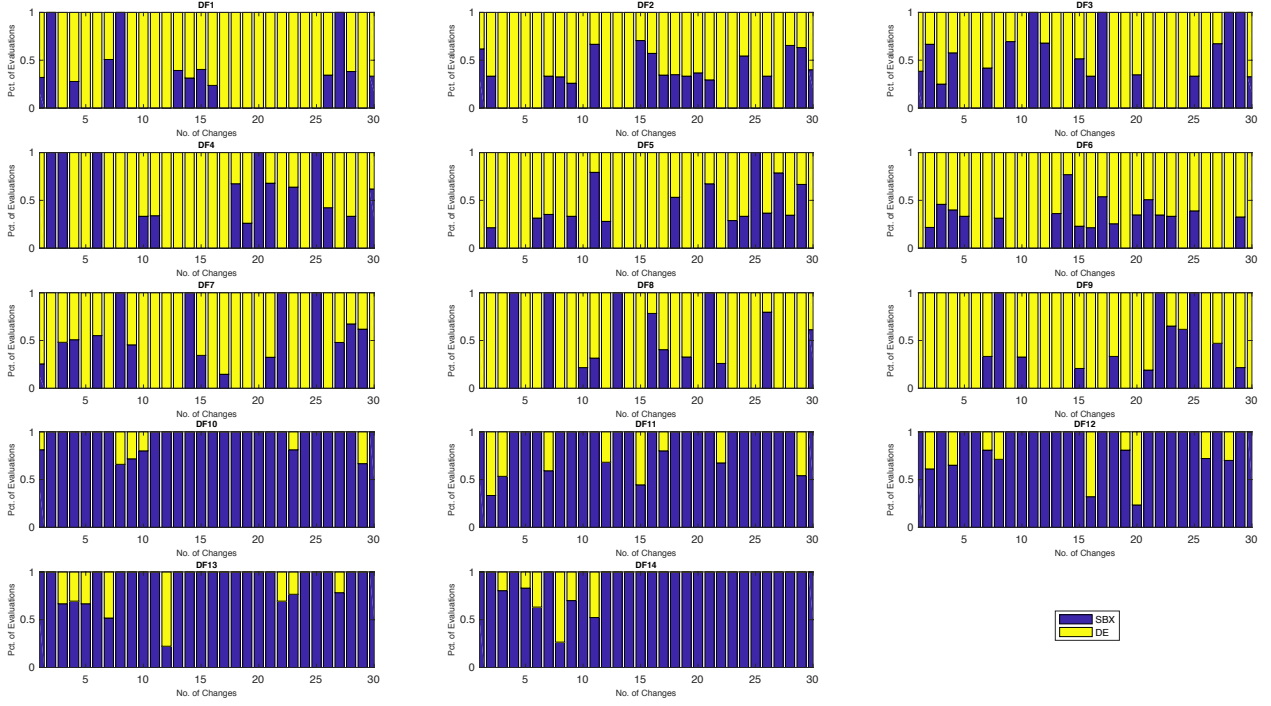


Figure 4.8: The proportion of allocation of computing resources to SBX and DE operators.

bi-objective functions. Conversely, the algorithm prefers to allocate more computing resources to the SBX operator when operating under tri-objective functions.

4.3.4 Runtime Performance

The runtime of a DMOEA is an important performance indicator in evaluating its efficacy. In this section, we compare the average runtime per environment step of various DMOEAs. The runtime results are presented in Table 4.5. The MCTS-DMOEA algorithm exhibits an impressive average runtime per environment step of $5.22e - 01$ seconds. This time performance is only surpassed by that of MDP and SVR, which take $6.10e - 02$ and $5.64e - 02$ seconds, respectively. Notably, compared with those prediction-based DMOEAs, the MCTS-DMOEA algorithm does not require any additional training time for prediction models, making it a highly efficient and time-saving option.

Table 4.5: Mean of runtime obtained by compared algorithms at $\tau_t=10$ and $\tau_t=5$. (in seconds)

Problem	τ_t	MCTS-DMOEA	MDP	MMTL	SVR	IGP	TK-DMOEA	CT-DMOEA
DF1	10	7.33e-01	1.01e-01	2.85e+00	5.78e-02	1.55e+00	1.62e+00	1.97e+01
	5	3.90e-01	4.98e-02	1.58e+00	4.83e-02	7.40e-01	8.77e-01	1.02e+01
DF2	10	6.70e-01	7.91e-02	2.77e+00	5.58e-02	1.56e+00	1.67e+00	1.94e+01
	5	3.54e-01	4.12e-02	1.65e+00	4.85e-02	8.35e-01	9.41e-01	9.40e+00
DF3	10	6.23e-01	1.27e-01	2.52e+00	6.13e-02	1.15e+00	1.33e+00	1.36e+01
	5	3.22e-01	7.03e-02	1.48e+00	5.35e-02	5.49e-01	8.30e-01	6.67e+00
DF4	10	6.14e-01	1.53e-01	2.30e+00	7.13e-02	1.20e+00	1.30e+00	2.01e+01
	5	3.24e-01	7.57e-02	1.36e+00	5.78e-02	6.04e-01	7.22e-01	1.19e+01
DF5	10	5.88e-01	6.61e-02	3.31e+00	5.59e-02	1.40e+00	1.71e+00	2.06e+01
	5	3.33e-01	3.10e-02	1.85e+00	4.96e-02	7.10e-01	9.37e-01	1.05e+01
DF6	10	7.84e-01	6.39e-02	2.52e+00	5.52e-02	1.31e+00	1.44e+00	1.45e+01
	5	4.27e-01	3.25e-02	1.51e+00	4.72e-02	6.13e-01	7.75e-01	8.11e+00
DF7	10	7.00e-01	5.30e-02	2.54e+00	6.08e-02	1.09e+00	1.06e+00	1.82e+01
	5	4.13e-01	3.13e-02	1.76e+00	5.33e-02	5.39e-01	6.17e-01	9.95e+00
DF8	10	7.02e-01	9.62e-02	2.60e+00	5.71e-02	1.49e+00	1.59e+00	1.60e+01
	5	3.74e-01	5.15e-02	1.48e+00	5.38e-02	7.41e-01	8.69e-01	6.92e+00
DF9	10	6.90e-01	8.96e-02	2.66e+00	6.66e-02	1.17e+00	1.55e+00	1.42e+01
	5	3.67e-01	4.43e-02	1.72e+00	5.50e-02	5.79e-01	8.43e-01	6.19e+00
DF10	10	6.43e-01	7.21e-02	3.95e+00	6.06e-02	1.38e+00	2.11e+00	1.91e+01
	5	3.69e-01	4.45e-02	2.28e+00	5.79e-02	6.89e-01	1.27e+00	1.04e+01
DF11	10	7.39e-01	4.06e-02	4.31e+00	5.28e-02	1.53e+00	2.19e+00	2.18e+01
	5	3.94e-01	4.24e-02	2.62e+00	4.72e-02	7.54e-01	1.38e+00	1.23e+01
DF12	10	6.93e-01	9.42e-02	1.57e+00	6.06e-02	7.91e-01	1.25e+00	2.27e+01
	5	3.77e-01	4.67e-02	9.23e-01	5.35e-02	4.06e-01	8.61e-01	1.26e+01
DF13	10	6.33e-01	3.47e-02	5.20e+00	6.63e-02	1.48e+00	2.09e+00	2.17e+01
	5	3.37e-01	1.91e-02	3.11e+00	5.42e-02	7.34e-01	1.39e+00	1.23e+01
DF14	10	6.76e-01	3.45e-02	4.98e+00	6.27e-02	1.49e+00	2.14e+00	1.95e+01
	5	3.60e-01	2.35e-02	2.83e+00	5.57e-02	7.22e-01	1.33e+00	9.74e+00
Rank		3	2	6	1	4	5	7

Overall, the proposed MCTS-DMOEA method demonstrates superior performance in terms of runtime, MIGD, and MHV results, indicating that it can achieve better convergence and diversity while maintaining a fast running speed.

4.4 Conclusion

This work designed a Monte Carlo tree-assisted computational resource allocation algorithm for solving DMOPs, called MCTS-DMOEA. Specifically, MCTS-DMOEA allocates computational resources with a higher priority to promising subpopulations and evolutionary operators. Firstly, the proposed MCTS-DMOEA uses a hierarchical clustering method to adaptively partition populations and create a number of subpopulations. Then, the Monte Carlo tree-assisted computational resource allocation strategy allocates computing resources to more promising subpopulations according to UCB values and automatically adopts specific evolutionary operators. To conduct a fairer experiment, this work extends the concept of the existing dynamic setting of the DMOPs to fit the scenario with limited computing resources. The experimental results have demonstrated that the proposed MCTS-DMOEA is more competitive than several state-of-the-art DMOEAs in terms of exploiting computational resources, population convergence, and population diversity.

There are several possible directions for future work. For example, the proposed MCTS-DMOEA could be extended for large-scale optimization, constrained optimization, etc. Further, transfer learning techniques such as evolutionary transfer optimization could inspire further innovations in co-evolution among subpopulations. These avenues offer promising opportunities for future research to enhance the efficiency and efficacy of EAs for real-world complex optimization problems.

Chapter 5

Knowledge-Driven Spatial-Temporal Knowledge Transfer for Dynamic Constrained Multiobjective Optimization

5.1 Motivation

DCMOPs, which are characterized as conflicting optimization objectives and constraints that change throughout the optimization, widely exist in real-world applications [156, 157, 5]. For instance, in the dynamic electric vehicle routing problem [5], the goal is to minimize total energy consumption and total travel time while considering time-varying environmental factors such as the number of vehicles and route topology. Moreover, solutions must adhere to constraints related to battery capacity and battery level to guarantee their feasibility. Due to the simultaneous presence of constraints and dynamism, DCMOPs are more challenging than DMOPs

and CMOPs since Pareto optimal solutions should be rapidly tracked, and the solutions found must be feasible.

Given that DCMOPs exhibit characteristics of both CMOPs and DMOPs, maintaining diversity to solve DCMOPs kills two birds with one stone. Firstly, diversity enables the population to navigate away from local optima after environmental changes. Secondly, preserving diverse solutions in infeasible regions is beneficial for the population to climb infeasible barriers to approach feasible regions. To this end, existing dynamic constrained multiobjective optimization algorithms (DCMOAs) have designed adaptive constraint-handling strategies to enhance feasibility while maintaining diversity [64, 66, 158, 65]. For example, Azzouz *et al.* [65] and Chen *et al.* [66] have devised adaptive penalty functions that utilize the feasibility ratio to strike a balance between exploring feasible and infeasible regions. These approaches allow evolutionary solvers to retain feasible solutions to ensure feasibility while preserving infeasible solutions to maintain diversity. Although the adaptive penalty function drives the solutions to the feasible regions, these adaptive penalty functions still suffer from the loss of diversity. As shown in Fig. 5.1 (a), adaptive penalty functions could force the population toward the nearly feasible region, but the population could not cover all feasible areas, particularly in the case of DCMOPs with narrow or disconnected feasible regions.¹ That is, the population diversity is damaged. Subsequently, when the environment changes from $t = 1$ to $t = 2$, the population easily becomes trapped in local optima due to the lack of diversity.

In addition to adaptive penalty functions, other constraint handling techniques such as constraint transformation methods [60, 61, 62] and hybrid techniques [76, 77, 78] have been proposed to enhance the feasibility. However, these methods exhibit selection bias by assigning higher priority to either objectives or constraints during solution selection. When constraints are prioritized over objectives, population diversity can rapidly decrease [66, 81]. Consequently, in the event of environmental changes, the population may become trapped in local optima.

¹The drawbacks of adaptive penalty functions in maintaining diversity are detailed in Chapter 2.

Conversely, if a higher weight is given to objectives, an excess number of infeasible solutions may slow the convergence speed [66, 81]. This contradicts the objective of rapidly tracking Pareto optimal solutions in dynamic environments. Overall, existing constraint-handling methods are not suitable to extend for DCMOPs, as they could lead to being trapped at local optima or slow in convergence.

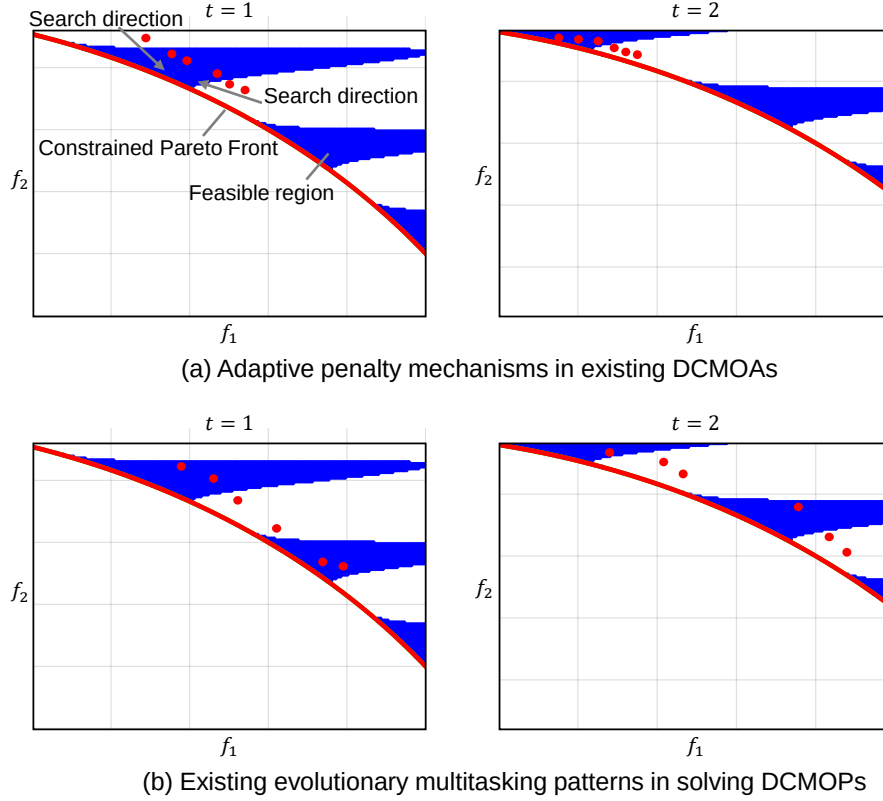


Figure 5.1: Limitations of existing methods in solving DCMOPs. (a) Adaptive penalty mechanisms in existing DCMOAs could lead to the loss of diversity when dealing with DCMOPs with narrow or disconnected feasible areas. (b) An evolutionary multitasking-based static constrained multiobjective optimization algorithm (MTCMO) [2] that is equipped with the dynamic response strategies in DNSGA2 [3] suffers from slow convergence when dealing with DCMOPs.

Recently, several evolutionary multitasking-based constrained multiobjective optimization algo-

rithms have emerged, aiming to leverage the potential synergies between relaxed-constrained or unconstrained search space and constrained search space [95, 2]. They constructed an auxiliary task with simplified constraints to search the relaxed-constrained/unconstrained search space, serving to enhance diversity to the main task, i.e., the original CMOP. Multitasking-based constrained multiobjective optimization has established a new paradigm for solving CMOPs. However, these approaches often exhibit slow convergence when applied to DCMOPs, as illustrated in Fig. 5.1 (b). The reason for this observation is the absence of sophisticated dynamic response strategies that can effectively utilize the correlation or similarity between the relaxed-constrained search space and constrained search space across consecutive environments. As a result, low-quality solutions could be introduced or retained, thereby impeding the convergence in newly generated environments.

It is worth noting that the unconstrained search space and constrained search space exhibit similarities from both temporal and spatial perspectives. Temporal similarity arises from the conversion between the unconstrained search space and the constrained search space in the dynamic environment. Spatial similarity stems from the fact that the constrained search space is a part of the unconstrained search space. This is because the unconstrained search space contains the set of all possible solutions without any constraints, while the constrained search space applies certain constraints, reducing the set of possible solutions to a subset of solutions. However, as mentioned above, existing DCMOAs, constraint handling techniques, and evolutionary multitasking-based methods suffer from poor diversity, local optima, or slow convergence when it comes to solving DCMOPs. They either overlook the utilization of spatial similarity to synergize the search for infeasible and feasible solutions or overlook the use of temporal similarity to accelerate the search in new environments. To bridge this research gap, we aim to leverage both temporal similarity and spatial similarity to effectively address DCMOPs. Temporal similarity allows us to leverage the search experience from both the unconstrained search space and the constrained search space in previous environments, enabling us to accelerate the search process in newly generated environments. At the same time, spatial similarity

allows for synergistic exploration between the constrained search space and the unconstrained search space, supplementing the population diversity.

This work proposes an evolutionary multitasking-based approach with a spatial-temporal knowledge transfer strategy, called the dynamic constrained multifactorial evolutionary algorithm (DC-MFEA), to solve DCMOPs. In view of the superiority of evolutionary multitasking, two synergistic tasks are established and optimized simultaneously. One is to search the constrained search space to drive the population to feasible regions, and the other is to search the unconstrained search space to maintain diversity. Within the multitasking framework, the spatial knowledge transfer is achieved by the MFEA algorithm tailored for handling constraints, and the temporal knowledge transfer is accomplished through an environmental similarity-based dynamic response mechanism to track the changing Pareto optimal front by reusing historical solutions. The main contributions of this work are summarized as follows,

1. To enhance feasibility while maintaining diversity under dynamic environments, DC-MFEA optimizes two synergetic tasks to promote the search of both unconstrained search space and constrained search space. This approach compensates for the diversity loss typically brought by constraint-handling methods in existing DCMOAs.
2. To accelerate the evolutionary process, DC-MFEA conducts knowledge transfer across the constrained and unconstrained spaces from both a temporal and spatial perspective. By transferring both temporal knowledge and spatial knowledge, the synergy between two tasks is promoted at large, boosting the search efficiency in dynamic environments.
3. To advance DCMOPs toward real-world cases, a test suite including 14 test problems is designed with various characteristics, including time-varying feasible and infeasible regions, time-varying geometry of the Pareto front, and time-varying connectivity of feasible areas, etc.

5.2 Proposed Framework

In this section, the details of the DC-MFEA are borrowed in a top-down style. The overview of DC-MFEA is first presented. Afterward, the spatial knowledge transfer process and temporal knowledge transfer process of DC-MFEA are given.

5.2.1 Overview

Briefly, DC-MFEA follows the paradigm of multifactorial evolutionary algorithms [159], where two tasks are encoded in the unified search space and optimized simultaneously: The main task is to search within the constrained search space, i.e., the original DCMOP, while the auxiliary task removes constraints from the DCMOP to conduct the search in the unconstrained search space. Within this framework, genetic operators implicitly transfer spatial knowledge. When the environment changes, solutions from the constrained search space and solutions from the unconstrained search space are archived in archives A^C and A^U , respectively. According to the estimated similarities between two consecutive environments, temporal knowledge transfer reuses the historical solutions from the two archives for the new environment. The two processes are organically incorporated to jointly improve the performance of solving DCMOPs. In Fig. 5.2, we give a detailed illustration of the DC-MFEA.

The main scheme of the proposed DC-MFEA is presented in Algorithm 8. At the beginning, the population P_0 for the main task and the population Q_0 for the auxiliary task are randomly initialized. If the environment remains unchanged, Algorithm 9 will optimize the two populations (Line 9), and the outputted population will be saved into the corresponding archives A^C and A^U (Line 10). If the environment changes, in Algorithm 10, solutions P_t and Q_t are retrieved from A^C and A^U to guide the search according to the environmental similarities (Line 7). After that, P_t and Q_t are iteratively optimized by Algorithm 9. For each environment, $DCPOS_t$ is identified from P_t as the outputted solutions.

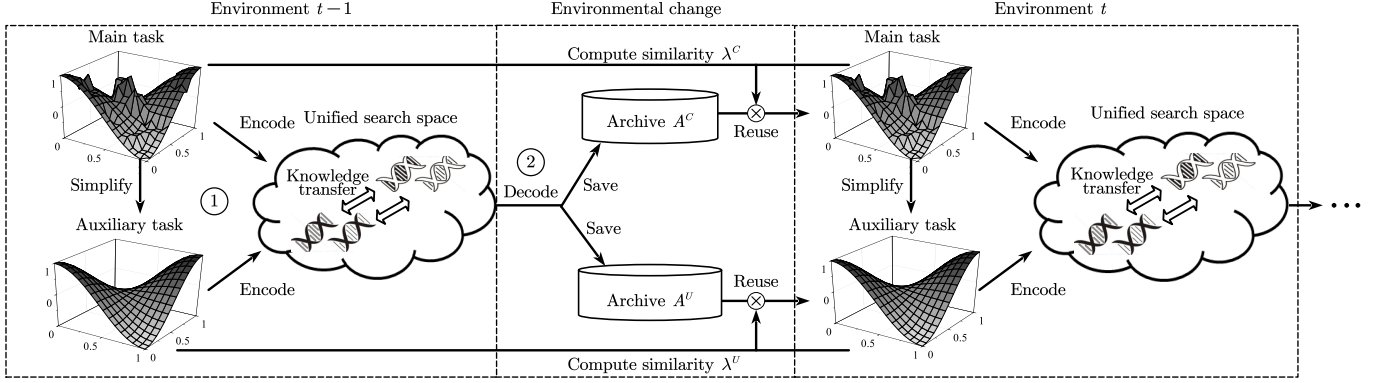


Figure 5.2: The framework of the proposed DC-MFEA. (1) Spatial knowledge transfer: An auxiliary task is derived from the main task (the original DCMOP), and then the two tasks are encoded and optimized in the unified search space, where the spatial knowledge is implicitly transferred by genetic operators. (2) Temporal knowledge transfer: When the environment changes, solutions from the constrained search space and solutions from the unconstrained search space are archived in archives A^C and A^U , respectively. According to the estimated similarities between two consecutive environments, solutions in the two archives are reused for the new environment.

In the following subsections, we will introduce the two main components: Algorithm 9 and Algorithm 10.

5.2.2 Spatial Knowledge Transfer

Due to the spatial similarity, the search knowledge or search experience gained from two tasks can be leveraged for efficient problem-solving. In our work, we transfer spatial knowledge in the form of solutions of the two tasks to synergize the search of the two tasks. Particularly, MFEA [159] is employed as the implicit knowledge transfer method for spatial knowledge transfer, where the knowledge transfer is achieved by the crossover operator.

Specifically, the main task focuses on searching the constrained space to drive the population

Algorithm 8: DC-MFEA

Input: the DCMOP $F_t(\cdot)$, the number of individuals for main task N_{T1} , the number of individuals for auxiliary task N_{T2}

Output: the $DCPOS_t$ of the DCMOP at the environment step t

```

1 Set environment step  $t = 0$ ;
2 Initialize a population  $P_0$  for the main task,  $|P_0| = N_{T1}$ ;
3 Initialize a population  $Q_0$  for the auxiliary task,  $|Q_0| = N_{T2}$ ;
4 while stopping criterion is not met do
5     if a change is detected then
6          $t = t + 1$ ;
7          $P_t, Q_t = \mathbf{TKT}(A^C, A^U)$ ;
8     end
9      $P_t, Q_t = \mathbf{SKT}(F_t(\cdot), P_t, Q_t)$ ;
10     $A^C = P_t, A^U = Q_t$ ;
11    Identify the  $DCPOS_t$  in  $P_t$ ;
12    return  $DCPOS_t$ ;
13 end

```

to feasible regions, while the auxiliary task is created to search the space without constraints to maintain diversity. Particularly, two environmental selection strategies are adopted to select solutions from the constrained space and unconstrained space. First, an auxiliary task $F'_t(\cdot)$ is created by ignoring the constraints of $F_t(\cdot)$. Then, SKT optimizes these two optimization problems $F_t(\cdot)$ and $F'_t(\cdot)$ simultaneously with the population P_t of the main task and the population Q_t of the auxiliary task. The two populations are encoded in a unified search space, and the encoded decision variable $x' \in R^D$ is calculated as,

$$x'_d = \frac{x_d - x_d^{lb}}{x_d^{ub} - x_d^{lb}}, \quad (5.1)$$

where x_d is the d -th dimension of an individual x in the original space, and x_d^{ub} and x_d^{lb} represent the upper bound and lower bound of the d -th dimension of x , respectively. After encoding, we then calculate the factorial properties of solutions, i.e., factorial rank, scalar fitness, and skill factor.

1) Factorial Rank: the factorial rank r_x^k of individual x for the k -th task denotes the index of x in the list of individuals sorted in ascending order w.r.t their nondominated fronts on the k -th

task. ($k = 1$ and $k = 2$ mean the main task and the auxiliary task, respectively.)

2) Scalar Fitness: The scalar fitness ϕ_x of an individual x is based on its best rank over all tasks and defined as the reciprocal of the minimum factorial rank of x , i.e., $\phi_x = 1/\min\{r_x^1, r_x^2\}$.

3) Skill Factor: The skill factor ℓ_x of an individual x is defined as $\ell_x = \arg \min_{j \in \{1,2\}} \{r_x^j\}$, that is, among all tasks, x performs best in task ℓ_x .

Then, assortative mating and vertical cultural transmission are sequentially carried out to reproduce the offspring, and the knowledge of the two tasks is transferred by the crossover operator in assortative mating. For the environmental selection, because the main task focuses on feasibility while the auxiliary task pays more attention to diversity, SKT adopts two environmental selection strategies for the two tasks. Specifically, the environmental selection of the main task employs a feasibility-driven method, nondominated sorting with CDP to select solutions, while the environmental selection of the auxiliary task is achieved by the classic nondominated sorting method. For nondominated sorting with CDP, feasible solutions dominate infeasible solutions all the time. The dominance between feasible solutions depends on the Pareto dominance between objective values. For two infeasible solutions, the dominance depends on the constraint violation (CV) value,

$$CV = (CV_1, \dots, CV_{N_H}, \dots, CV_{N_H+N_G}), \quad (5.2)$$

where

$$CV_k = \begin{cases} \max(0, h_k(x, t)), & k = 1, \dots, N_H, \\ \max(0, g_k(x, t)), & k = 1 + N_H, \dots, N_H + N_G, \end{cases} \quad (5.3)$$

where x comes from the main task used for searching towards feasibility, i.e., x has the skill factor $\ell_x = 1$. For two infeasible solutions x_1 and x_2 , x_2 dominates x_1 ($x_2 \succ x_1$), if and only if $\forall k = 1, \dots, N_H + N_G, CV_k(x_2) \leq CV_k(x_1)$ and $\exists k = 1, \dots, N_H + N_G, CV_k(x_2) < CV_k(x_1)$.

In SKT, the main task focuses on feasibility, and the auxiliary task aims to maintain diversity. Due to the spatial similarity between the two tasks and the synergy of infeasible and feasible

Algorithm 9: Spatial Knowledge Transfer (SKT)

Input: the DCMOP $F_t(\cdot)$, the population of the main task P_t , the population of the auxiliary task Q_t

Output: P_t, Q_t

- 1 Remove constraints from the $F_t(\cdot)$ to create an auxiliary task $F'_t(\cdot)$;
 - 2 Encode $P_t \cup Q_t$ in the unified space;
 - 3 Calculate the skill factor, factorial rank, and scalar fitness of each solution in $P_t \cup Q_t$;
 - 4 Perform assortative mating on $P_t \cup Q_t$ to obtain the offspring O_t ;
 - 5 Determine the task for each solution in O_t by vertical cultural transmission;
 - 6 Calculate $CV(x)$, where $\{x|x \in P_t \cup Q_t \cup O_t, \ell_x := 1\}$;
 - 7 Perform environmental selection for $F_t(\cdot)$ by using CDP to obtain P_t ;
 - 8 Perform environmental selection for $F'_t(\cdot)$ by using nondominated sorting to obtain Q_t ;
 - 9 **return** P_t, Q_t ;
-

solutions, the DC-MFEA enables useful spatial knowledge to be transferred among the two tasks via the crossover operators. Once the diversity is improved by the spatial knowledge transfer, solutions could climb infeasible barriers to approach feasible regions, thus improving the feasibility of the main task. On the other hand, enhancing diversity enables the population to escape from local optima and improve the convergence quality. The procedure of SKT is outlined in Algorithm 9.

5.2.3 Temporal Knowledge Transfer

Temporal knowledge refers to the historical solutions obtained from both tasks. The TKT module leverages the similarity between consecutive environments to reuse the stored historical solutions from the main task's archive A^C and the auxiliary task's archive A^U . The higher the temporal similarity between environments, the more solutions from the two archives can be reused to guide the search for the newly generated environment.

Specifically, the main task's similarity λ^C and the auxiliary task's similarity λ^U between the new environment and the past environment are estimated according to the change of rankings between the new environment and the past environment. For each solution x in A^C or A^U , the ranking of it in the t -th environment is denoted as $\rho_{x,t}^{(\cdot)}$. Here, $(\cdot)$ means the solution x is from

the main task C or auxiliary task U for simplicity. The rankings $\rho_{x,t}^U$ and $\rho_{x,t}^C$ are obtained by nondominated sorting and nondominated sorting with CDP, respectively.

Then, the rankings $\rho_{x,t}^U$ and $\rho_{x,t}^C$ are modified by crowding distance $CD(x)$. The larger the $CD(x)$ of a solution x , the better diversity can be maintained when keeping the solution x in the population, and the better the ranking $\rho_{x,t}^{(\cdot)}$ of the solution x . In this way, we expect solutions with higher diversity can be kept to help the population climb the infeasible barriers. Given a solution $x \in A^{(\cdot)}$ and its ranking $\rho_{x,t}^{(\cdot)}$ in the t -th environment, the crowding distance $CD(x)$ can be calculated as,

$$CD(x) = \sum_{x^* \in R_{x,t}} \sum_{m=1}^M |f_m(x) - f_m(x^*)|, \quad (5.4)$$

where $R_{x,t}$ is the set of solutions with the same rankings to x .

$$R_{x,t} = \{x^* | x^* \in A^{(\cdot)} / x, \rho_{x^*,t}^{(\cdot)} := \rho_{x,t}^{(\cdot)}\}, \quad (5.5)$$

where $A^{(\cdot)} / x$ denotes the set of solutions excluding x from $A^{(\cdot)}$. Then, $CD(x)$ is normalized to $[0, 0.5]$,

$$\overline{CD}(x) = \frac{CD(x) - CD_{min}}{2(CD_{max} - CD_{min})}, \quad (5.6)$$

where CD_{max} and CD_{min} are the maximum and minimum values of CD . Afterwards, the modified ranking is given by,

$$\rho_{x,t}^{(\cdot)} = \rho_{x,t}^{(\cdot)} - \overline{CD}(x). \quad (5.7)$$

It should be noted that by utilizing the normalized range of $[0, 0.5]$, the modified ranking of x in the current nondominated front will not be equal to the ranking of the previous nondominated front.

When the change is detected, all solutions in the A^C and A^U are immediately re-evaluated, and the newly generated rankings are also modified by the crowding distance. Given two consecutive environments $t-1$ and t , the ranking sets $\{\rho_{x,t-1}^{(\cdot)}\}$ and $\{\rho_{x,t}^{(\cdot)}\}$ are different due

Algorithm 10: Temporal Knowledge Transfer (TKT)

Input: the archive for the main task A^C , the archive for the auxiliary task A^U

Output: the population P_t of the main task, the population Q_t of the auxiliary task

- 1 Get the historical ranking $\rho_{x,t-1}^{(\cdot)}$ for each solution from $A^{(\cdot)}$;
 - 2 Re-evaluate solutions from $A^{(\cdot)}$ to get the current ranking $\rho_{x,t}^{(\cdot)}$;
 - 3 Compute the crowding distance CD ;
 - 4 Modify the rankings according to Eq. (5.7);
 - 5 Compute Spearman's rank correlation coefficient according to Eq. (5.8);
 - 6 $\lambda^C = \max\{\lambda^C, 0\}$, $\lambda^U = \max\{\lambda^U, 0\}$;
 - 7 Select the top $\lambda^C N_{T1}$ ranking solutions from A^C and denote as P_t ;
 - 8 Select the top $\lambda^U N_{T2}$ ranking solutions from A^U and denote as Q_t ;
 - 9 Randomly generate $N_{T1} - \lambda^C N_{T1}$ solutions and add them to P_t ;
 - 10 Randomly generate $N_{T2} - \lambda^U N_{T2}$ solutions and add them to Q_t ;
 - 11 **return** P_t, Q_t ;
-

to the environmental change. Consequently, Spearman's rank correlation coefficient is used to compute the change in rankings to obtain the two similarities $\lambda^{(\cdot)}$,

$$\lambda^{(\cdot)} = \frac{cov(\{\rho_{x,t}^{(\cdot)}\}, \{\rho_{x,t-1}^{(\cdot)}\})}{std(\{\rho_{x,t}^{(\cdot)}\})std(\{\rho_{x,t-1}^{(\cdot)}\})}, \quad (5.8)$$

where $cov(\cdot, \cdot)$ is the covariance of a pair of sets and $std(\cdot)$ is the standard deviation of the input set. When $|\lambda^{(\cdot)}| \rightarrow 1$, it means the two consecutive search spaces are more correlated. Considering the negative correlation ($\lambda^{(\cdot)} < 0$) should be eliminated, we rewrite the similarities as,

$$\lambda^{(\cdot)} = \max\{\lambda^{(\cdot)}, 0\}. \quad (5.9)$$

According to the two similarities, the top $\lambda^C N_{T1}$ ranking solutions in A^C and the top $\lambda^U N_{T2}$ ranking solutions in A^U are directly reused in the new environment, where N_{T1} and N_{T2} are the numbers of solutions for the main and auxiliary tasks, respectively. The remaining $N_{T1} - \lambda^C N_{T1}$ solutions for the main task and $N_{T2} - \lambda^U N_{T2}$ solutions for the auxiliary task are randomly reinitialized to supplement the populations P_t and Q_t , respectively. The procedure of TKT is outlined in Algorithm 10.

5.2.4 Complexity Analysis

In SKT, the time complexity for knowledge transfer and generating offspring is $O(DN)$, where $N = N_{T1} + N_{T2}$ indicates the population size. In environmental selection, the nondominated sorting and crowding distance account for the computational complexities of $O(MN^2)$ and $O(\max(M + N_H + N_G)N \log N)$. For TKT, the computational complexity is $O(MN^2)$ due to the nondominated sorting. Overall, the computational complexity of DC-MFEA is $O(MN^2)$.

5.3 Proposed Benchmark

The benchmark problem is the basis of evaluating the effectiveness of an algorithm and can guide algorithmic design. Although many test suites for DMOPs [160, 161, 154] and CMOPs [162, 163, 164] have been elaborately devised, the design of DCMOPs is in its infancy. Azzouz *et al.* [65] designed a set of test instances for DCMOPs where the true DCPOS in the designed test problems show only small variations between t and $t + 1$. Chen *et al.* [66] introduced two important characteristics: the change in connectivity of the DCPOF and the change in the size of the feasible regions. Chen *et al.* [165] designed a set of test instances for DCMOPs that incorporate various characteristics such as time-varying size, location, and shape of feasible regions, controllable change severity, as well as small feasible regions. However, these DCMOP test suites are still far from reality and do not involve several complicated characteristics. In real-world applications of DCMOPs, complicated dynamic characteristics possess not only the above two characteristics but also the change in the connectivity of the feasible region and varying geometry of the DCPOF and DUPOF [162]. Specifically, the dynamism in more realistic test problems can change the boundary, connectivity, and feasibility ratio of the feasible regions; and can also affect the geometry of the DCPOF and DUPOF. In our design, we aim to design a DCMOP test suite with such complicated dynamic characteristics.

According to the change in DCPOF and DUPOF, we classify DCMOPs into the following four

types, and then based on the four types, we design our DCMOP test suite containing the above complicated dynamic characteristics. It should be noted that in our classification, all DCPOSs change over time.

1. Type I: The DCPOF and DUPOF remain the same in the dynamic environment, while the DCPOS is changed.
2. Type II: DUPOF remains the same in the dynamic environment, while DCPOF and DCPOS are changed over time.
3. Type III: DCPOF remains the same, while DUPOF and DCPOS are changed.
4. Type IV: DCPOF, DUPOF, and DCPOS are all changed over time.

Following the principle of designing constraints [162] and the principle of dynamics [154], we design a set of 14 test problems (called DCF1~ DCF14) in this work. These test problems cover all the above four types according to our taxonomy. More specifically, the developed test suite involves the following characteristics: the change of connectivity of DCPOF, the change of the connectivity and the size of the feasible regions, the varying geometry of DCPOF and DUPOF, nonlinear constraints, varying feasibility ratio, and other important characteristics.

Regarding the dynamics, the proposed DCF test suite follows the commonly used discrete environment step [154], where the environment step t in DCFs is governed by,

$$t = \frac{2}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor, \quad (5.10)$$

where τ , n_t , and τ_t refer to the maximum number of evolutionary generations, the severity of the environmental change, and the frequency of the environmental change, i.e., the maximum number of generations for each environment step, respectively. A higher τ_t value indicates a slower changing environment, while a lower n_t value denotes a more dramatic changing environment.

Table 5.1: Characteristics of the proposed DCF test functions.

Problem	Type	Feasible region		DCPOF	DUPOF	Constraints		Feasibility Ratio
		Size	Connectivity	Geometry	Geometry	No. of	Nonlinear	
DCF1	I	Changed	Connected	Connected, Concave	Connected, Concave	1	Yes	4.18%~4.36%
DCF2	IV	Changed	Connected	Connected, Linear	Connected, Linear	1	Yes	8.45%~15.28%
DCF3	I	Changed	Disconnected	Disconnected, Concave	Connected, Concave	2	Yes	5.69%~20.44%
DCF4	II	Changed	Disconnected	Disconnected, Concave	Connected, Concave	2	Yes	3.51%~17.86%
DCF5	IV	Changed	Disconnected	Disconnected, Concave	Connected, Concave	2	Yes	10.92%~20.34%
DCF6	III	Changed	Connected	Disconnected, Mixed	Disconnected, Mixed	2	Yes	1.75%~12.43%
DCF7	I	Unchanged	Connected	Disconnected, Mixed	Connected, Concave	2	Yes	3.39%
DCF8	II	Changed	Changed	Changed connectivity, Mixed	Connected, Linear	2	Yes	1.73%~15.83%
DCF9	IV	Changed	Disconnected	Disconnected, Mixed	Connected, Concave	2	Yes	7.26%~9.61%
DCF10	III	Unchanged	Disconnected	Disconnected, Linear	Connected, Linear	4	Yes	3.97%
DCF11	I	Unchanged	Disconnected	Connected, Mixed	Connected, Mixed	2	Yes	12.44%
DCF12	II	Changed	Changed	Connected, Mixed	Connected, Mixed	2	Yes	7.99%~12.44%
DCF13	IV	Changed	Disconnected	Connected, Mixed	Connected, Mixed	2	Yes	4.59%~14.20%
DCF14	III	Unchanged	Disconnected	Connected, Mixed	Disconnected, Changed	2	Yes	12.44%

Table 5.1 summarizes the main characteristics of the proposed DCFs. The mathematical forms, geometries, and detailed characteristics of the 14 test problems are provided as follows. In Table 5.1, the feasibility ratio is the ratio of feasible search space to the total search space, and here we calculate the area of the search space using a grid.

DCF1:

$$\begin{aligned}
\min & \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{1 - f_1^2} \end{cases} \\
s.t. & c(x) = l_1 + 0.5 \sin(t + l_2)^{10} \leq 0 \\
& G(t) = 0.5|\sin(0.5\pi t)| \\
& g = 1 + 0.1\left(\left|\sum_{i=2}^D x_i - G\right|\right)^{0.5} \\
& l_1 = 1 - f_1^2 - f_2^2 \\
& l_2 = 5\sqrt{2}\pi f_2 - \sqrt{2}f_1 - 1
\end{aligned} \tag{5.11}$$

DCF1 has a changing feasible region. The DCPOF and DUPOF are totally overlapped, and the feasible region is connected by DCPOF. Although the connectivity of the feasible region

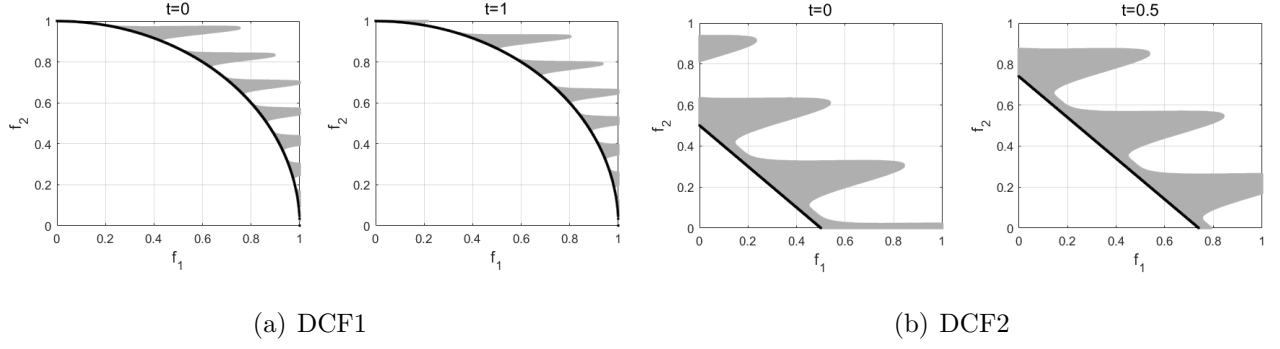


Figure 5.3: The objective space of (a) DCF1 and (b) DCF2.

is connected, narrow connectivity challenges the algorithm. DCF1 is plotted in Fig. 5.3 (a), where the overlapped DCPOF and DUPOF are shown in black. The gray area is the feasible region. The search space is $[0, 1]^D$.

DCF2:

$$\begin{aligned}
 \min \quad & \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(0.5 + G - f_1) \end{cases} \\
 \text{s.t.} \quad & c(x) = l_1 + 0.6 \sin(2t + l_2)^4 \leq 0 \\
 & G(t) = 0.5|\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & l_1 = 0.55 + G - f_1 - f_2 \\
 & l_2 = 2\sqrt{2}\pi f_2 - \sqrt{2}f_1 - 1
 \end{aligned} \tag{5.12}$$

DCF2 also has a changing feasible region, and the changing DCPOF and DUPOF of DCF2 are overlapped. In addition, DCF2 has a changing search space $[0, 0.5 + G]^D$. DCF2 is plotted in Fig. 5.3 (b).

DCF3:

$$\begin{aligned}
& \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{1-f_1} \end{cases} \\
& \text{s.t.} \begin{cases} c_1(x) = 1.1 + C - f_1^2 - f_2^2 \leq 0 \\ c_2(x) = (1 + 0.5 \sin(6l))^2 - f_1^2 - f_2^2 \leq 0 \end{cases} \\
& G(t) = |\sin(t)| \\
& g = 1 + 0.1 \left(\left| \sum_{i=2}^D x_i - G \right| \right)^{0.5} \\
& C(t) = |\sin(t)| \\
& l = 0.5\pi - 2 \left| \arctan \frac{f_2}{f_1} - 0.25\pi \right|
\end{aligned} \tag{5.13}$$

The DCPOF of DCF3 is a part of the DUPOF. DCF3 has multiple constraints, and the constraints make the changing feasible region disconnected. The search space is $[0, 1]^D$. DCF3 is plotted in Fig. 5.4 (a), where the DCPOF and DUPOF are shown in black and green, respectively.

DCF4:

$$\begin{aligned}
& \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{1-f_1} \end{cases} \\
& \text{s.t.} \begin{cases} c_1(x) = 1.1 + C - f_1^2 - f_2^2 \leq 0 \\ c_2(x) = (1 + 0.6 \sin(t + 6l))^2 - f_1^2 - f_2^2 \leq 0 \end{cases} \\
& G(t) = |\sin(t)| \\
& g = 1 + 0.1 \left(\left| \sum_{i=2}^D x_i - G \right| \right)^{0.5} \\
& C(t) = |\sin(t)| \\
& l = 0.5\pi - 2 \left| \arctan \frac{f_2}{f_1} - 0.25\pi \right|
\end{aligned} \tag{5.14}$$

The changing DCPOF of DCF4 is always a part of the unchanged DUPOF. The multiple constraints make the changing feasible region disconnected. At the same time, the number of

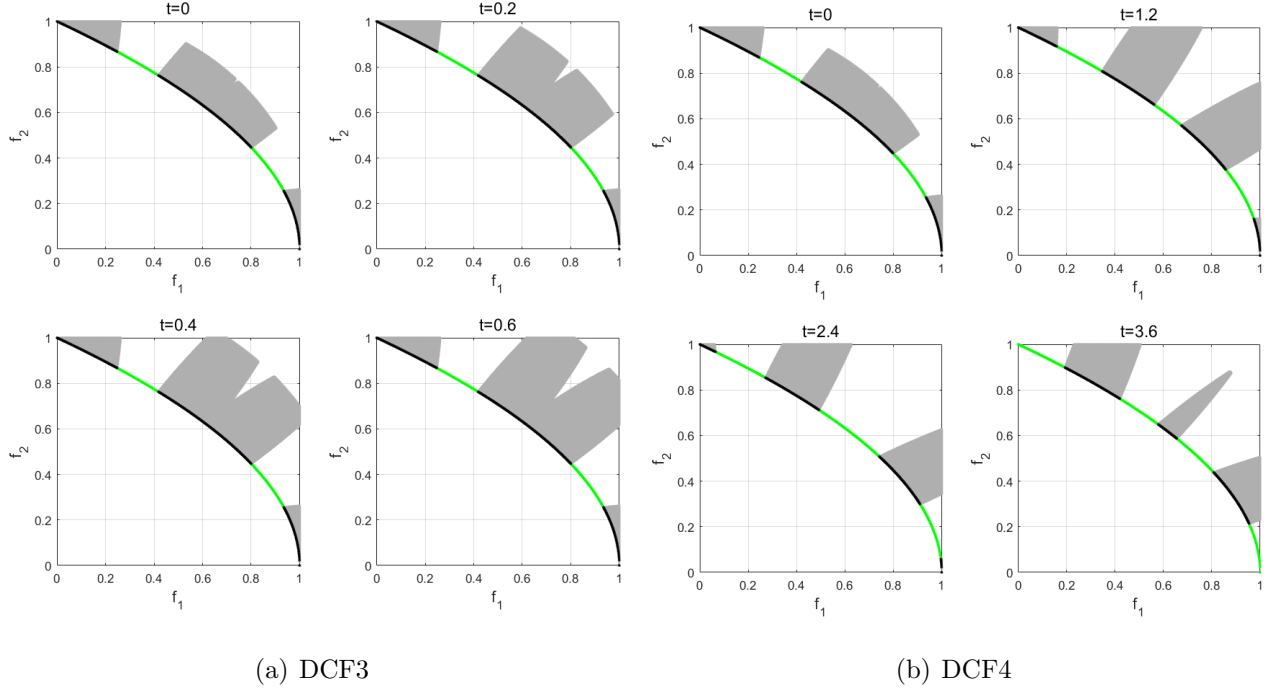


Figure 5.4: The objective space of (a) DCF3 and (b) DCF4.

disconnected feasible regions changes over time. The search space is $[0, 1]^D$. DCF4 is plotted in Fig. 5.4 (b).

DCF5:

$$\begin{aligned}
 \min & \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{0.5 + G - f_1} \end{cases} \\
 s.t. & \begin{cases} c_1(x) = 1.1 + C - f_1^2 - f_2^2 \leq 0 \\ c_2(x) = (1 + 0.6 \sin(t + 6l))^2 - f_1^2 - f_2^2 \leq 0 \end{cases} \\
 & G(t) = 0.5|\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & C(t) = 0.5|\sin(t)| \\
 & l = 0.5\pi - 2|\arctan \frac{f_2}{f_1} - 0.25\pi|
 \end{aligned} \tag{5.15}$$

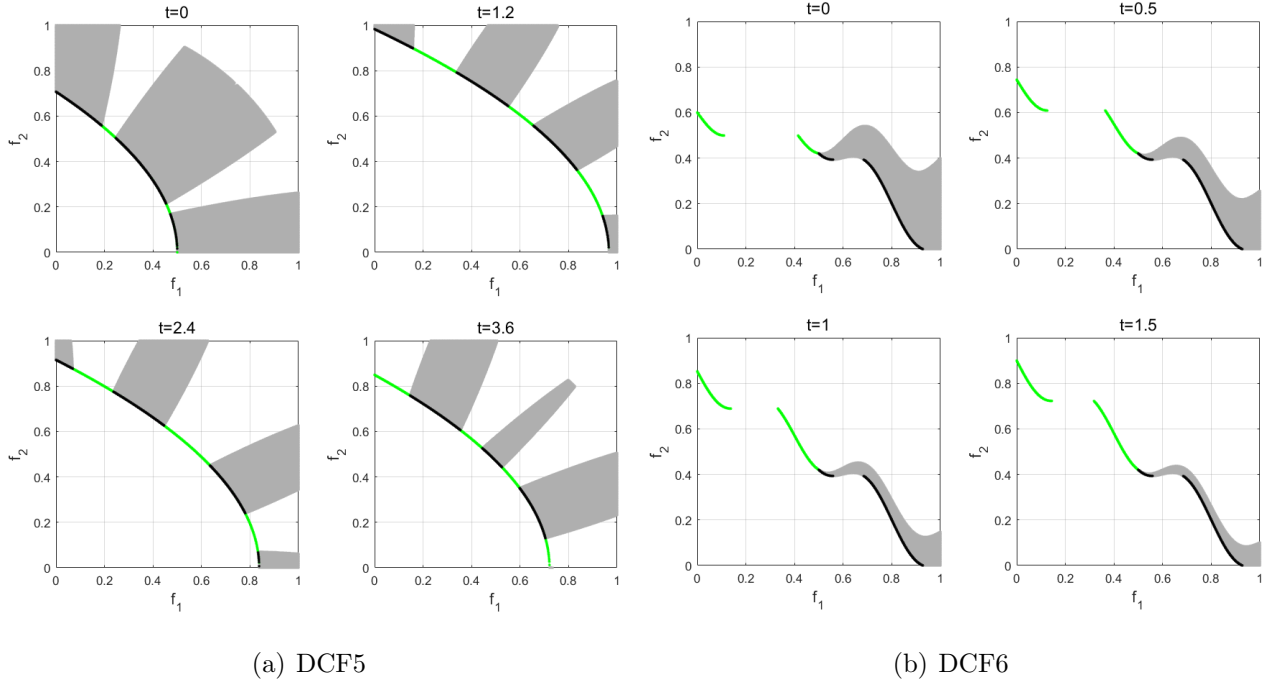


Figure 5.5: The objective space of (a) DCF5 and (b) DCF6.

The changing DCPOF of DCF5 is always a part of the changing DUPOF. The multiple constraints make the changing feasible region disconnected. Similar to DCF4, the number of disconnected feasible regions changes over time, as shown in Fig. 5.5 (a). DCF5 has a changing search space $[0, 0.5 + G]^D$.

DCF6:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = \begin{cases} g(1 - f_1 - 0.08\sin(5\pi f_1)) & f_1 \geq 0.5 \\ g(G - kf_1 - 0.08\sin(5\pi f_1)) & f_1 < 0.5 \end{cases} \end{cases} \\
 & \text{s.t.} \begin{cases} c_1(x) = C - kf_1 - f_2 - l \leq 0 \\ c_2(x) = 1 - f_1 - f_2 - l \leq 0 \end{cases} \\
 & G(t) = 0.6 + 0.3|\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & k = 2G - 1 \\
 & C(t) = 0.6 + 0.3|\sin(t)| \\
 & l = 0.08\sin(5\pi f_1)
 \end{aligned} \tag{5.16}$$

The DUPOF of DCF6 is partially changed, and the DCPOF remains the same location and is a part of the changing DUPOF. The feasible region of DCF6 can become very narrow, as shown in Fig. 5.5 (b). DCF6 has a changing search space $[0, 0.5 + G]^D$.

DCF7:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{1 - f_1^2} \end{cases} \\
 & \text{s.t.} \begin{cases} c_1(x) = 1.1 - f_1^2 - f_2^2 + 0.125\sin(0.1 + 2\pi l)^4 \geq 0 \\ c_2(x) = 1 - f_1^2 - f_2^2 + 0.125\sin(0.1 + 2\pi l)^4 \leq 0 \end{cases} \\
 & G(t) = |\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & l = \sqrt{2}f_2 - \sqrt{2}f_1
 \end{aligned} \tag{5.17}$$

The DCPOF of DCF7 consists of a part of the DUPOF and a part of the boundary of the feasible region, as shown in Fig. 5.6 (a). DCF7 belongs to Type I DCMOP. That is, the DCPOF and DUPOF are unchanged. Moreover, DCF7 has a narrow, connected, feasible region. The search space is $[0, 1]^D$.

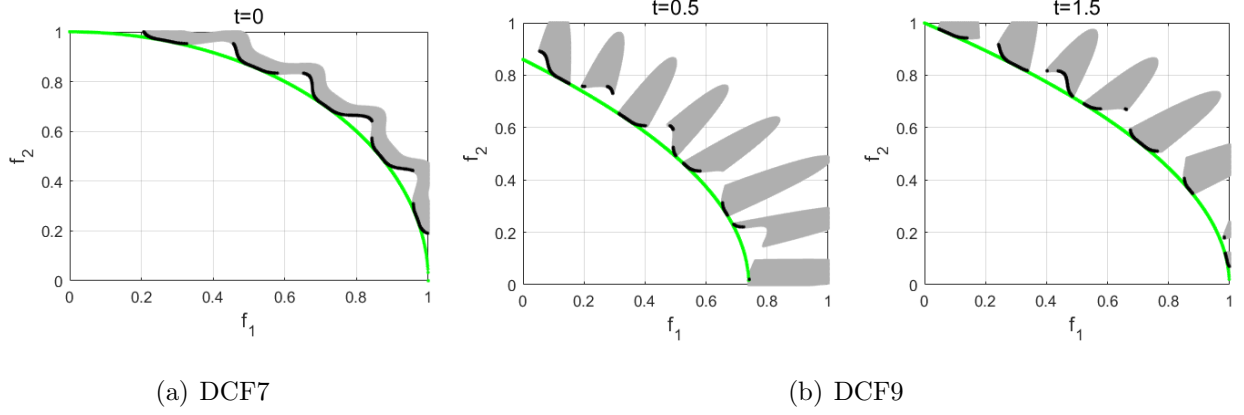


Figure 5.6: The objective space of (a) DCF7 and (b) DCF9.

DCF8:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(1 - f_1) \end{cases} \\
 & s.t. \begin{cases} c_1(x) = f_1 + f_2 - 1.4 - \frac{\sin(t)\sin(t+l)^6}{2} \leq 0 \\ c_2(x) = 0.95 - f_1 - f_2 + 0.235\sin(t+l)^2 \leq 0 \end{cases} \\
 & G(t) = |\sin(t)| \\
 & g = 1 + 0.1 \left(\left| \sum_{i=2}^D x_i - G \right| \right)^{0.5} \\
 & l = 0.75\pi(\sqrt{2}f_2 - \sqrt{2}f_1)
 \end{aligned} \tag{5.18}$$

The DCPOF of DCF8 consists of a part of the DUPOF and a part of the boundary of the feasible region. In addition, The connectivity of the feasible region changes over time, and DCF8 contains a narrow feasible region, as shown in Fig. 5.7 (a). The search space is $[0, 1]^D$.

DCF9:

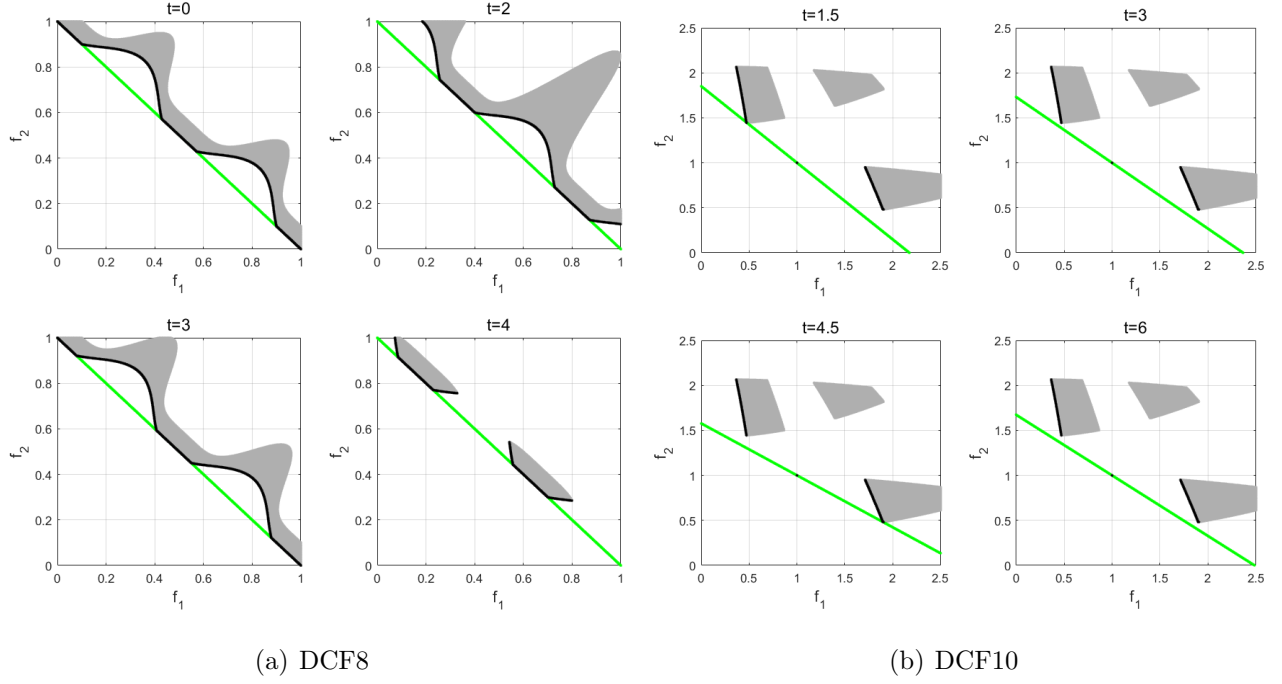


Figure 5.7: The objective space of (a) DCF8 and (b) DCF10.

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g\sqrt{0.5 + G - f_1} \end{cases} \\
 & s.t. \begin{cases} c_1(x) = G + 0.7 - f_1 - f_2^2 + 0.3\sin(10\pi l_1) \leq 0 \\ c_2(x) = G + 0.5 - f_1 - f_2^2 + 0.11\sin(t + 2\pi l_2) \leq 0 \end{cases} \\
 & G(t) = 0.5|\sin(t)| \\
 & g = 1 + 0.1\left(\left|\sum_{i=2}^D x_i - G\right|\right)^{0.5} \\
 & l_1 = \arctan \frac{f_2}{f_1} \\
 & l_2 = \sqrt{2}f_2 - \sqrt{2}f_1
 \end{aligned} \tag{5.19}$$

The DCPOF of DCF9 consists of a part of the DUPOF and a part of the boundary of the feasible region. DCF9 is a Type IV DCMOP. The DUPOF and DCPOF are changing over time, as shown in Fig. 5.6 (b). The changing search space is $[0, 0.5 + G]^D$.

DCF10:

$$\begin{aligned}
& \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(G + kf_1) \end{cases} \\
& \text{s.t.} \begin{cases} c_1(x) = -(3 - 0.7f_1^2 - f_2)^{l_1} \leq 0 \\ c_2(x) = (3 - 0.325f_1^2 - f_2)^{l_2} \leq 0 \\ c_3(x) = -(1.42 + 0.11f_1^2 - f_2)^{l_3} \leq 0 \\ c_4(x) = (2.07 - 0.03f_1^2 - f_2)^{l_4} \leq 0 \end{cases} \\
& G(t) = 1.712565 + 0.138755 \sin(t) \\
& k = 1 - G \\
& g = 1 + 0.1 \left(\left| \sum_{i=2}^D x_i - G \right| \right)^{0.5} \\
& l_1 = 3 - 2f_1^2 - f_2 \\
& l_2 = 3 - 7f_1^2 - f_2 \\
& l_3 = 1.025 - 0.025f_1^2 - f_2 \\
& l_4 = 0.3 + 0.05f_1^2 - f_2
\end{aligned} \tag{5.20}$$

The DCPOF of DCF10 consists of an isolated point ($f_1 = 1, f_2 = 1$) located on the DUPOF and a part of the boundary of the feasible region. The distance between DUPOF and DCPOF changes over time, as shown in Fig. 5.7 (b). DCF10 has four constraints, and these constraints make the gap between different disconnected feasible regions huge. The changing search space is $[0, 0. - G/k]^D$.

DCF11:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(0.95 - 0.8f_1 - 0.08|\sin(3.2\pi f_1)|) \end{cases} \\
 & \text{s.t.} \begin{cases} c_1(x) = (1 - 0.8f_1 - f_2 + l_1)^{1.8-1.125f_1-f_2+l_2} \leq 0 \\ c_2(x) = (1 - 0.625f_1 - f_2 + l_3)^{1.4-0.875f_1-f_2+l_4} \geq 0 \end{cases} \\
 & G(t) = |\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & l_1 = 0.08 \sin(2\pi(f_2 - f_1/1.5)) \\
 & l_2 = 0.08 \sin(2\pi(f_2/1.8 - f_1/1.6)) \\
 & l_3 = 0.08 \sin(2\pi(f_2 - f_1/1.6)) \\
 & l_4 = 0.08 \sin(2\pi(f_2/1.4 - f_1/1.6))
 \end{aligned} \tag{5.21}$$

The objective space of DCF11 is given in Fig. 5.8 (a). DCF11 is derived from MW12 [162] and belongs to Type I DCMOP. The DUPOF of DCF11 is entirely located outside the feasible region. Thus, the DCPOF is composed of a part of the boundary of the feasible region. DCF11 has a disconnected feasible region. The search space of DCF11 is $[0, 1]^D$.

DCF12:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(0.95 - 0.8f_1 - 0.08|\sin(3.2\pi f_1)|) \end{cases} \\
 & \text{s.t.} \begin{cases} c_1(x) = (1 - 0.8f_1 - f_2 + l_1)^{1.8-1.125f_1-f_2+l_2} \leq 0 \\ c_2(x) = \begin{cases} (l_3)^{l_4} \geq 0 & \text{if } \sin(5\pi t) = 0 \\ (l_3)^{l_4} < 0 & \text{otherwise} \end{cases} \end{cases} \\
 & G(t) = |\sin(t)| \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & l_1 = 0.08 \sin(2\pi(f_2 - f_1/1.5)) \\
 & l_2 = 0.08 \sin(2\pi(f_2/1.8 - f_1/1.6)) \\
 & l_3 = 1 - 0.625f_1 - f_2 + 0.08 \sin(2\pi(f_2 - f_1/1.6)) \\
 & l_4 = 1.4 - 0.875f_1 - f_2 + 0.08 \sin(2\pi(f_2/1.4 - f_1/1.6))
 \end{aligned} \tag{5.22}$$

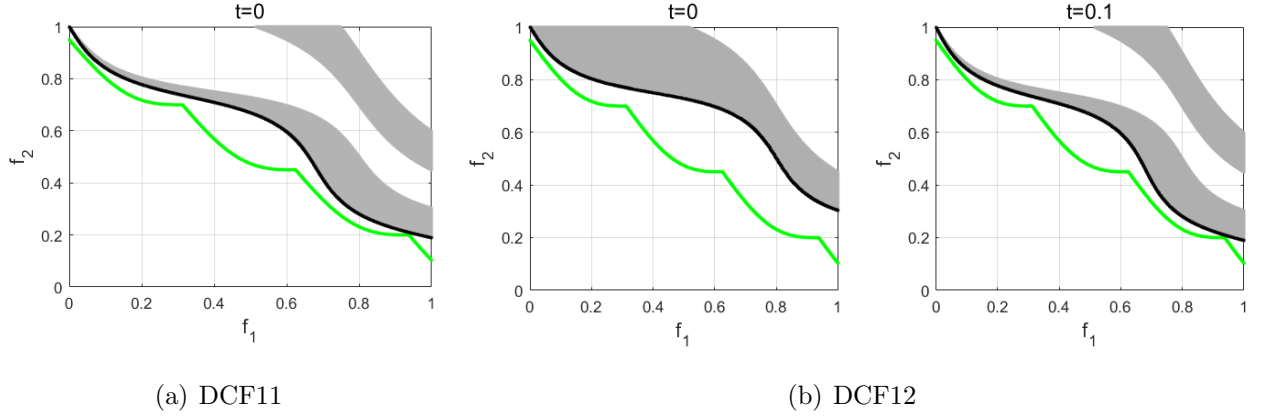


Figure 5.8: The objective space of (a) DCF11 and (b) DCF12.

DCF12 is derived from DCF11 and belongs to Type II DCMOP. The connectivity of DCF12 changes over time. To be specific, the connectivity is from connected to disconnected to connected, as shown in Fig. 5.8 (b). In particular, the feasible regions of two consecutive time steps are non-overlapping and complementary, i.e., the feasible region at $t = 0$ changes to an infeasible region at $t = 1$. The search space of DCF12 is $[0, 1]^D$.

DCF13:

$$\begin{aligned}
 & \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(G - 0.8f_1 - 0.08|\sin(3.2\pi f_1)|) \end{cases} \\
 & s.t. \begin{cases} c_1(x) = (1 - 0.8f_1 - f_2 + l_1)^{2G-1.125f_1-f_2+l_2} \leq 0 \\ c_2(x) = (G + 0.15 - 0.625f_1 - f_2 + l_3)^{l_4} \geq 0 \end{cases} \\
 & G(t) = 0.85 + 0.05 \sin(t) \\
 & g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
 & l_1 = 0.08 \sin(2\pi(f_2 - f_1/1.5)) \\
 & l_2 = 0.08 \sin(2\pi(f_2/1.8 - f_1/1.6)) \\
 & l_3 = 0.08 \sin(2\pi(f_2 - f_1/1.6)) \\
 & l_4 = 1.4 - 0.875f_1 - f_2 + 0.08 \sin(2\pi(f_2/1.4 - f_1/1.6))
 \end{aligned} \tag{5.23}$$

DCF13 is also derived from DCF11 and belongs to Type IV DCMOP, i.e., the DUPOF and

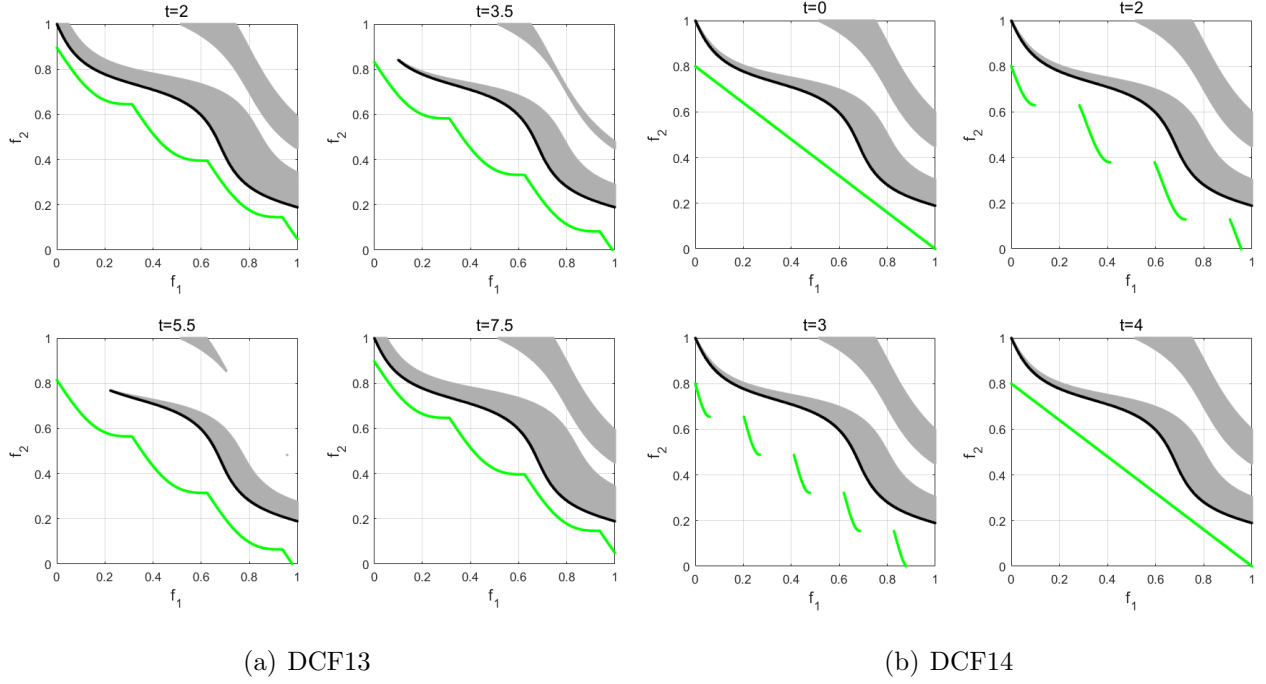


Figure 5.9: The objective space of (a) DCF13 and (b) DCF14.

DCPOF both change over time, and the DUPOF is always outside of the feasible region. As shown in Fig. 5.9 (a), the size of two unconnected feasible regions will change over time. The search space of DCF13 is $[0, 1]^D$.

DCF14:

$$\begin{aligned}
& \min \begin{cases} f_1(x) = x_1 \\ f_2(x) = g(0.8 - 0.8f_1 - 0.1\sin(3.2\pi f_1(t \bmod 4))) \end{cases} \\
& \text{s.t.} \begin{cases} c_1(x) = (1 - 0.8f_1 - f_2 + l_1)^{1.8-1.125f_1-f_2+l_2} \leq 0 \\ c_2(x) = (1 - 0.625f_1 - f_2 + l_3)^{1.4-0.875f_1-f_2+l_4} \geq 0 \end{cases} \\
& G(t) = 0.85 + 0.05 \sin(t) \\
& g = 1 + 0.1(|\sum_{i=2}^D x_i - G|)^{0.5} \\
& l_1 = 0.08 \sin(2\pi(f_2 - f_1/1.5)) \\
& l_2 = 0.08 \sin(2\pi(f_2/1.8 - f_1/1.6)) \\
& l_3 = 0.08 \sin(2\pi(f_2 - f_1/1.6)) \\
& l_4 = 0.08 \sin(2\pi(f_2/1.4 - f_1/1.6))
\end{aligned} \tag{5.24}$$

Similarly, DCF14 is derived from DCF11 and belongs to Type III DCMOP. The DUPOF will change from a linear geometry to a disconnected geometry over time, as shown in Fig. 5.9 (b). The search space of DCF14 is $[0, 1]^D$.

5.4 Experiments

In this section, comprehensive numerical experiments are conducted on the proposed test suite and a real-world application to examine the performance of the proposed DC-MFEA. In addition, ablation studies and sensitivity studies are conducted to verify the effectiveness of the proposed DC-MFEA.

5.4.1 Experimental Settings

Compared Algorithms and Experimental Settings. The algorithms used for comparison are as follows:

1. TDCEA [165]: A two-stage based dynamic constrained multiobjective evolutionary algo-

rithm, where the first stage randomly introduces some individuals and the second stage adaptively disturbs these solutions to further improve the diversity.

2. dCMOEA [66]: An adaptive penalty function-based dynamic constrained multiobjective evolutionary algorithm, where a memory scheme is designed to reinitialize the partial solutions for the new environment.
3. DC-MOEA [64]: An adaptive penalty function-based dynamic constrained multiobjective evolutionary algorithm, where a feasibility-driven strategy is designed to guide the search toward the new feasible regions.
4. CTAEA [166]: A two-archive-based evolutionary algorithm for CMOPs. We use the variant proposed for DCMOPs [81], where two archives are used for maintaining both convergence and diversity.
5. KTNSGA2 [53]: A knowledge transfer-based dynamic multiobjective evolutionary algorithm, where an ensemble learning framework is devised to predict the knee points in the new environment.
6. MTCMO [95]: An evolutionary multitasking algorithm for CMOPs, in which an auxiliary task is created to assist in solving a CMOP.

Because KTNSGA2 was initially proposed for DMOPs, we incorporate the CDP strategy with it to endow it with the ability to solve DCMOPs. MTCMO and CTAEA are originally proposed for CMOPs. Following existing work [81], MTCMO and CTAEA incorporate the dynamic response strategy in DNSGA2 [3] for a fair comparison, i.e., the $\xi = 20\%$ individuals are replaced with new randomly created solutions. The parameters are set as follows:

1. Population size and the number of decision variables: the population size is set to 100 for all problems. The dimension D of decision variables for all DCFs is set to 10.

2. Dynamic test settings: The dynamic setting of Eq. (5.10) comprises three pairs of change characteristics, $(n_t = 1, \tau_t = 20)$, $(n_t = 10, \tau_t = 20)$ and $(n_t = 10, \tau_t = 10)$ for each problem. The total number of evaluations τ is fixed to $30 \cdot \tau_t$, which ensures that there are 30 changes in each run. All the problems are run 10 times independently.
3. Parameters of genetic operators for all algorithms: SBX crossover: $p_c = 0.5$, $\eta_c = 1$. Polynomial mutation: $p_m = 0.1$, $\eta_m = 1$.
4. Parameters of DC-MFEA: rpm is set to 0.9, and the population sizes for main task N_{T1} and auxiliary task N_{T2} are set to 80 and 20, respectively.

Performance Indicators. In the experimental studies, a variant of inverted generational distance (IGD), called MIGD, and a variant of hypervolume (HV), called MHV, are used to evaluate the performance of the compared algorithms. In the following section, the concepts of IGD and HV are given.

1) IGD quantifies the convergence of solutions obtained by the algorithm. The mathematical form of IGD is presented as follows:

$$IGD(POF^*, POF) = \frac{1}{n} \sum_{p^* \in POF^*} \min_{p \in POF} \|p^* - p\|^2, \quad (5.25)$$

where POF^* is the true POF of the multiobjective optimization problem, POF is an approximation set of POF^* obtained by an optimization algorithm and n is the number of solutions in the POF^* .

The MIGD metric is introduced to measure the convergence of solutions, it is calculated as the average of the IGD values across multiple time steps. A lower MIGD value indicates an improvement in the convergence of the solutions.

$$MIGD(POF^*, POF) = \frac{1}{|T|} \sum_{t \in T} IGD(POF^*, POF_t), \quad (5.26)$$

where T is a set of discrete time steps in a single run and $|T|$ is the cardinality of T .

2) HV quantifies both the convergence and diversity of solutions obtained by the algorithm.

The HV is defined as follows:

$$HV(POF, ref) = \Lambda\left(\bigcup_{p \in POF} \{p' | ref \succ p' \succ p\}\right), \quad (5.27)$$

where Λ is the Lebesgue measure and $ref \in R^M$ is a set of user-defined reference points.

MHV is calculated as the average of the HV values across a series of time steps.

$$MHV(POF, ref) = \frac{1}{|T|} \sum_{t \in T} HV(POF_t, ref). \quad (5.28)$$

5.4.2 Comparisons with Baselines

Table 5.2 and Table 5.3 present the statistical results of the average MIGD and MHV values over 10 runs. In these tables, the symbols (-), (=), and (+) represent the statistical significance levels based on the Wilcoxon test with a significance level of 0.05. Specifically, (-) indicates that DC-MFEA is significantly worse, (=) implies indifference, and (+) signifies that DC-MFEA is significantly better than its counterparts.

The experimental results presented in Table 5.2 reveal that DC-MFEA consistently outperforms the compared algorithms in terms of MIGD values across DCF1 and DCF3~DCF9 under all dynamic test settings. However, for DCF10~DCF13, DC-MFEA demonstrates slightly inferior performance compared to the best-performing algorithms.

Fig. 5.10 depicts the convergence traces of IGD values for all environmental steps obtained by the compared algorithms to visualize the performance further. The convergence traces of the proposed DC-MFEA are consistently stable and positioned at the bottom in most cases.

Table 5.3 provides the experimental results for the MHV values. Furthermore, we plot the HV values obtained by various compared algorithms after each change, as shown in Fig. 5.11.

Table 5.2: Mean and standard deviation of MIGD values obtained by compared algorithms over DCF test functions.

Problem	n_t, τ_t	DC-MFEA	MTCMO	DC-MOEA	KT-NSGA2	dCMOEA	TDCEA	CTAEA
DCF1	1,20	2.57e-02(4.58e-06)	9.73e-02(1.72e-03)+	1.37e-01(5.00e-05)+	1.55e-01(2.07e-05)+	5.13e-01(1.33e-02)+	6.10e-02(2.18e-05)+	9.08e-02(8.67e-06)+
	10,20	4.41e-02(8.53e-07)	7.33e-02(5.82e-04)+	9.60e-02(2.72e-05)+	1.24e-01(9.84e-06)+	5.16e-01(7.29e-03)+	6.18e-02(6.59e-05)+	5.96e-02(5.77e-06)+
	10,10	6.07e-02(3.30e-06)	9.12e-02(9.91e-04)=	9.75e-02(1.40e-05)+	1.23e-01(2.25e-05)+	2.43e-01(1.43e-03)+	7.74e-02(9.21e-05)+	7.10e-02(5.72e-06)+
DCF2	1,20	4.96e-02(7.05e-09)	5.43e-02(4.60e-04)=	6.07e-02(5.39e-07)+	6.56e-02(2.28e-06)+	2.86e-01(5.63e-04)+	6.07e-02(2.68e-06)+	4.84e-02(3.76e-09)-
	10,20	4.45e-02(5.38e-09)	4.89e-02(1.89e-04)=	5.66e-02(9.87e-08)+	6.07e-02(6.16e-07)+	2.91e-01(8.60e-06)+	5.08e-02(7.08e-06)+	4.36e-02(9.35e-09)-
	10,10	4.54e-02(4.18e-09)	5.77e-02(2.62e-04)=	6.26e-02(1.78e-06)+	6.01e-02(8.88e-07)+	2.69e-01(4.14e-05)+	5.47e-02(4.48e-06)+	4.62e-02(4.47e-07)+
DCF3	1,20	2.55e-02(8.16e-07)	3.08e-02(3.60e-05)=	5.22e-02(5.86e-06)+	7.98e-02(6.97e-06)+	2.88e-01(1.43e-03)+	2.96e-02(5.04e-06)+	2.76e-02(2.21e-07)+
	10,20	1.38e-02(2.20e-07)	3.43e-02(1.32e-04)+	5.18e-02(8.28e-07)+	7.58e-02(9.54e-06)+	2.91e-01(1.01e-03)+	2.24e-02(6.98e-06)+	2.56e-02(1.32e-07)+
	10,10	2.08e-02(1.36e-07)	3.94e-02(4.75e-05)+	5.76e-02(2.84e-06)+	7.51e-02(3.04e-06)+	2.68e-01(3.24e-04)+	2.96e-02(3.83e-06)+	4.02e-02(4.67e-07)+
DCF4	1,20	2.40e-02(2.07e-07)	3.78e-02(6.38e-05)+	7.92e-02(1.57e-04)+	7.62e-02(5.08e-06)+	3.81e-01(1.95e-03)+	2.90e-02(6.12e-06)+	2.82e-02(3.73e-07)+
	10,20	1.31e-02(1.80e-08)	3.56e-02(2.10e-05)+	8.61e-02(2.42e-04)+	7.60e-02(3.32e-06)+	3.88e-01(3.29e-03)+	2.29e-02(4.77e-07)+	2.70e-02(6.75e-07)+
	10,10	1.98e-02(2.08e-07)	4.32e-02(1.06e-04)+	6.48e-02(5.44e-06)+	7.62e-02(3.05e-06)+	3.13e-01(4.20e-03)+	3.38e-02(4.15e-05)+	4.07e-02(7.80e-07)+
DCF5	1,20	1.12e-02(1.21e-08)	1.46e-02(3.54e-05)=	4.08e-02(4.16e-06)+	3.67e-02(1.67e-06)+	2.29e-01(4.23e-04)+	1.41e-02(1.26e-07)+	1.12e-02(2.11e-08)=
	10,20	9.72e-03(1.93e-08)	1.36e-02(1.59e-05)=	4.09e-02(1.02e-05)+	3.66e-02(1.25e-06)+	2.32e-01(2.89e-04)+	1.20e-02(1.65e-07)+	1.06e-02(1.64e-08)+
	10,10	1.17e-02(6.50e-08)	1.65e-02(2.34e-05)=	5.00e-02(1.30e-05)+	3.55e-02(2.84e-06)+	1.97e-01(4.52e-05)+	1.60e-02(4.46e-06)+	1.41e-02(1.29e-07)+
DCF6	1,20	1.06e-02(2.11e-06)	5.50e-02(2.60e-04)+	2.89e-02(2.69e-06)+	4.44e-02(2.37e-06)+	2.15e-01(1.17e-03)+	2.33e-02(9.76e-06)+	3.87e-02(4.05e-06)+
	10,20	6.67e-03(6.68e-09)	4.89e-02(4.97e-04)+	3.07e-02(5.15e-07)+	4.50e-02(3.03e-06)+	1.91e-01(9.39e-04)+	2.30e-02(1.03e-05)+	3.91e-02(1.35e-05)+
	10,10	8.19e-03(1.71e-07)	6.27e-02(1.17e-04)+	3.44e-02(3.15e-07)+	4.46e-02(2.80e-06)+	1.11e-01(3.14e-04)+	2.04e-02(2.57e-06)+	4.84e-02(1.10e-05)+
DCF7	1,20	4.58e-02(1.62e-05)	1.01e-01(5.20e-04)+	8.44e-02(6.23e-06)+	1.31e-01(8.27e-05)+	1.91e-01(1.60e-04)+	6.20e-02(1.80e-05)+	7.83e-02(8.05e-06)+
	10,20	2.92e-02(6.65e-07)	9.42e-02(6.36e-04)+	8.24e-02(1.51e-05)+	1.25e-01(5.38e-05)+	1.81e-01(1.56e-04)+	4.93e-02(5.56e-06)+	7.46e-02(2.56e-05)+
	10,10	3.41e-02(2.43e-06)	1.31e-01(6.69e-04)+	9.01e-02(2.55e-05)+	1.25e-01(2.52e-05)+	1.74e-01(6.89e-05)+	8.29e-02(9.73e-06)+	1.17e-01(1.83e-04)+
DCF8	1,20	7.72e-02(2.19e-06)	9.05e-02(2.06e-03)=	9.19e-02(3.45e-05)+	1.03e-01(2.16e-05)+	1.64e-01(5.56e-05)+	1.07e-01(1.28e-04)+	8.62e-02(3.11e-05)+
	10,20	8.01e-02(9.36e-07)	1.15e-01(6.67e-04)+	9.17e-02(1.13e-05)+	1.12e-01(2.45e-05)+	1.72e-01(1.99e-04)+	9.33e-02(3.89e-05)+	8.44e-02(1.36e-05)+
	10,10	7.78e-02(1.13e-06)	1.36e-01(1.78e-03)+	9.70e-02(1.41e-04)+	1.18e-01(2.04e-05)+	1.52e-01(5.67e-05)+	9.16e-02(1.20e-04)+	1.01e-01(3.55e-05)+
DCF9	1,20	2.50e-02(3.33e-07)	9.27e-02(1.28e-03)+	3.77e-02(2.00e-06)+	4.42e-02(5.09e-06)+	2.23e-01(8.52e-05)+	5.12e-02(3.55e-05)+	7.17e-02(5.01e-05)+
	10,20	2.44e-02(2.67e-07)	7.08e-02(1.35e-03)+	4.01e-02(3.21e-06)+	4.42e-02(2.74e-06)+	2.08e-01(2.17e-04)+	5.49e-02(4.89e-05)+	6.54e-02(2.93e-05)+
	10,10	2.43e-02(3.47e-07)	1.06e-01(5.02e-04)+	4.73e-02(1.10e-05)+	4.49e-02(3.04e-06)+	2.04e-01(2.75e-04)+	5.19e-02(4.12e-05)+	7.61e-02(2.46e-05)+
DCF10	1,20	6.50e-01(2.02e-03)	8.90e-01(9.38e-02)=	6.21e-01(1.33e-03)=	7.59e-01(9.58e-02)+	9.10e-01(3.09e-03)+	8.39e-01(1.56e-02)=	6.96e-01(1.94e-03)=
	10,20	6.42e-01(1.12e-03)	7.53e-01(2.43e-02)=	6.76e-01(1.98e-03)=	6.30e-01(1.53e-01)=	8.99e-01(2.16e-03)+	7.44e-01(2.84e-03)+	6.84e-01(6.09e-03)=
	10,10	6.56e-01(3.85e-03)	9.21e-01(6.67e-02)=	6.65e-01(1.27e-03)=	5.48e-01(2.07e-01)=	9.11e-01(9.73e-04)+	7.65e-01(4.36e-03)=	6.98e-01(3.12e-03)=
DCF11	1,20	3.21e-01(2.51e-04)	1.90e-01(3.44e-03)=	2.73e-01(4.86e-04)=	3.50e-01(1.03e-03)+	5.56e-01(6.08e-03)+	1.74e-01(1.27e-04)+	1.72e-01(1.91e-04)=
	10,20	2.45e-01(6.65e-04)	2.33e-01(3.16e-03)=	2.64e-01(7.23e-04)+	3.79e-01(1.04e-03)+	5.09e-01(8.26e-03)+	2.01e-01(3.54e-04)+	1.82e-01(2.08e-04)=
	10,10	3.31e-01(8.18e-04)	2.23e-01(1.75e-03)=	2.61e-01(7.41e-04)=	3.60e-01(6.51e-04)+	4.43e-01(7.24e-04)+	2.15e-01(1.30e-03)=	1.95e-01(1.48e-03)=
DCF12	1,20	7.71e-01(2.70e-04)	3.11e-01(1.35e-02)-	5.21e-01(4.38e-04)-	6.57e-01(2.16e-04)=	6.52e-01(1.00e-03)=	6.09e-01(1.11e-03)-	2.66e-01(3.48e-04)-
	10,20	5.90e-01(1.31e-03)	3.33e-01(1.04e-02)-	5.25e-01(2.35e-04)-	6.23e-01(5.38e-04)=	5.80e-01(4.55e-03)-	6.40e-01(1.75e-03)=	2.50e-01(7.30e-04)-
	10,10	5.99e-01(2.44e-03)	3.95e-01(1.29e-02)-	5.05e-01(8.40e-05)-	6.04e-01(3.40e-04)=	4.02e-01(5.79e-04)-	6.47e-01(1.13e-03)=	3.04e-01(1.01e-03)-
DCF13	1,20	4.11e-01(2.47e-04)	4.19e-01(3.20e-02)=	3.38e-01(2.52e-04)-	3.47e-01(5.63e-04)-	3.54e-01(5.16e-04)-	4.46e-01(2.27e-03)=	2.98e-01(2.02e-03)-
	10,20	4.41e-01(8.01e-04)	3.51e-01(1.41e-02)=	3.62e-01(5.75e-04)-	3.53e-01(2.12e-04)-	4.04e-01(1.65e-03)=	4.42e-01(1.92e-03)=	2.86e-01(1.01e-03)-
	10,10	4.53e-01(1.63e-04)	2.96e-01(3.44e-03)-	3.39e-01(9.31e-05)-	3.70e-01(1.77e-04)-	3.79e-01(1.40e-03)-	4.85e-01(1.84e-04)+	3.32e-01(6.96e-04)-
DCF14	1,20	1.65e-01(5.64e-05)	2.29e-01(1.15e-03)=	1.78e-01(3.39e-04)=	2.81e-01(4.17e-04)+	4.57e-01(1.21e-02)+	2.05e-01(6.74e-04)=	1.76e-01(1.36e-04)=
	10,20	1.88e-01(5.18e-05)	3.34e-01(2.61e-03)+	2.00e-01(1.98e-04)=	1.93e-01(2.06e-04)=	3.35e-01(1.25e-03)+	2.14e-01(5.91e-04)=	2.21e-01(4.18e-04)+
	10,10	1.97e-01(7.94e-05)	2.53e-01(1.27e-02)=	1.86e-01(6.55e-05)=	2.08e-01(3.56e-04)=	2.86e-01(1.23e-03)+	2.50e-01(1.51e-03)+	2.40e-01(1.06e-03)+
+/-=-			19/19/4	28/8/6	32/7/3	36/2/4	32/9/2	26/8/8

Chapter 5. Knowledge-Driven Spatial-Temporal Knowledge Transfer for Dynamic Constrained Multiobjective Optimization

Table 5.3: Mean and standard deviation of MHV values obtained by compared algorithms over DCF test functions.

Problem	n_t, τ_t	DC-MFEA	MTCMO	DC-MOEA	KT-NSGA2	dCMOEA	TDCEA	CTAEA
DCF1	1,20	3.97e-01(7.60e-06)	2.75e-01(4.49e-04)+	2.75e-01(2.76e-06)+	2.69e-01(1.68e-05)+	1.56e-01(1.36e-04)+	3.24e-01(4.56e-04)+	3.17e-01(5.54e-07)+
	10,20	3.75e-01(1.82e-06)	3.46e-01(9.92e-05)+	3.12e-01(2.62e-06)+	2.61e-01(2.10e-05)+	1.72e-01(2.48e-04)+	3.23e-01(1.14e-03)+	3.56e-01(9.79e-07)+
	10,10	3.54e-01(8.45e-06)	3.14e-01(4.46e-04)+	3.07e-01(9.49e-07)+	2.61e-01(1.34e-05)+	2.31e-01(1.18e-04)+	3.40e-01(1.72e-04)+	3.34e-01(2.50e-06)+
DCF2	1,20	6.28e-01(2.49e-08)	5.70e-01(2.48e-03)+	6.08e-01(1.20e-06)+	6.02e-01(2.51e-06)+	3.84e-01(3.02e-04)+	5.85e-01(2.69e-03)=	6.31e-01(1.49e-08)-
	10,20	6.35e-01(1.60e-08)	5.70e-01(1.34e-03)+	6.13e-01(3.05e-07)+	6.08e-01(4.71e-07)+	3.81e-01(1.40e-06)+	5.91e-01(4.77e-04)+	6.36e-01(3.64e-08)-
	10,10	6.33e-01(1.39e-08)	5.48e-01(1.38e-03)+	6.03e-01(3.15e-06)+	6.09e-01(1.29e-06)+	4.00e-01(2.30e-05)+	5.28e-01(3.61e-03)+	6.32e-01(6.61e-07)+
DCF3	1,20	4.80e-01(9.63e-07)	4.27e-01(9.60e-04)+	4.39e-01(2.46e-06)+	3.99e-01(9.06e-06)+	2.77e-01(5.90e-05)+	4.36e-01(1.59e-04)+	4.77e-01(8.98e-07)+
	10,20	4.97e-01(5.24e-07)	4.46e-01(6.87e-04)+	4.40e-01(1.48e-06)+	3.99e-01(5.21e-06)+	2.81e-01(4.73e-04)+	4.13e-01(6.13e-04)+	4.80e-01(9.05e-07)+
	10,10	4.87e-01(3.31e-07)	4.16e-01(5.87e-04)+	4.30e-01(1.68e-06)+	4.01e-01(1.74e-06)+	3.03e-01(8.66e-05)+	4.16e-01(9.02e-04)+	4.57e-01(1.09e-06)+
DCF4	1,20	4.20e-01(4.32e-07)	3.85e-01(8.27e-04)+	3.59e-01(2.53e-05)+	3.41e-01(3.47e-06)+	2.13e-01(2.41e-04)+	3.64e-01(2.60e-03)=	4.13e-01(1.28e-06)+
	10,20	4.34e-01(2.23e-07)	3.38e-01(5.95e-05)+	3.54e-01(4.57e-05)+	3.39e-01(2.06e-06)+	1.98e-01(5.82e-04)+	3.95e-01(2.38e-04)+	4.13e-01(1.51e-06)+
	10,10	4.24e-01(1.62e-07)	3.69e-01(1.38e-04)+	3.58e-01(3.43e-06)+	3.38e-01(3.96e-06)+	2.17e-01(1.15e-03)+	3.75e-01(5.39e-04)+	3.92e-01(3.49e-06)+
DCF5	1,20	3.39e-01(2.62e-08)	3.02e-01(2.20e-04)+	2.90e-01(8.69e-06)+	2.99e-01(1.23e-06)+	1.47e-01(1.06e-04)+	3.11e-01(7.77e-04)=	3.39e-01(1.14e-07)=
	10,20	3.45e-01(5.75e-08)	3.17e-01(5.02e-04)+	2.93e-01(1.73e-05)+	3.02e-01(3.55e-06)+	1.49e-01(7.15e-05)+	3.22e-01(3.23e-06)+	3.44e-01(1.47e-07)+
	10,10	3.43e-01(2.44e-07)	2.96e-01(2.09e-04)+	2.81e-01(1.36e-05)+	3.04e-01(4.09e-06)+	1.69e-01(1.33e-05)+	3.20e-01(4.84e-04)=	3.39e-01(3.43e-07)+
DCF6	1,20	1.30e-01(3.29e-07)	1.10e-01(6.36e-05)+	1.21e-01(2.17e-07)+	1.14e-01(3.40e-07)+	4.82e-02(1.23e-04)+	1.15e-01(6.16e-05)+	1.17e-01(5.37e-07)+
	10,20	1.32e-01(5.78e-09)	1.00e-01(9.48e-06)+	1.20e-01(5.70e-08)+	1.14e-01(1.30e-07)+	5.53e-02(1.83e-04)+	1.13e-01(1.17e-04)+	1.17e-01(1.02e-06)+
	10,10	1.31e-01(4.15e-08)	9.93e-02(2.76e-05)+	1.18e-01(5.97e-08)+	1.14e-01(2.63e-07)+	8.65e-02(3.91e-05)+	1.16e-01(2.66e-05)+	1.13e-01(9.64e-07)+
DCF7	1,20	3.40e-01(1.17e-05)	2.64e-01(3.58e-04)+	3.05e-01(1.00e-06)+	2.60e-01(1.26e-05)+	2.30e-01(6.40e-05)+	2.86e-01(1.62e-04)+	2.97e-01(4.78e-06)+
	10,20	3.62e-01(6.91e-07)	2.71e-01(4.83e-04)+	3.06e-01(5.09e-06)+	2.62e-01(1.60e-05)+	2.33e-01(1.01e-04)+	2.97e-01(3.95e-04)+	3.01e-01(9.51e-06)+
	10,10	3.54e-01(2.66e-06)	2.45e-01(4.09e-04)+	2.99e-01(7.49e-06)+	2.61e-01(3.21e-06)+	2.40e-01(8.85e-06)+	3.11e-01(8.88e-04)=	2.71e-01(3.00e-05)+
DCF8	1,20	6.56e-01(2.67e-06)	5.63e-01(1.32e-03)+	6.38e-01(1.91e-05)+	6.16e-01(1.35e-05)+	5.55e-01(5.23e-05)+	5.17e-01(4.36e-03)+	6.41e-01(1.20e-05)+
	10,20	6.49e-01(1.76e-06)	5.60e-01(1.80e-03)+	6.30e-01(3.17e-06)+	6.04e-01(1.35e-05)+	5.42e-01(1.57e-04)+	5.79e-01(2.72e-03)+	6.32e-01(1.06e-05)+
	10,10	6.47e-01(3.42e-06)	5.55e-01(3.03e-03)+	6.23e-01(3.13e-05)+	6.01e-01(9.81e-06)+	5.63e-01(2.24e-05)+	5.29e-01(6.94e-04)+	6.13e-01(1.13e-05)+
DCF9	1,20	3.69e-01(4.41e-07)	3.20e-01(6.73e-05)+	3.35e-01(1.62e-05)+	3.28e-01(1.58e-05)+	1.88e-01(8.26e-06)+	3.31e-01(4.56e-04)+	3.41e-01(6.32e-06)+
	10,20	3.82e-01(3.79e-07)	3.28e-01(2.20e-04)+	3.44e-01(1.64e-05)+	3.39e-01(1.47e-05)+	1.95e-01(1.29e-05)+	3.27e-01(6.48e-04)+	3.55e-01(5.64e-06)+
	10,10	3.82e-01(4.93e-07)	3.00e-01(1.63e-04)+	3.28e-01(2.28e-05)+	3.37e-01(1.88e-05)+	1.92e-01(4.97e-05)+	3.37e-01(9.45e-05)+	3.39e-01(5.80e-06)+
DCF10	1,20	1.49e+00(1.25e-02)	1.30e+00(1.03e-03)=	1.68e+00(1.03e-02)-	6.81e-01(4.59e-03)+	1.16e+00(1.55e-03)+	1.20e+00(2.17e-02)=	1.42e+00(3.13e-03)=
	10,20	1.48e+00(3.67e-03)	1.27e+00(2.18e-02)+	1.57e+00(8.51e-03)-	6.81e-01(3.29e-03)+	1.23e+00(4.52e-04)+	1.27e+00(5.57e-03)+	1.41e+00(1.47e-02)=
	10,10	1.50e+00(1.27e-02)	1.14e+00(7.65e-03)+	1.57e+00(3.57e-03)=	5.87e-01(4.58e-03)+	1.21e+00(1.99e-03)+	1.29e+00(8.43e-03)=	1.26e+00(5.51e-03)+
DCF11	1,20	8.56e-01(5.60e-04)	9.56e-01(7.35e-03)-	5.26e-01(2.17e-04)+	7.91e-01(1.98e-03)=	4.74e-01(1.20e-02)+	6.91e-01(1.60e-02)=	1.06e+00(4.52e-04)-
	10,20	9.82e-01(1.85e-03)	9.36e-01(8.90e-03)-	5.35e-01(4.82e-04)+	7.53e-01(1.92e-03)=	5.44e-01(1.38e-02)+	6.70e-01(3.94e-03)=	1.05e+00(4.28e-04)-
	10,10	8.42e-01(1.62e-03)	8.75e-01(4.79e-03)-	5.38e-01(4.59e-04)+	7.77e-01(1.25e-03)=	6.39e-01(1.24e-03)=	6.60e-01(6.95e-04)=	1.00e+00(2.72e-03)-
DCF12	1,20	2.32e-01(1.22e-03)	1.11e+00(9.42e-03)-	4.28e-01(7.38e-04)-	4.30e-01(8.53e-04)-	4.62e-01(4.33e-03)-	5.40e-01(1.96e-02)-	1.30e+00(1.46e-03)-
	10,20	5.03e-01(4.75e-03)	1.05e+00(5.66e-04)-	4.25e-01(1.99e-04)=	4.28e-01(2.11e-03)=	5.49e-01(1.47e-02)-	5.15e-01(5.57e-03)-	1.21e+00(2.51e-03)-
	10,10	4.92e-01(8.89e-03)	9.89e-01(1.04e-02)-	4.39e-01(9.51e-05)=	4.60e-01(1.07e-03)=	8.61e-01(2.36e-03)-	5.32e-01(1.02e-02)-	1.07e+00(4.02e-03)-
DCF13	1,20	6.07e-01(5.28e-04)	6.41e-01(8.43e-03)=	6.76e-01(2.84e-04)-	6.21e-01(9.32e-04)=	6.01e-01(1.19e-03)=	5.53e-01(3.74e-03)=	7.29e-01(3.61e-03)-
	10,20	5.67e-01(1.59e-03)	6.37e-01(1.29e-03)=	6.41e-01(1.12e-03)-	6.18e-01(4.02e-04)-	5.46e-01(2.18e-03)=	6.08e-01(5.28e-03)=	7.44e-01(1.58e-03)-
	10,10	5.46e-01(3.65e-04)	5.50e-01(6.88e-04)=	6.64e-01(1.55e-04)-	5.94e-01(2.76e-04)-	5.81e-01(2.01e-03)=	5.16e-01(1.22e-04)+	6.46e-01(1.02e-03)-
DCF14	1,20	1.17e+00(1.68e-04)	1.02e+00(3.33e-03)+	1.13e+00(8.03e-04)=	9.15e-01(1.07e-03)+	6.14e-01(1.97e-02)+	1.10e+00(5.46e-03)=	1.10e+00(5.20e-04)+
	10,20	1.16e+00(1.73e-04)	9.46e-01(5.09e-03)+	1.12e+00(4.68e-04)+	1.07e+00(4.90e-04)+	8.02e-01(2.90e-03)+	9.81e-01(4.77e-03)+	1.05e+00(1.63e-03)+
	10,10	1.14e+00(3.36e-04)	8.37e-01(6.15e-03)+	1.13e+00(2.10e-04)=	1.05e+00(1.08e-03)+	8.78e-01(3.87e-03)+	9.94e-01(5.50e-03)+	9.58e-01(3.25e-03)+
+/-			32/4/6	31/5/6	33/6/3	35/4/3	26/13/3	28/3/11

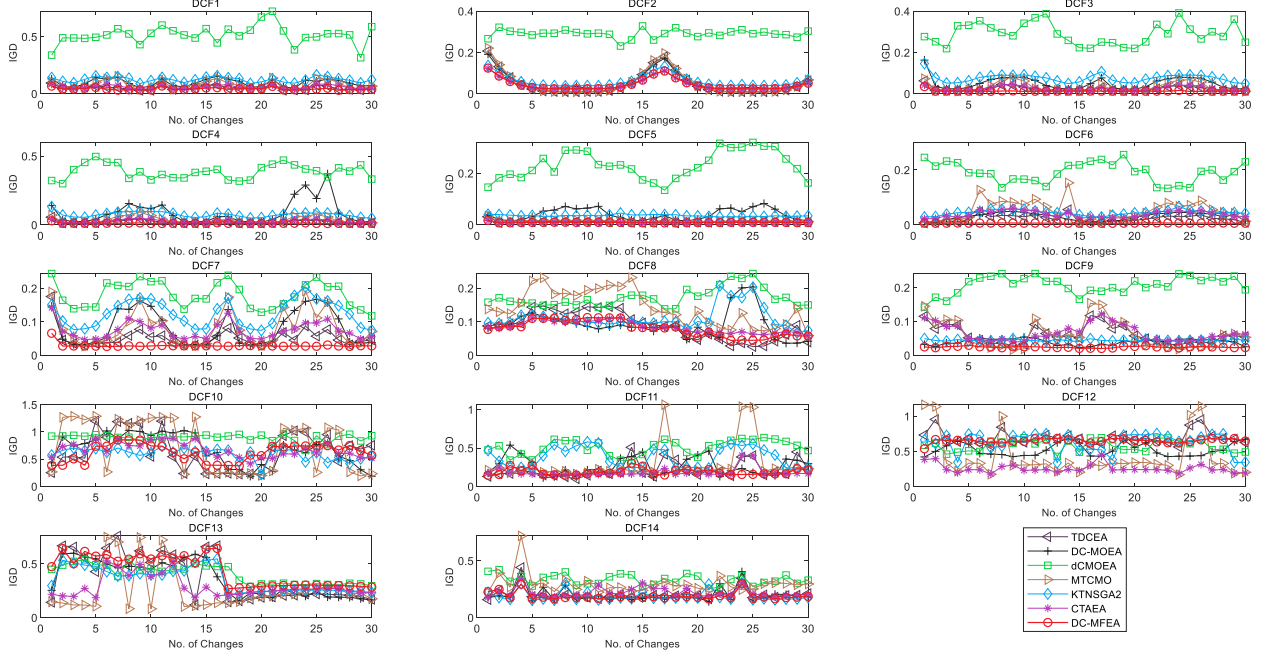


Figure 5.10: Convergence traces of average IGD values for all environmental steps obtained by compared algorithms with the dynamic test setting ($n_t = 10, \tau_t = 20$).

The results indicate a significant superiority of DC-MFEA over the other five algorithms in 28 out of 42 test cases. Specifically, DC-MFEA exhibits superior performance in terms of DCF1, DCF3~DCF9, and DCF14 under all dynamic test configurations. These compelling experimental results demonstrate the notable advantages of the proposed DC-MFEA over the compared algorithms across the majority of test functions.

From the experimental results, it is obvious that DC-MFEA achieves unsatisfactory results on DCF10~DCF13 problems. In these test instances, the auxiliary task cannot effectively help the search for the main task, since the DUPOFs and DCPOFs of DCF10~DCF13 are relatively far away from each other. This phenomenon shows that the large gap between the unconstrained space and the constrained space will cause auxiliary tasks to hinder the search for the main task and cause negative transfer. Therefore, in future work, it is necessary to design strategies to adjust the knowledge transfer between the two tasks, such as adjusting the rmp parameter.

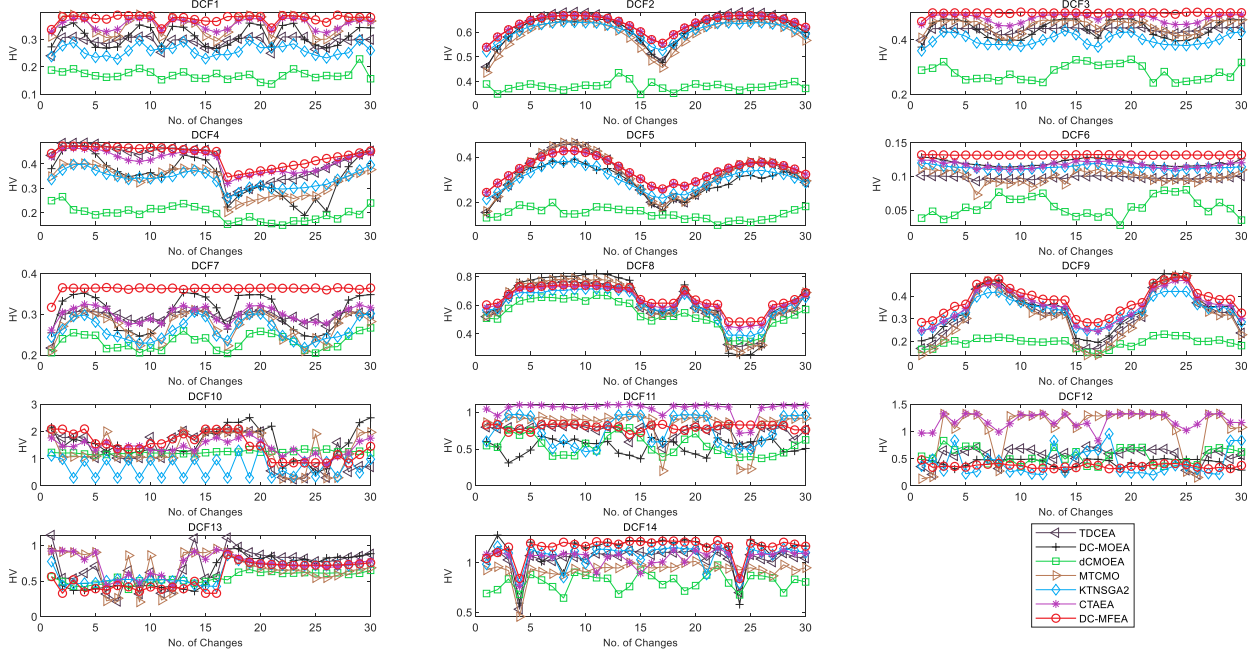


Figure 5.11: Convergence traces of average HV values obtained by compared algorithms with the dynamic test setting ($n_t = 10, \tau_t = 20$)

5.4.3 Sensitivity Studies

One of the objectives of DC-MFEA is to facilitate knowledge transfer between the unconstrained search space and the constrained search spaces. Furthermore, DC-MFEA leverages previously acquired knowledge to promote the search efficiency of the newly generated environment. The former objective is achieved by the SKT component, while the latter objective is achieved by the TKT component. In this part, experimental studies are conducted to verify the efficacy of the SKT component and the TKT component.

To assess the efficacy of the SKT component, we set the parameter rm_p of the SKT component to 0, such that the spatial knowledge cannot be exchanged across the unconstrained search space and the constrained search spaces. We rename the ablated algorithm as $DC-MFEA_{w/o-SKT}$. To validate the effectiveness of the proposed TKT component, we compared the performance

of DC-MFEA and DC-MFEA_{w/o-TKT}, which ablates the TKT component from the evolution process.

Table 5.4 provides the results of the MIGD values and the MHV values of DC-MFEA, DC-MFEA_{w/o-SKT} and DC-MFEA_{w/o-TKT}, respectively. Compared to DC-MFEA_{w/o-TKT}, DC-MFEA with TKT component achieves a significant improvement of convergence in terms of IGD values in most cases. On the other hand, compared to DC-MFEA_{w/o-SKT}, by incorporating the SKT component, DC-MFEA attains significantly higher MHV values in almost all cases. This outcome indicates that the auxiliary task effectively enhances the diversity of the main task. Also, the improved diversity helps the population escape from local optima, thereby improving the quality of convergence. The ablation studies confirm the role of the TKT component and the SKT component and the effectiveness of spatial knowledge and temporal knowledge.

5.4.4 Real-World Applications

To further verify the effectiveness of DC-MFEA in a real-world context, we conducted experiments on a raw ore allocation (ROA) problem [9]. ROA is a multiobjective programming problem encountered in mineral processing. The objective of the ROA problem is to maximize the concentrate yield $C(x)$, maximize concentrate grade $G(x)$, maximize metal recovery $M(x)$, and minimize concentrate cost $P(x)$ simultaneously. Additionally, the ROA problem involves equipment-related constraints, such as the ball mill for magnetic separation. Since the operational conditions are subject to nonstationarity, the constraints of the ROA problem change over time accordingly. In this study, the constraint of the low-intensity magnetic separation $S(x)$ and the constraint of the tailing process cell $\theta(x)$ are considered. Mathematically, the

Chapter 5. Knowledge-Driven Spatial-Temporal Knowledge Transfer for Dynamic Constrained Multiobjective Optimization

Table 5.4: Mean and standard deviation of MIGD and MHV values obtained by ablation algorithms over DCF test functions.

Problem	n_t, τ_t	MIGD			MHV		
		DC-MFEA	DC-MFEA _{w/o-TKT}	DC-MFEA _{w/o-SKT}	DC-MFEA	DC-MFEA _{w/o-TKT}	DC-MFEA _{w/o-SKT}
DCF1	1,20	2.57e-02(4.58e-06)	9.18e-02(2.97e-06)+	1.64e-01(2.07e-04)+	3.97e-01(7.60e-06)	3.08e-01(2.07e-06)+	2.05e-01(4.49e-04)+
	10,20	4.41e-02(8.53e-07)	5.42e-02(1.17e-06)+	1.16e-01(1.21e-04)+	3.75e-01(1.82e-06)	3.60e-01(7.78e-07)+	2.45e-01(7.52e-04)+
	10,10	6.07e-02(3.30e-06)	6.89e-02(7.11e-07)+	1.14e-01(6.18e-05)+	3.54e-01(8.45e-06)	3.43e-01(6.67e-08)+	2.35e-01(5.51e-04)+
DCF2	1,20	4.96e-02(7.05e-09)	4.95e-02(3.76e-09)=	6.42e-02(8.22e-06)+	6.28e-01(2.49e-08)	6.29e-01(1.42e-08)=	5.11e-01(2.89e-03)+
	10,20	4.45e-02(5.38e-09)	4.47e-02(1.91e-09)+	5.98e-02(1.48e-05)+	6.35e-01(1.60e-08)	6.35e-01(4.84e-09)+	5.18e-01(2.87e-03)+
	10,10	4.54e-02(4.18e-09)	4.59e-02(1.27e-08)+	6.50e-02(2.27e-06)+	6.33e-01(1.39e-08)	6.32e-01(3.44e-08)+	5.31e-01(1.51e-03)+
DCF3	1,20	2.55e-02(8.16e-07)	2.40e-02(2.30e-07)=	8.34e-02(8.80e-05)+	4.80e-01(9.63e-07)	4.82e-01(1.19e-06)=	3.88e-01(8.54e-04)+
	10,20	1.38e-02(2.20e-07)	2.21e-02(6.46e-08)+	7.73e-02(2.49e-05)+	4.97e-01(5.24e-07)	4.85e-01(4.81e-07)+	3.53e-01(1.52e-03)+
	10,10	2.08e-02(1.36e-07)	3.43e-02(5.50e-07)+	7.68e-02(6.13e-05)+	4.87e-01(3.31e-07)	4.67e-01(2.24e-06)+	3.58e-01(6.82e-04)+
DCF4	1,20	2.40e-02(2.07e-07)	2.35e-02(1.19e-07)=	8.18e-02(5.52e-05)+	4.20e-01(4.32e-07)	4.22e-01(5.99e-07)=	2.97e-01(1.05e-04)+
	10,20	1.31e-02(1.80e-08)	2.07e-02(2.39e-06)+	8.22e-02(7.02e-05)+	4.34e-01(2.23e-07)	4.23e-01(3.96e-06)+	3.17e-01(5.73e-04)+
	10,10	1.98e-02(2.08e-07)	3.40e-02(6.58e-07)+	7.95e-02(3.77e-05)+	4.24e-01(1.62e-07)	4.04e-01(1.17e-06)+	3.03e-01(6.07e-04)+
DCF5	1,20	1.12e-02(1.21e-08)	1.07e-02(4.19e-08)=	3.46e-02(8.29e-06)+	3.39e-01(2.62e-08)	3.40e-01(3.01e-08)=	2.60e-01(3.57e-04)+
	10,20	9.72e-03(1.93e-08)	1.05e-02(1.45e-08)+	3.63e-02(1.15e-05)+	3.45e-01(5.75e-08)	3.44e-01(9.31e-08)+	2.79e-01(7.41e-04)+
	10,10	1.17e-02(6.50e-08)	1.31e-02(1.24e-07)+	3.61e-02(9.73e-06)+	3.43e-01(2.44e-07)	3.42e-01(2.17e-07)+	2.63e-01(6.36e-04)+
DCF6	1,20	1.06e-02(2.11e-06)	1.31e-02(5.99e-07)=	4.85e-02(2.69e-05)+	1.30e-01(3.29e-07)	1.29e-01(1.25e-07)=	1.03e-01(2.67e-05)+
	10,20	6.67e-03(6.68e-09)	1.39e-02(2.37e-06)+	5.25e-02(2.22e-05)+	1.32e-01(5.78e-09)	1.28e-01(3.49e-07)+	9.50e-02(4.03e-05)+
	10,10	8.19e-03(1.71e-07)	2.35e-02(5.13e-07)+	4.97e-02(2.64e-05)+	1.31e-01(4.15e-08)	1.24e-01(3.74e-08)+	9.79e-02(3.58e-05)+
DCF7	1,20	4.58e-02(1.62e-05)	4.20e-02(1.45e-05)=	1.28e-01(1.19e-04)+	3.40e-01(1.17e-05)	3.43e-01(1.02e-05)=	2.44e-01(8.91e-04)+
	10,20	2.92e-02(6.65e-07)	4.27e-02(1.87e-05)+	1.31e-01(2.43e-04)+	3.62e-01(6.91e-07)	3.43e-01(9.31e-06)+	2.56e-01(2.75e-04)+
	10,10	3.41e-02(2.43e-06)	6.30e-02(7.74e-06)+	1.34e-01(1.36e-04)+	3.54e-01(2.66e-06)	3.24e-01(2.63e-06)+	2.50e-01(3.20e-04)+
DCF8	1,20	7.72e-02(2.19e-06)	7.89e-02(8.61e-07)=	9.84e-02(7.97e-05)+	6.56e-01(2.67e-06)	6.55e-01(6.25e-07)=	5.36e-01(8.43e-04)+
	10,20	8.01e-02(9.36e-07)	7.79e-02(5.57e-06)=	1.14e-01(2.63e-04)+	6.49e-01(1.76e-06)	6.46e-01(4.15e-06)=	5.57e-01(2.61e-03)+
	10,10	7.78e-02(1.13e-06)	7.86e-02(3.36e-05)=	1.16e-01(2.40e-04)+	6.47e-01(3.42e-06)	6.41e-01(1.57e-05)=	5.24e-01(2.75e-03)+
DCF9	1,20	2.50e-02(3.33e-07)	2.46e-02(2.29e-07)=	3.71e-02(1.26e-05)+	3.69e-01(4.41e-07)	3.70e-01(1.04e-06)=	3.00e-01(1.37e-03)+
	10,20	2.44e-02(2.67e-07)	2.46e-02(1.14e-07)=	3.53e-02(7.72e-06)+	3.82e-01(3.79e-07)	3.81e-01(7.60e-07)=	3.16e-01(9.25e-04)+
	10,10	2.43e-02(3.47e-07)	2.54e-02(1.45e-07)+	3.80e-02(2.66e-05)+	3.82e-01(4.93e-07)	3.80e-01(3.11e-07)+	3.21e-01(6.68e-04)+
DCF10	1,20	6.50e-01(2.02e-03)	6.46e-01(1.90e-03)=	7.64e-01(1.65e-03)=	1.49e+00(1.25e-02)	1.48e+00(7.30e-03)=	1.09e+00(8.66e-03)+
	10,20	6.42e-01(1.12e-03)	6.96e-01(5.37e-04)=	7.75e-01(4.12e-03)+	1.48e+00(3.67e-03)	1.36e+00(6.22e-03)=	1.04e+00(6.06e-03)+
	10,10	6.56e-01(3.85e-03)	7.14e-01(1.95e-03)=	8.19e-01(2.20e-03)+	1.50e+00(1.27e-02)	1.26e+00(4.06e-03)=	1.18e+00(2.00e-03)+
DCF11	1,20	3.21e-01(2.51e-04)	3.33e-01(6.97e-05)=	1.66e-01(5.26e-05)=	8.56e-01(5.60e-04)	8.51e-01(5.99e-05)=	5.31e-01(3.62e-03)+
	10,20	2.45e-01(6.65e-04)	2.49e-01(1.93e-04)=	1.65e-01(4.72e-04)=	9.82e-01(1.85e-03)	9.70e-01(1.12e-04)=	5.49e-01(2.19e-03)+
	10,10	3.31e-01(8.18e-04)	3.54e-01(1.45e-04)=	1.80e-01(2.83e-04)=	8.42e-01(1.62e-03)	8.34e-01(8.97e-05)=	5.58e-01(1.36e-03)+
DCF12	1,20	7.71e-01(2.70e-04)	7.58e-01(9.30e-05)=	4.88e-01(3.48e-03)-	2.32e-01(1.22e-03)	2.27e-01(1.61e-04)=	4.19e-01(2.68e-03)-
	10,20	5.90e-01(1.31e-03)	5.96e-01(5.66e-06)=	4.81e-01(1.82e-03)-	5.03e-01(4.75e-03)	5.00e-01(3.66e-06)=	4.49e-01(1.61e-03)=
	10,10	5.99e-01(2.44e-03)	5.84e-01(7.27e-05)=	5.15e-01(5.91e-04)-	4.92e-01(8.89e-03)	4.91e-01(1.31e-04)=	4.19e-01(8.08e-04)=
DCF13	1,20	4.11e-01(2.47e-04)	3.91e-01(3.25e-04)=	4.02e-01(8.90e-04)=	6.07e-01(5.28e-04)	6.10e-01(3.45e-04)=	5.48e-01(1.44e-03)=
	10,20	4.41e-01(8.01e-04)	4.09e-01(3.82e-04)=	3.92e-01(1.16e-03)-	5.67e-01(1.59e-03)	5.78e-01(5.62e-04)=	5.38e-01(1.54e-03)=
	10,10	4.53e-01(1.63e-04)	3.82e-01(6.55e-05)-	3.74e-01(8.09e-04)-	5.46e-01(3.65e-04)	5.95e-01(2.55e-04)-	5.24e-01(2.33e-03)=
DCF14	1,20	1.65e-01(5.64e-05)	2.25e-01(3.43e-04)+	4.50e-01(3.81e-04)+	1.17e+00(1.68e-04)	1.05e+00(1.01e-03)+	4.77e-01(9.95e-04)+
	10,20	1.88e-01(5.18e-05)	1.97e-01(4.84e-04)=	3.57e-01(1.90e-03)+	1.16e+00(1.73e-04)	1.11e+00(1.78e-03)=	6.02e-01(2.86e-03)+
	10,10	1.97e-01(7.94e-05)	2.20e-01(1.23e-04)+	3.36e-01(6.12e-04)+	1.14e+00(3.36e-04)	1.05e+00(4.58e-04)+	5.85e-01(2.14e-03)+

ROA problem can be formulated as follows,

$$\begin{aligned} & < \max C(x), \max M(x), \max G(x) \min P(x) > \\ \text{s.t. } & \begin{cases} S(x) \leq S_{upper} \\ \theta(x) \leq \theta_H, \end{cases} \end{aligned} \quad (5.29)$$

where $x = (x_1, \dots, x_I)$ is the decision variable indicating the quantity of the i -th raw ore, S_{upper} is the upper bound for the low-intensity magnetic separation, and θ_H is the upper bound for the tailing process cell constraint. $C(x)$, $M(x)$, $G(x)$, $P(x)$, $S(x)$, and $\theta(x)$ are formulated as follows respectively,

$$C(x) = \sum_{i=1}^I \frac{(1 - u_i)x_i}{k_{1,i}} + \sum_{i=1}^I \frac{u_i x_i}{k_{2,i}}, \quad (5.30)$$

where $k_{1,i}$ and $k_{2,i}$ represent the beneficiation ratio of the powder ore and the lump ore, respectively. u_i is the ratio of the lump ore obtained from the raw ore.

$$M(x) = \frac{\sum_{i=1}^I (1 - u_i) \beta_{1,i} x_i / k_{1,i} + \sum_{i=1}^I u_i \beta_{2,i} x_i / k_{2,i}}{\sum_{i=1}^I \alpha_i x_i}, \quad (5.31)$$

where α_i is the grade of the raw ore, and β_i is the grade of the concentrate ore.

$$G(x) = \frac{\sum_{i=1}^I (1 - u_i) \beta_{1,i} x_i / k_{1,i} + \sum_{i=1}^I u_i \beta_{2,i} x_i / k_{2,i}}{\sum_{i=1}^I (1 - u_i) x_i / k_{1,i} + \sum_{i=1}^I u_i x_i / k_{2,i}}, \quad (5.32)$$

$$P(x) = \frac{\sum_{i=1}^I r_i x_i + C_{other}}{\sum_{i=1}^I (1 - \mu_i) x_i / k_{1,i} + \sum_{i=1}^I \mu_i x_i / k_{2,i}} + C_{energy}, \quad (5.33)$$

where r_i represents the price of the i -th raw ore, and C_{other} and C_{energy} are the cost of energy and other fixed fee.

The constraint of the low-intensity magnetic separation $S(x)$ is formulated as,

$$S(x) = \frac{\eta_b \sum_{i=1}^I u_i x_i}{\sum_{k=1}^K N_k T_k}, \quad (5.34)$$

Table 5.5: Five conditions for raw ore allocation of mineral processing.

Series Number	Parameters	Condition 1	Condition 2	Condition 3	Condition 4	Condition 5
1	Capacity (N_k)	80.60	58.56	36.52	36.52	36.52
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
2	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
3	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
4	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	25.37	23.66
5	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08
6	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08
7	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08

where N_k and T_k represent the operating number and the running time of the equipment with the series number k , respectively.

$$\begin{aligned}
\theta_p(x) = & \left[\sum_{i=1}^I (1 - u_i) a_{1,i} x_i - \sum_{i=1}^I \frac{(1 - \mu_i) \beta_{1,i}}{k_{1,i}} x_i \right. \\
& + \left(\sum_{i=1}^I \mu_i a_{2,i} x_i - \beta_w \eta_w \sum_{i=1}^I \mu_i x_i \right) \\
& - \sum_{i=1}^I \frac{\beta_{2,i} \mu_i x_i}{k_{2,i}} \Big] \Big/ \left[\sum_{i=1}^I (1 - \mu_i) x_i - \sum_{i=1}^I \frac{(1 - \mu_i)}{k_{1,i}} x_i \right. \\
& \left. + \eta_b \sum_{i=1}^I u_i x_i - \sum_{i=1}^I \frac{\mu_i x_i}{k_{2,i}} \right],
\end{aligned} \tag{5.35}$$

where η_w is the waste ore rate, η_b is the roasting ratio.

In the context of the ROA problem, the parameters N_k and T_k are subject to changes over time.

Table 5.6: Raw ore properties.

i	α_i (%)	$\beta_{1,i}$ (%)	$\beta_{2,i}$ (%)	u_i (%)	$k_{1,i}$ (times)	$k_{2,i}$ (times)
1	33.3	47.5	56.1	62.0	2.3	2.0
2	34.1	47.5	56.1	54.0	2.0	1.92
3	32.5	47.0	54.0	50.0	2.3	2.2
4	33.1	44.5	52.0	65.0	2.8	2.4
5	55.0	60.0	61.0	55.0	1.45	1.4
6	52.0	60.0	0.0	0.0	1.3	2.0

Table 5.7: Mean and standard deviation of HV values obtained by competing algorithms on the ROA problem.

Environment	DC-MFEA	MTCMO	DC-MOEA	KT-NSGA2	dCMOEA	CTAEA	TDCEA
Condition 1	6.19e-01(1.40e-02)	5.11e-01(1.19e-02)	5.90e-01(1.22e-05)=	2.00e-01(9.87e-02)+	7.07e-01(7.65e-02)-	1.60e-01(3.95e-06)+	6.06e-01(2.21e-02)
Condition 2	5.55e-01(2.52e-03)	4.86e-01(3.01e-03)	5.86e-01(1.07e-05)=	3.11e-01(1.69e-01)+	1.10e-01(7.43e-06)+	1.60e-01(4.49e-06)+	5.39e-01(2.92e-03)
Condition 3	6.08e-01(1.88e-03)	5.10e-01(2.15e-03)	5.88e-01(9.73e-06)=	3.27e-02(1.61e-03)+	1.10e-01(7.04e-06)+	1.59e-01(3.93e-06)+	5.73e-01(2.78e-03)
Condition 4	6.11e-01(1.32e-03)	5.14e-01(1.47e-03)	5.86e-01(3.93e-05)=	2.65e-02(5.60e-03)+	1.10e-01(6.80e-06)+	1.59e-01(3.59e-06)+	5.68e-01(2.07e-03)
Condition 5	6.18e-01(1.87e-03)	5.22e-01(2.39e-03)	5.88e-01(1.99e-05)=	1.40e-02(1.19e-03)+	1.10e-01(7.44e-06)+	1.60e-01(1.47e-06)+	5.68e-01(1.96e-03)

The corresponding values for these parameters at each operational condition can be found in Table 5.5. Table 5.6 provides information on the properties of the raw ore, including α_i , $\beta_{1,i}$, $\beta_{2,i}$, u_i , $k_{1,i}$, and $k_{2,i}$. Specifically, the values of η_b , S_{upper} , C_{other} , C_{energy} , and η_w are determined as 82%, 80, 341, 17.66, and 14%, respectively.

In this simulation, we use the HV metric to evaluate the quality of solutions obtained by our algorithm. The statistical results of HV for five continuously changing operational conditions (Condition 1-Condition 5) are reported in Table 5.7. As shown in Table 5.7, the proposed DC-MFEA demonstrates superior performance in three out of the five conditions, as evident from its higher HV values. However, it falls behind dCMOEA and DC-MOEA in Condition 1 and Condition 2, respectively. Nonetheless, it is worth noting that DC-MFEA exhibits relatively stable performance across different conditions when compared to other algorithms. Based on the experimental findings, DC-MFEA displays robust performance across various conditions.

5.5 Conclusion

This work introduced an evolutionary multitask optimization algorithm with a spatial-temporal knowledge transfer strategy for solving DCMOPs, called DC-MFEA. Particularly, to achieve spatial knowledge, two synergistic tasks are established based on the constrained search space and unconstrained search space, respectively. The two tasks are simultaneously optimized within the multifactorial evolutionary algorithm framework, thus enhancing feasibility while maintaining diversity. For temporal knowledge, a similarity-based archive mechanism was designed to reuse historical solutions, accelerating the convergence in newly generated environments. Moreover, to advance the test suite toward real-world cases, a DCMOP benchmark with diverse properties was devised to evaluate the effectiveness of the proposed method. Experimental studies conducted on the proposed test benchmark and a real-world problem have confirmed the efficacy of our proposed algorithm.

In the realm of DCMOPs, numerous avenues exist for future research. A particularly promising direction is the exploration of a broader spectrum of machine learning techniques to transfer knowledge for positive transfer. By harnessing the advancements in machine learning and transfer optimization, we anticipate that the efficacy of transfer optimization can be further improved in solving DCMOPs, particularly in tackling a wider array of real-world problems.

Chapter 6

Knowledge-Driven Solution Selection Strategy for Evolutionary Multitask Optimization

6.1 Motivation

Evolutionary multitasking (EMT), which concurrently conducts the evolutionary search on multiple optimization tasks, has become an emerging paradigm of evolutionary optimization [159, 11, 167, 168, 169]. From path planning [170, 171] to job shop scheduling [172, 173], and multiphysics products [174, 175], EMT has gained promising performance in many domains where multiple optimization tasks routinely occur. Being able to exploit the latent synergies between multiple optimization tasks to accelerate the evolutionary search is what distinguishes the EMT from traditional EAs and arguably what leads to its success in various real-world optimization tasks [176, 88, 177, 178, 179, 180, 181, 167].

A plethora of EMT algorithms have been proposed recently. According to the manner of

knowledge transfer, existing EMT algorithms can be classified into two categories: EMT with implicit transfer and EMT with explicit transfer [182]. For the EMT with the implicit transfer, the exchange of knowledge across tasks occurs through genetic crossover operators. For example, the multifactorial evolutionary algorithm (MFEA) [159] evolves a single population within a unified search space and optimizes these tasks under the framework of multifactorial optimization, where the knowledge across tasks is implicitly transferred through chromosomal crossovers. However, this implicit transfer method may not adequately address the negative transfer where solutions from distinct tasks cannot benefit from one another’s evolutionary search [84].

To reduce the negative transfer, a series of strategies called explicit transfer strategies have been developed [84]. Typically, explicit transfer strategies use additional mechanisms to transfer task-specific knowledge, such as optimized solutions, between different tasks. As domain adaptation techniques have been effective in improving knowledge transferability from the source task to the target task, numerous studies employ these techniques to enhance the transferability of solutions. For instance, EMT-EA [86] employed two separate solvers with distinct operators to solve two tasks, in which a denoising autoencoder is designed to transfer solutions across the two tasks. LDA-MFEA [90] derived a linear mapping between tasks by calculating the ordinal correlation of tasks to solve heterogeneous multitasking optimization problems. To prevent chaotic mapping, AT-MEFA [104] proposed a rank loss function to minimize the chaotic mapping between the landscapes of tasks. Besides, various space alignment methods have been devised to minimize discrepancies between tasks and then transfer solutions via the aligned space [99, 183, 35, 100, 90].

Despite the remarkable progress that has been made, it is worth noting that existing explicit transfer strategies assume transferring the best solutions from the source task could produce the ‘best’ solutions for the target task. These methods thus often simply transfer the best solutions from the source task to boost positive knowledge transfer. However, information from the target

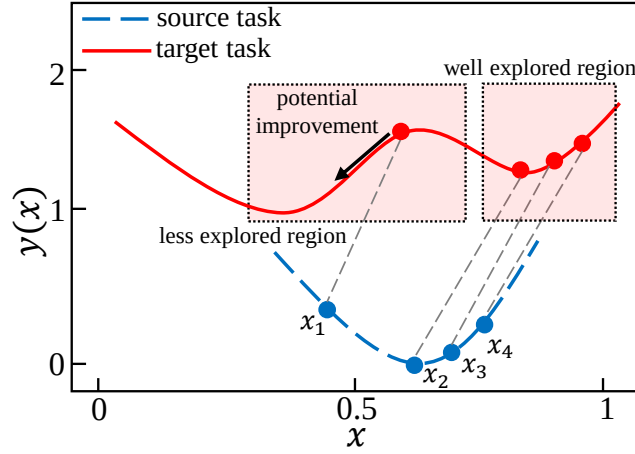


Figure 6.1: EMT for minimizing the single-objective function of two tasks. When an ideal source-target mapping is learned, only transferring the top three best solutions (x_2 , x_3 , and x_4) while falling short of exploiting the solution distribution in the target task could lead to local optima. On the contrary, by leveraging the solution distribution in the target task, transferring x_1 exhibits the potential of escaping from local optima as the region nearby x_1 is less explored than other regions.

task is not fully exploited when selecting solutions for knowledge transfer [86, 83, 91, 99, 53]. As a result, some transferred solutions cannot adapt well to the target task, thereby limiting the potential of positive transfer across tasks. As illustrated in Fig. 6.1, an ideal source-target mapping is learned, where the transferred solutions in the target task have the same ranking as the solutions in the source task. In such an ideal case, if only the best solutions (e.g., top three best solutions x_2 , x_3 , and x_4) are transferred while ignoring x_1 that can help escape from local optima of the target task, the knowledge transfer can only bring limited improvement even negative transfer to the target task. On the contrary, investigating the solution distribution in the target task can gain the prior knowledge that the region near the mapped solution of x_1 is less explored. Based on this, transferring x_1 could lead to the potential to escape from local optima and bring substantial improvement for the target task.

Keeping the above in mind, in this paper, we leverage the information from the target task and

thus propose a lower upper bound-based solution selection approach to measure the potential improvement of a solution on the target task. In particular, a transfer Gaussian process (TGP) model [184] is first employed to analyze the solution distribution information of the target task and the source task. Meanwhile, the inter-task similarity is derived from the TGP model. By combining the inter-task similarity with the solution distribution information, lower confidence bound (LCB) values of the solutions prepared to be transferred are inferred to indicate the potential improvement of the solutions for the target task. Lastly, instead of transferring the best or all solutions in existing explicit transfer algorithms, solutions with lower LCB values are selected for knowledge transfer. Notably, this manner of solution selection can be embedded into most of the existing EMT algorithms. The contributions of this work can be summarized as follows:

1. To alleviate the negative transfer resulting from the lack of exploiting the solution distribution in most existing works, one LCB-based solution selection module is proposed to bridge the research gap by employing the TGP model to analyze the solution distribution of both the source task and target task.
2. To select transferred solutions that can enhance the exploration and exploitation of the target task, an inter-task similarity-based LCB metric is designed to estimate the potential improvement of the transferred solutions on the target task. Compared with conventional LCB, the inter-task similarity-based LCB leverages the inter-task similarity to select solutions, thus transferring solutions adapted to the target task to promote positive transfer.
3. Our proposed solution selection approach can be seamlessly incorporated into most existing EMT algorithms, including single-population-based EMT algorithms and multipopulation-based EMT algorithms. Experimental studies confirm the generality and scalability of the proposed solution selection module.

6.2 Preliminaries

Gaussian process models an unknown function $y(\cdot)$ via the uncertainty, which is given by:

$$y(x) = f(x) + \epsilon, \quad (6.1)$$

where $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ is an independently and uniformly distributed Gaussian noise, and $f(x)$ is a Gaussian random variable of $\mathcal{N}(\mu, \mathbf{K})$. The kernel matrix $\mathbf{K}$ has entries $\mathbf{K}_{mn} = k(x_m, x_n)$ in which $k(\cdot, \cdot)$ is a covariance function between two data points x_n and x_m . In general, any positive semidefinite matrix is valid to be a kernel matrix [185]. To model the function $y(\cdot)$, the log-likelihood function of the Gaussian process is maximized as follows:

$$\begin{aligned} \max_{\theta} \log p(y|\theta) = & -\frac{1}{2} \log |\mathbf{K} + \sigma_n^2 \mathbf{I}| - \frac{N}{2} \log(2\pi) \\ & - \frac{1}{2} (y - \mu)^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} (y - \mu), \end{aligned} \quad (6.2)$$

where N is the number of data points and θ is the parameter of the kernel matrix in the Gaussian process.

After the Gaussian process is trained, $\mathcal{N}(\mu, \mathbf{K})$ can be used as the predictive distribution to represent $y(\cdot)$. Given the input data x^* , the prediction expectation $f(x^*)$ and the prediction uncertainty (prediction standard variance) $\sigma(x^*)$ on the function $y(\cdot)$ can be inferred. The prediction expectation $f(x^*)$ is estimated by:

$$\begin{aligned} f(x^*) &= \hat{\mu} + k^{*T} (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} (y - \hat{\mu}), \\ \hat{\mu} &= \frac{\mathbf{I}^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} y}{\mathbf{I}^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{I}}, \end{aligned} \quad (6.3)$$

where k^* is the covariance vector between the training samples and the input sample x^* . The prediction uncertainty $\sigma(x^*)$ of $f(x^*)$ can be estimated as:

$$\sigma(x^*) = \left(k(x^*, x^*) - k^{*T} (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} k^* \right)^{1/2}. \quad (6.4)$$

By leveraging the prediction expectation and the prediction uncertainty, many well-studied infill criteria are designed to measure the potential performance of a solution to the function

$y(\cdot)$ [186, 187, 188]. For instance, given an objective function, $y(\cdot)$ that needs to be minimized, the LCB describing the potential performance of the solution x^* can be denoted as:

$$LCB(x^*) = f(x^*) - \alpha\sigma(x^*), \quad (6.5)$$

where α is a parameter used to steer toward exploration or exploitation. A larger value of α encourages the algorithm to explore the uncertain region. Conversely, a small value encourages exploitation in the current region [189, 190].

However, the predictive distribution $\mathcal{N}(\mu, \mathbf{K})$ can only handle the independent identically distributed data. To deal with non-independent identically distributed data from distinct tasks, in our paper, a TGP model is employed to exploit knowledge from both the source and target tasks to construct the predictive distribution of the target task. Assume $\mathcal{T}_1$ and $\mathcal{T}_2$ are the target task and the source task, respectively, TGP also follows the assumption that the objective value $y_{\mathcal{T}_1}$ of the target task for any solution $x_{\mathcal{T}_1}$ can be modeled by a Gaussian random variable and an additive noise:

$$y_{\mathcal{T}_1}(x_{\mathcal{T}_1}) = f_{\mathcal{T}_1}(x_{\mathcal{T}_1}) + \epsilon_{\mathcal{T}_1}, \quad (6.6)$$

where $f_{\mathcal{T}_1}(x_{\mathcal{T}_1})$ is a Gaussian random variable of $\mathcal{N}(\mu_{\mathcal{T}_1}, \mathbf{K})$, $\mu_{\mathcal{T}_1}, \mathbf{K}$ and $\epsilon_{\mathcal{T}_1}$ are the mean value, covariance kernel matrix $\mathbf{K}$, and the additive noise term $\epsilon_{\mathcal{T}_1} \sim \mathcal{N}(0, \sigma_{\mathcal{T}_1}^2)$, respectively. The objective function $f_{\mathcal{T}_1}$ of the target task can be modeled by:

$$p(f_{\mathcal{T}_1}|D_{\mathcal{T}_1}, D_{\mathcal{T}_2}) = \mathcal{N}(\mu_{\mathcal{T}_1}, \mathbf{K}), \quad (6.7)$$

where $D_{\mathcal{T}_1} = \{(x_{\mathcal{T}_1}, y_{\mathcal{T}_1}(x_{\mathcal{T}_1}))\}$ is the solution set of the target task, $D_{\mathcal{T}_2} = \{(x_{\mathcal{T}_2}, y_{\mathcal{T}_2}(x_{\mathcal{T}_2}))\}$ is the solution set of the source task. In this manner, by making use of information from non-independent identically distributed data $D_{\mathcal{T}_1}$ and $D_{\mathcal{T}_2}$, TGP can better model the solution distribution of the target task $p(f_{\mathcal{T}_1}|D_{\mathcal{T}_1}, D_{\mathcal{T}_2})$, thus well describing the potential improvement of a solution on the target task.

6.3 Proposed Framework

In this work, we adopt a variant of the Gaussian process TGP to derive the LCB value, thus indicating the potential improvement of each transferred solution. Before presenting our method, the Gaussian process and LCB are briefly introduced.

6.3.1 Overview

Afterward, we first give the training process of TGP, which plays a core component in analyzing the solution distribution information of the target task and the source task. Then, based on the trained TGP, the process of the proposed solution selection approach is detailed.

The workflow of the proposed approach is illustrated in Fig. 6.2. For the training process, with the solution set $D_{\mathcal{T}_1}$ of the target task $\mathcal{T}_1$ and solution set $D_{\mathcal{T}_2}$ of the source task $\mathcal{T}_2$, a TGP model is trained and the inter-task similarity λ can be inferred simultaneously. After that, the solution selection module adopts the trained TGP model and inter-task similarity λ to synthesize the LCB value for each solution prepared to be transferred, where solutions with higher potential improvement have lower LCB values. Finally, several solutions with lower LCB values are transferred to the target task. In what follows, the details of the training process and the solution selection module are presented.

6.3.2 Training Process of Transfer Gaussian Process

As illustrated in Eq. (6.7), TGP models the objective function $f_{\mathcal{T}_1}$ of the target task $\mathcal{T}_1$ by $p(f_{\mathcal{T}_1}|D_{\mathcal{T}_1}, D_{\mathcal{T}_2}) = \mathcal{N}(\mu_{\mathcal{T}_1}, \mathbf{K})$, where the covariance kernel matrix models the correlation between samples from different tasks. Each element $\mathbf{K}_{mn}$ in kernel matrix $\mathbf{K}$ is

$$\mathbf{K}_{nm} = \begin{cases} \lambda k(x_n, x_m), & x_n, x_m \text{ come from different tasks} \\ k(x_n, x_m), & \text{otherwise,} \end{cases} \quad (6.8)$$

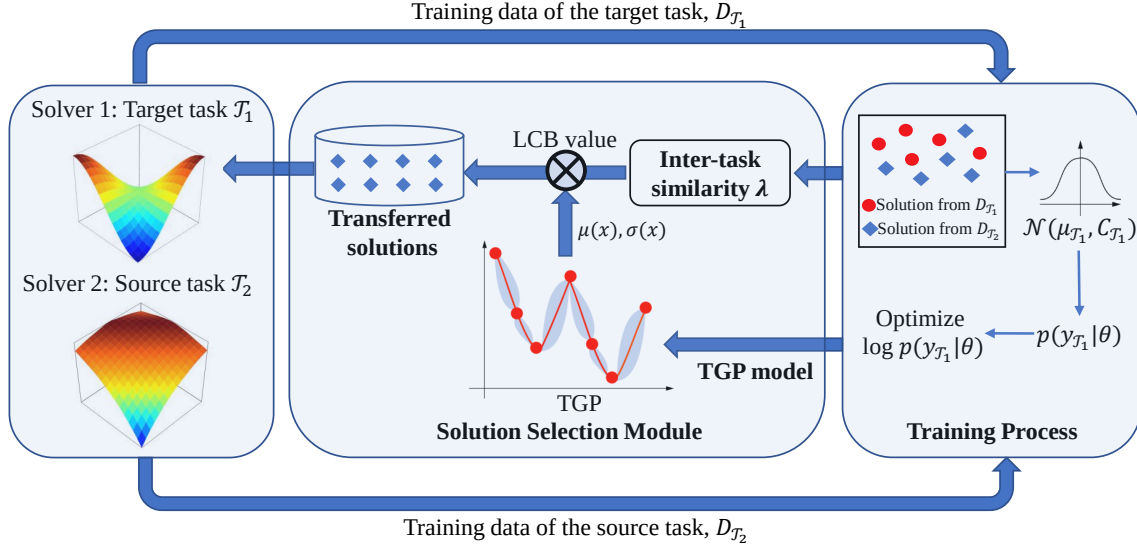


Figure 6.2: The framework of the proposed solution selection approach. With the training data from the two tasks, a TGP model respective to the target task is trained, where the inter-task similarity is derived simultaneously. Solution Selection Module: The LCB values of the solutions are synthesized using the TGP model and inter-task similarity. According to the LCB values, several solutions with lower LCB values are selected and transferred to the target task.

where

$$k(x_n, x_m) = \theta_l \exp \left(\sum_{d=1}^D -\frac{(x_m^{(d)} - x_n^{(d)})^2}{2\theta_d^2} \right), \quad (6.9)$$

where x_n, x_m come from the solution set $D_{\mathcal{T}_1} \cup D_{\mathcal{T}_2}$, D is the dimension of the decision variable, θ_l and θ_d are the hyper-parameters of the kernel matrix. In Eq. (6.8), the additional factor λ reveals the correlation between samples of the different tasks that are less or equal to the correlation between the ones in the same task, and the additional factor λ can be defined as,

$$\lambda = 2 \left(\frac{1}{1 + \exp(b)} \right) - 1. \quad (6.10)$$

The factor λ has range $(-1, 1)$, which ensures that the kernel matrix is a valid kernel function according to Theorem 1 [184].

Theorem 1. Let $\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix}$ be a positive semidefinite matrix with $\mathbf{K}_{12} = \mathbf{K}_{21}^T$. Then for $|\lambda| \leq 1$, $\mathbf{K}^* = \begin{pmatrix} \mathbf{K}_{11} & \lambda \mathbf{K}_{12} \\ \lambda \mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix}$ is also a positive semidefinite matrix.

Considering the conditional distribution with additive noise term, i.e., $\epsilon_{\mathcal{T}}$ in Eq. (6.6), the conditional distribution Eq. (6.7) can be rewritten as follows,

$$p(y_{\mathcal{T}_1} | y_{\mathcal{T}_2}, D_{\mathcal{T}_1}, D_{\mathcal{T}_2}) \sim \mathcal{N}(\mu_{\mathcal{T}_1}, \mathbf{C}_{\mathcal{T}_1}), \quad (6.11)$$

where

$$\mu_{\mathcal{T}_1} = \mathbf{K}_{21}(\mathbf{K}_{11} + \sigma_{\mathcal{T}_2}^2 \mathbf{I})^{-1} y_{\mathcal{T}_2}, \quad (6.12)$$

and

$$\mathbf{C}_{\mathcal{T}_1} = (\mathbf{K}_{22} + \sigma_{\mathcal{T}_1}^2 \mathbf{I}) - \mathbf{K}_{21}(\mathbf{K}_{11} + \sigma_{\mathcal{T}_2}^2 \mathbf{I})^{-1} \mathbf{K}_{12}, \quad (6.13)$$

$\mathbf{K}_{11}$ and $\mathbf{K}_{22}$ are the kernel matrices of the solutions in the source task and target task, respectively. $\mathbf{K}_{12} = \mathbf{K}_{21}^T$ is the kernel matrix across tasks.

Once the TGP model is defined, the related hyper-parameter $\theta = \{\sigma_{\mathcal{T}_1}^2, \sigma_{\mathcal{T}_2}^2, b, \theta_l, \theta_1, \dots, \theta_D\}$ can be optimized. For hyper-parameter learning, we optimize the log-likelihood equation,

$$\begin{aligned} \log p(y_{\mathcal{T}_1} | \theta) = & -\frac{1}{2} \log |\mathbf{C}_{\mathcal{T}_1}| - \frac{N}{2} \log(2\pi) \\ & - \frac{1}{2} (y_{\mathcal{T}_1} - \mu_{\mathcal{T}_1})^T \mathbf{C}_{\mathcal{T}_1}^{-1} (y_{\mathcal{T}_1} - \mu_{\mathcal{T}_1}). \end{aligned} \quad (6.14)$$

After optimizing the hyper-parameter θ , the TGP model is saved for the inference of the prediction expectation and prediction uncertainty of a solution on the target task. In this process, by optimizing the hyper-parameter b , λ can be derived to denote the inter-task similarity due to the following properties: According to Eq. (6.8), $\lambda \rightarrow 1$ denotes solutions from different tasks have high covariance and the two tasks are highly correlated, while $\lambda \rightarrow 0$ means the tasks are not similar or have no confidence in their relation. In our work, the negative correlation (i.e.,

$\lambda < 0$) is eliminated, and the inter-task similarity is denoted as $\lambda = \max\{\lambda, 0\}$ ¹. The details of the Training Process are shown in Algorithm 11.

Algorithm 11: Training Process

Input: $D_{\mathcal{T}_1}, D_{\mathcal{T}_2}$ (the dataset of the target task, and the dataset of the source task)

Output: TGP (the TGP model for the target task), λ (the inter-task similarity)

- 1 Initialize hyper-parameters $\theta = \{\sigma_{\mathcal{T}_1}^2, \sigma_{\mathcal{T}_2}^2, b, \theta_l, \theta_1, \dots, \theta_D\}$;
- 2 Define the TGP model as $p(y_{\mathcal{T}_1} | y_{\mathcal{T}_2}, D_{\mathcal{T}_1}, D_{\mathcal{T}_2}) \sim \mathcal{N}(\mu_{\mathcal{T}_1}, \mathbf{C}_{\mathcal{T}_1})$;
- 3 Compute the log-likelihood as

$$\begin{aligned} \log p(y_{\mathcal{T}_1} | \theta) = & -\frac{1}{2} \log |\mathbf{C}_{\mathcal{T}_1}| - \frac{N}{2} \log(2\pi) \\ & - \frac{1}{2} (y_{\mathcal{T}_1} - \mu_{\mathcal{T}_1})^T \mathbf{C}_{\mathcal{T}_1}^{-1} (y_{\mathcal{T}_1} - \mu_{\mathcal{T}_1}) \end{aligned}$$

- 4 Optimize the TGP with θ by maximizing the log-likelihood;
 - 5 $\lambda = 2(\frac{1}{1+e^b}) - 1$;
 - 6 $\lambda = \max\{\lambda, 0\}$;
 - 7 **return** TGP, λ
-

6.3.3 Solution Selection Module

The proposed solution selection module selects solutions for knowledge transfer by estimating the potential improvement of a solution x' prepared to be transferred to the target task $\mathcal{T}_1$. The solution x' prepared to be transferred can be obtained by:

$$x' = \mathcal{M}(x), x \in \mathcal{T}_2, \quad (6.15)$$

where $\mathcal{M}(\cdot)$ could be any type of mapping, such as identify mapping, linear mapping, e.g., denoising autoencoder in [86], affine transformation in [104], or non-linear mapping, e.g., kernelized autoencoder in [91].

¹Although “correlation” and “similarity” are not equivalent, following the previous study using Spearman correlation to represent the inter-task similarity [191], we use the proposed correlation to represent the inter-task similarity.

Algorithm 12: Solution Selection Module

Input: λ_{gen} (the inter-task similarity at generation gen), TGP (the TGP model), S (the solutions prepared to be transferred)

Output: X (selected solutions)

```

1  $X = \emptyset;$ 
2  $p(S|gen) = \lambda_{gen};$ 
3 for each solution  $x' \in S$  do
4    $f(x'), \sigma(x') = TGP(x');$ 
5    $SLCB(x') = f(x') - \lambda_{gen}\sigma(x');$ 
6 end
7  $p(\text{transfer } x' | S) = \frac{-SLCB(x')}{\sum_{x \in S} \exp(-SLCB(x))};$ 
8  $p(\text{transfer } x' | gen) = \frac{\lambda_{gen} \exp(-SLCB(x'))}{\sum_{x \in S} \exp(-SLCB(x))};$ 
9  $p(\text{transfer } x' | gen) = \frac{\exp(p(\text{transfer } x' | gen))}{\sum_{x \in S} \exp(p(\text{transfer } x | gen))};$ 
10 while  $|X| < N_t$  do
11   Sample  $x' \sim p(\text{transfer } x' | gen);$ 
12    $X = X \cup \{x'\};$ 
13 end
14 return  $X$ 

```

Next, an inter-task similarity-based LCB, called SLCB, is designed to measure the potential improvement by using inter-task similarity and the trained TGP model.

$$SLCB(x') = f(x') - \lambda_{gen}\sigma(x'), \quad (6.16)$$

where λ_{gen} is the similarity factor at generation gen , $f(x')$ and $\sigma(x')$ are the prediction expectation and prediction uncertainty of the objective value on the target task, which are inferred by the TGP model. Compared with conventional LCB, the designed SLCB combines inter-task similarity to measure the potential performance of candidate transfer solutions on the target task. In this manner, solutions adapted to the target task solution distribution will have lower SLCB values for selection, thus promoting positive transfer. In general, there are two cases for Eq. (6.16):

1. A lower λ_{gen} (i.e., $\lambda \rightarrow 0$): The SLCB value for the solution x' is mainly dependent on the objective value $f(x')$. In this case, a lower objective value $f(x')$ encourages x' to be

selected for better exploitation.

2. A higher λ_{gen} (i.e., $\lambda \rightarrow 1$): The SLCB value for the solution x' is dependent on both the predicted objective value $f(x')$ and the predicted uncertainty $\sigma(x')$. Particularly, a higher $\sigma(x')$ (i.e., $\sigma(x') \rightarrow +\infty$) encourages x' to be selected for exploring the target task to escape from the local optima.

Incorporating the SLCB value, a conditional probability $p(\text{transfer } x' | gen)$ is proposed to indicating the probability of a solution $x' \in S$ being selected for knowledge transfer at generation gen , where S is the solution set $S = \{x' | x' = \mathcal{M}(x), x \in \mathcal{T}_2\}$. Firstly, we denote the probability of the set S being selected at generation gen as,

$$p(S | gen) = \lambda_{gen}. \quad (6.17)$$

Then, another conditional probability is synthesized to indicate the probability of each solution $x' \in S$ to be selected,

$$p(\text{transfer } x' | S) = \frac{\exp(-SLCB(x'))}{\sum_{x \in S} \exp(-SLCB(x))}. \quad (6.18)$$

Combined with Eq. (6.17) and Eq. (6.18), the probability for selecting and transferring solution x' at generation gen is given by,

$$p(\text{transfer } x' | gen) = \frac{\lambda_{gen} \exp(-SLCB(x'))}{\sum_{x \in S} \exp(-SLCB(x))}. \quad (6.19)$$

The probability $p(\text{transfer } x' | gen)$ needs to be normalized to make sure $\sum_{x \in S} p(\text{transfer } x | gen) = 1$,

$$p(\text{transfer } x' | gen) = \frac{\exp(p(\text{transfer } x' | gen))}{\sum_{x \in S} \exp(p(\text{transfer } x | gen))}. \quad (6.20)$$

Finally, we sample at most N_t solutions according to the probability $p(\text{transfer } x' | gen)$ and transfer them to the target task. The details of the solution selection module are shown in Algorithm 12.

6.3.4 Incorporation with Existing Multitask Optimization Algorithms

The proposed solution selection module can be seamlessly incorporated with multipopulation-based EMT algorithms (i.e., multiple solvers for multiple tasks) but also can be embedded into single-population-based EMT algorithms (e.g., GMFEA [85], AT-MFEA [104], MFEA-KAE [91], etc). The details of the proposed solution selection module incorporating multipopulation-based EMT algorithms and single-population-based EMT algorithms are presented in Algorithm 13 and Algorithm 14, respectively. We take the single-population-based EMT algorithms for illustration here. The LCB-based solution selection is triggered during the inter-task crossover stage. Specifically, this procedure (Line 17-Line 27) can be divided into the following four steps.

Algorithm 13: Multipopulation-based EMT with the Proposed Solution Selection Module

Input: $\mathcal{T}_k, k = 1, \dots, K$ (K optimization tasks), N (number of solutions for each task), $\mathcal{M}(\cdot)$ (any type of mapping for knowledge transfer)

Output: $P_{\mathcal{T}_k}$ (Populations for the k -th task)

```

1 Initialize the populations  $P_{\mathcal{T}_k}$  for  $\mathcal{T}_k$ ;
2 while stopping conditions are not satisfied do
3   Solve the  $\mathcal{T}_k$  with  $P_{\mathcal{T}_k}$  by a single-task optimization solver;
4   if the knowledge transfer interval  $TG$  is met then
5      $D_{\mathcal{T}_k} = \{(x, y_{\mathcal{T}_k}(x))\}, x \in P_{\mathcal{T}_k}$ ;
6     //Select  $\mathcal{T}_k, \mathcal{T}_{mod(k+1, K)}$  as target task and source task, respectively;
7     //The subscript  $mod(k+1, K), k$  in the following symbols  $TGP_{mod(k+1, K), k}, \lambda_{mod(k+1, K), k}, S_{mod(k+1, K), k}$ , and
8        $X_{mod(k+1, K), k}$  denotes the knowledge transfer is conducted from  $\mathcal{T}_{mod(k+1, K)}$  to  $\mathcal{T}_k$ ;
9     Infer  $TGP_{mod(k+1, K), k}$  and  $\lambda_{mod(k+1, K), k} \leftarrow$  Algorithm 1 with  $D_{\mathcal{T}_k}$  and  $D_{\mathcal{T}_{mod(k+1, K)}}$ ;
10    Get the solution sets  $S_{mod(k+1, K), k} = \{x' | x' = \mathcal{M}(x), x \in \mathcal{T}_{mod(k+1, K)}\}$ ;
11    Obtain  $X_{mod(k+1, K), k} \leftarrow$  Algorithm 2 with  $\lambda_{mod(k+1, K), k}, TGP_{mod(k+1, K), k}$ , and  $S_{mod(k+1, K), k}$ ;
12    Environmental Selection for  $P_{\mathcal{T}_k} \cup X_{mod(k+1, K), k}$ ;
13  end
14 return  $P_{\mathcal{T}_k}$ 

```

1. Learning of TGP and Inter-task Similarity: Denoting the solutions with skill factor τ_i and τ_j as $D_{\mathcal{T}_i}$ and $D_{\mathcal{T}_j}$, respectively, we take turns treating τ_i and τ_j as the source and target tasks. Two TGP models TGP_{ij} and TGP_{ji} and the inter-task similarity λ_{ij} and λ_{ji} can be inferred by Algorithm 11².

² TGP_{ij} and λ_{ij} respectively denote the TGP model and the inter-task similarity when regarding $D_{\mathcal{T}_i}$ as the source task and $D_{\mathcal{T}_j}$ as the target task.

Algorithm 14: MFEA-based EMT with the Proposed Solution Selection Module

Input: K (number of tasks), N (number of solutions for each task), $\mathcal{M}(\cdot)$ (any type of mapping for knowledge transfer)

Output: P (the outputted solutions)

```

1  Randomly sample  $N \cdot K$  solutions to form initial population  $P(0)$ ;
2  for every solution  $p_i$  in  $P(0)$  do
3      Assign skill factor  $\tau_i = \text{mod}(i, K) + 1$ , for the case of  $K$  tasks;
4      Evaluate  $p_i$  for task  $\tau_i$  only;
5  end
6  Set  $t = 1$ ;
7  while stopping conditions are not satisfied do
8       $P(t) \leftarrow$  Select top  $N \cdot K/2$  of generated solutions based on scalar fitness;
9      Configure offspring population  $P_c(t) = \emptyset$ ;
10     while offspring generated for each task  $< N$  do
11         Sample two solutions at random:  $x_i$  and  $x_j$  from  $P(t)$ ;
12         if  $\tau_i = \tau_j$  then
13             //Intra-task crossover;
14              $[x_a, x_b] \leftarrow$  intra-task crossover between  $x_i$  and  $x_j$ ;
15             Assign offspring  $x_a$  and  $x_b$  with skill factor  $\tau_i$ ;
16         else if  $\text{rand} \leq \text{rmp}$  then
17             //Inter-task crossover;
18             Collect solutions with  $\tau_i$  and solutions with  $\tau_j$  as the datasets  $D_{\mathcal{T}_i}$  and  $D_{\mathcal{T}_j}$ , respectively;
19             Infer  $TGP_{ij}$  and  $\lambda_{ij} \leftarrow$  Algorithm 1 with  $D_{\mathcal{T}_i}$  and  $D_{\mathcal{T}_j}$ ;
20             Infer  $TGP_{ji}$  and  $\lambda_{ji} \leftarrow$  Algorithm 1 with  $D_{\mathcal{T}_j}$  and  $D_{\mathcal{T}_i}$ ;
21             Get the solution sets  $S_{ij} = \{x' | x' = \mathcal{M}(x), x \in \mathcal{T}_i\}$  and  $S_{ji} = \{x' | x' = \mathcal{M}(x), x \in \mathcal{T}_j\}$ ;
22             Obtain  $X_{ij} \leftarrow$  Algorithm 2 with  $\lambda_{ij}$ ,  $TGP_{ij}$ , and  $S_{ij}$ ;
23             Obtain  $X_{ji} \leftarrow$  Algorithm 2 with  $\lambda_{ji}$ ,  $TGP_{ji}$ , and  $S_{ji}$ ;
24             Select solutions  $x'_i \in X_{ij}$  and  $x'_j \in X_{ji}$  with the highest probability values according to Eq. (20);
25              $x_a \leftarrow$  crossover between  $x'_i$  and  $x_j$ ;
26              $x_b \leftarrow$  crossover between  $x_i$  and  $x'_j$ ;
27             Each offspring is randomly assigned skill factor  $\tau_i$  or  $\tau_j$ ;
28         else
29              $[x_a] \leftarrow$  local variation (mutation) of  $x_i$ ;
30             Assign offspring  $x_a$  skill factor  $\tau_i$ ;
31              $[x_b] \leftarrow$  local variation (mutation) of  $x_j$ ;
32             Assign offspring  $x_b$  skill factor  $\tau_j$ ;
33         end
34         Evaluate  $[x_a, x_b]$  for their assigned skill factors only;
35          $P_c(t) = P_c(t) \cup [x_a, x_b]$ ;
36     end
37      $t = t + 1$ ;
38 end

```

2. **Solution Mapping:** By mapping solutions via Eq. (6.15), the solutions from $D_{\mathcal{T}_i}$ and $D_{\mathcal{T}_j}$ are respectively mapped, and thus we get the mapped solution sets S_{ij} and S_{ji} .
3. **Solution Selection:** The transferred solutions X_{ij} can be outputted by Algorithm 12 with λ_{ij} , TGP_{ij} , and S_{ij} . X_{ji} can be similarly obtained. According to Eq. (6.20), solutions x'_i and x'_j with the highest probability values are respectively selected from X_{ij} and X_{ji} for inter-task crossover.
4. **Inter-task Crossover:** One specific crossover operator is performed on the selected parents x'_i and x'_j to generate the offspring.

6.4 Experiments

In this section, comprehensive numerical experiments have been conducted on two single-objective multitasking benchmarks, one multiobjective multitasking benchmark, and a multiobjective multitasking real-world application. Specifically, we embed the solution selection approach into a multipopulation-based EMT algorithm and several single-population-based EMT algorithms to validate the effectiveness and generality of the proposed solution selection approach.

6.4.1 Experimental Settings

Baseline Algorithms and Test Problems. In this study, we first incorporate the solution selection module into a vanilla multipopulation-based EMT algorithm (i.e., multiple solvers handle multiple tasks separately), namely LCB-EMT. The following EMT algorithms are selected for comparison: MFEA [159], EMT-EA [86], GMFEA [85], MFEA-KAE [91], AT-MFEA [104], and EMT-GFK [192].

Besides, we verify the effectiveness of the proposed solution selection approach on several existing single-population-based EMT algorithms. Specifically, we embedded the solution selection approach into GMFEA, AT-MFEA, and MFEA-KAE, the corresponding algorithms are renamed as the prefix “LCB-” with their original names.

The widely used classical single-objective multitasking benchmark [191] and F series single-objective multitasking benchmark [85] are employed to investigate the performance of the proposed solution selection approach. In addition to the two single-objective multitasking benchmarks, a multiobjective multitasking benchmark [191] is employed in our experiments.

For LCB-EMT, it is worth noting that different tasks may have different bounds of decision variables [85]. Therefore, we need to align the solutions from the source task to the target task. We do the following scale for alignment,

$$x' = LB_{\mathcal{T}_1} + (UB_{\mathcal{T}_1} - LB_{\mathcal{T}_1}) \frac{x - LB_{\mathcal{T}_2}}{UB_{\mathcal{T}_2} - LB_{\mathcal{T}_2}}, x \in P_{\mathcal{T}_2}, \quad (6.21)$$

where $LB_{\mathcal{T}}$ and $UB_{\mathcal{T}}$ denote the lower bound and upper bound of decision variables of the task $\mathcal{T}$, and x' means the transferred solution of the target task $\mathcal{T}_1$. Besides, if the two tasks have different dimensions of decision variables, we directly truncate the overlong genes or randomly generate the missing genes.

6.4.2 Comparisons with Baselines

1) Single-objective multitasking benchmark: The statistical results of the average objective values and standard variance values over 10 independent runs are summarized in Tables 6.1 and 6.2. Superior performance is highlighted in bold. In the tables, (+), (=), and (-) indicate that LCB-EMT is statistically significantly better, similar, or significantly worse than the compared algorithms, respectively, by using the Wilcoxon test. The findings are listed as follows.

Firstly, the superiority of LCB-EMT over the other algorithms is obvious from the results in Tables 6.1 and 6.2. Specifically, for the classical single-objective multitasking benchmark, LCB-

Table 6.1: Mean objective values and standard deviation obtained by compared algorithms on the classical single-objective multitask problems.

Problem	Task	MFEA	EMT-EA	GMFEA	AT-MFEA	MFEA-KAE	EMT-GFK	LCB-EMT
CI+HS	T1	1.67e-01±5.89e-02(+)	9.83e-04±7.12e-04(+)	1.61e-02±8.34e-03(+)	2.95e-03±4.94e-03(+)	2.56e-03±2.31e-03(+)	9.43e-05±1.21e-04(+)	1.31e-07±2.58e-07
	T2	4.73e+01±8.39e+00(+)	1.02e+00±4.91e-01(+)	3.23e+01±1.49e+01(+)	7.04e-01±8.80e-01(+)	1.61e+00±8.42e-01(+)	5.70e-03±2.41e-04(+)	1.53e-03±4.28e-03
CI+MS	T1	1.92e+01±1.24e-01(+)	2.07e+00±4.00e-01(+)	1.93e+01±5.33e-01(+)	1.81e+00±2.69e-01(+)	2.05e+00±4.09e-01(+)	1.30e-03±1.47e-04(+)	8.07e-04±9.11e-04
	T2	6.08e+01±1.22e+01(+)	1.52e+00±8.56e-01(+)	3.01e+01±2.85e+00(+)	1.49e+00±8.29e-01(+)	1.25e+00±6.66e-01(+)	1.17e+00±3.46e-01(+)	5.48e-01±1.63e+00
CI+LS	T1	2.01e+01±5.80e-02(+)	1.81e+00±5.14e-01(+)	2.01e+01±2.12e-02(+)	1.08e+00±5.81e-01(+)	1.74e+00±6.06e-01(+)	9.85e-03±5.16e-03(+)	5.13e-03±4.60e-03
	T2	6.42e+03±3.43e+02(+)	2.25e+03±5.51e+02(-)	6.63e+03±9.99e+01(+)	2.27e+03±4.67e+02(-)	2.62e+03±6.26e+02(-)	1.05e+03±7.05e+02(-)	5.27e+03±2.98e+03
PI+HS	T1	5.17e+01±8.47e+00(+)	5.54e-01±2.00e-01(-)	5.56e+00±5.31e+00(-)	1.41e-01±1.07e-01(-)	1.71e+00±7.74e-01(-)	7.10e-04±2.53e-03(-)	2.92e+01±2.90e+01
	T2	7.03e+00±2.48e+00(+)	9.58e-03±2.81e-03(-)	1.13e-01±1.09e-01(+)	2.88e-03±2.16e-03(-)	5.09e-02±1.40e-02(=)	3.22e-01±1.33e-02(+)	2.74e-02±2.51e-02
PI+MS	T1	1.92e+01±2.72e-01(+)	1.88e+00±4.60e-01(+)	1.91e+01±2.50e-01(+)	1.50e+00±4.69e-01(+)	2.08e+00±3.97e-01(+)	6.61e-02±9.80e-03(+)	4.47e-03±8.84e-04
	T2	8.76e+02±2.96e+02(+)	1.01e+02±5.40e+01(+)	1.38e+03±1.91e+03(+)	2.52e+02±2.04e+02(+)	1.68e+02±3.85e+01(+)	6.18e+01±4.52e+01(+)	4.79e+01±5.33e+01
PI+LS	T1	1.94e+01±2.45e-01(+)	2.07e+00±2.92e-01(+)	2.01e+01±5.59e-02(+)	1.41e+00±5.31e-01(+)	1.94e+00±7.53e-01(+)	3.00e-03±2.55e-03(+)	3.05e-05±6.29e-05
	T2	3.70e+00±7.61e-01(+)	4.45e-01±7.41e-02(-)	6.51e+00±3.03e+00(+)	3.04e-01±7.04e-02(-)	4.71e-01±2.46e-02(-)	1.09e+00±1.40e-01(=)	9.31e-01±1.84e+00
NI+HS	T1	1.00e+03±1.54e+02(+)	5.43e+01±1.68e+01(-)	2.25e+02±8.81e+01(=)	9.06e+01±5.82e+01(-)	1.67e+02±6.25e+01(=)	4.54e+01±1.90e+00(-)	3.47e+02±2.87e+02
	T2	5.45e+01±1.21e+01(+)	1.25e+00±1.00e+00(+)	4.14e+01±1.11e+01(+)	1.15e+00±7.22e-01(+)	1.91e+00±8.46e-01(+)	7.03e-04±1.11e-04(+)	5.55e-08±2.90e-08
NI+MS	T1	1.30e+01±5.28e-02(+)	4.50e-04±2.09e-04(+)	1.71e-02±1.75e-02(+)	6.88e-05±6.28e-05(+)	6.58e-03±6.85e-03(+)	3.66e+01±3.84e+00(+)	1.96e-12±5.09e-12
	T2	3.29e+00±3.50e-01(+)	3.64e-01±7.58e-02(+)	1.49e+00±7.17e-01(+)	1.68e-01±7.06e-02(+)	4.91e-01±7.02e-02(+)	2.51e-02±4.32e-03(+)	2.98e-03±3.50e-03
NI+LS	T1	6.67e+03±4.14e+02(+)	1.50e+00±9.39e-01(-)	3.36e+01±4.99e+00(+)	1.75e+00±9.71e-01(-)	1.18e+00±4.20e-01(-)	3.18e+00±1.66e-01(-)	1.50e+01±9.38e+00
	T2	6.55e+03±3.19e+02(=)	1.94e+03±4.74e+02(-)	8.26e+03±3.42e+02(+)	2.18e+03±4.64e+02(-)	3.11e+03±1.50e+03(=)	1.91e+04±8.49e+03(+)	5.26e+03±2.98e+03
+/-/-		17/1/0	12/0/6	16/1/1	11/0/7	11/3/4	13/1/4	

Table 6.2: Mean objective values and standard deviation obtained by compared algorithms on the F series single-objective multitask problems.

Problem	Task	MFEA	EMT-EA	GMFEA	AT-MFEA	MFEA-KAE	EMT-GFK	LCB-EMT
F1	T1	4.11e+01±1.13e-01(+)	3.08e-02±7.93e-03(+)	1.13e-01±1.18e-01(+)	1.72e-02±1.01e-02(+)	2.66e-02±8.60e-03(+)	6.06e-02±7.79e-03(+)	1.65e-03±4.92e-03
	T2	2.09e+02±8.56e+01(+)	1.39e+02±2.58e+01(+)	1.08e+02±1.10e+02(+)	2.46e+01±1.67e+01(+)	1.44e+02±3.07e+01(+)	4.24e+01±5.21e+00(+)	4.09e-01±8.00e-01
F2	T1	7.48e+00±1.25e+00(+)	2.09e+00±3.09e-01(+)	1.31e+01±1.18e+00(+)	9.28e-01±5.22e-01(+)	2.05e+00±2.27e-01(+)	4.96e-01±3.56e-02(+)	3.80e-03±6.53e-03
	T2	4.41e+02±1.06e+02(+)	1.31e+02±2.76e+01(+)	1.54e+03±4.64e+02(+)	3.10e+01±2.12e+01(+)	1.36e+02±3.56e+01(+)	2.85e+01±4.91e+00(+)	2.32e-03±3.73e-03
F3	T1	1.12e+03±6.22e+02(+)	8.75e+01±3.59e+01(-)	2.44e+02±7.68e+01(=)	1.67e+02±1.16e+02(=)	1.54e+02±4.56e+01(=)	1.86e+02±7.71e+00(=)	1.65e+02±1.42e+02
	T2	5.77e+01±3.71e+00(+)	1.56e+00±1.29e+00(+)	3.45e+01±7.03e+00(+)	1.61e+00±1.14e+00(+)	1.37e+00±6.38e-01(+)	3.46e+00±4.76e-02(+)	6.39e-08±1.64e-08
F4	T1	1.14e+03±6.05e+01(+)	1.45e+02±1.59e+01(+)	1.35e+03±4.61e+02(+)	1.43e+02±2.51e+01(+)	1.40e+02±2.94e+01(+)	1.90e+02±4.07e+00(+)	1.92e+01±3.47e+00
	T2	6.87e+03±2.41e+02(=)	2.16e+03±4.35e+02(-)	7.47e+03±3.48e+02(=)	2.06e+03±4.89e+02(-)	3.40e+03±1.35e+03(-)	1.66e+03±1.14e+01(-)	7.02e+03±9.65e-13
F5	T1	5.10e+01±1.01e+01(+)	1.62e+00±1.08e+00(-)	2.97e+01±1.13e+01(=)	1.28e+00±4.04e-01(-)	2.56e+00±1.40e+00(-)	3.68e+00±6.53e-01(-)	3.14e+01±2.26e+01
	T2	5.95e+00±1.70e+00(+)	1.98e-02±9.90e-03(=)	4.77e-01±8.27e-01(+)	2.28e-02±5.26e-03(=)	5.13e-02±1.84e-02(+)	4.63e-01±1.74e-01(+)	2.08e-02±1.96e-02
F6	T1	1.56e+01±4.12e+00(+)	7.58e-02±1.62e-02(+)	1.28e+01±8.02e-01(+)	5.37e-02±9.79e-03(+)	7.76e-02±9.24e-03(+)	4.17e+00±4.84e-01(+)	2.10e-03±1.08e-03
	T2	6.19e+01±4.41e+00(+)	2.07e+01±2.77e+00(+)	6.14e+01±1.47e+00(+)	2.30e+01±8.42e+00(+)	2.08e+01±3.72e+00(+)	1.02e+01±2.10e-01(+)	6.68e-03±7.19e-03
F7	T1	9.05e+02±1.79e+02(+)	1.17e+02±5.93e+01(-)	2.09e+02±7.59e+01(=)	1.18e+02±7.10e+01(-)	1.45e+02±3.61e+01(-)	6.63e+01±1.02e+01(-)	5.71e+02±1.07e+03
	T2	5.59e+01±4.67e+00(+)	1.25e+00±6.18e-01(+)	4.34e+01±4.08e+00(+)	1.68e+00±1.23e+00(+)	1.66e+00±1.18e+00(+)	5.96e+00±3.79e-01(+)	7.11e-08±3.66e-08
F8	T1	5.54e+01±1.09e+01(+)	1.52e+00±8.57e-01(-)	3.43e+01±6.68e+00(=)	1.42e+00±1.08e+00(-)	1.90e+00±1.01e+00(-)	4.72e+00±8.35e-01(-)	2.01e+01±4.61e+00
	T2	6.35e+03±8.16e+02(=)	1.98e+03±4.50e+02(-)	8.74e+03±4.17e+02(+)	2.35e+03±6.04e+02(-)	2.08e+03±4.15e+02(-)	1.81e+03±8.66e+01(-)	6.77e+03±6.92e+00
+/-/-		14/2/0	9/1/6	11/5/0	9/2/5	10/1/5	10/1/5	

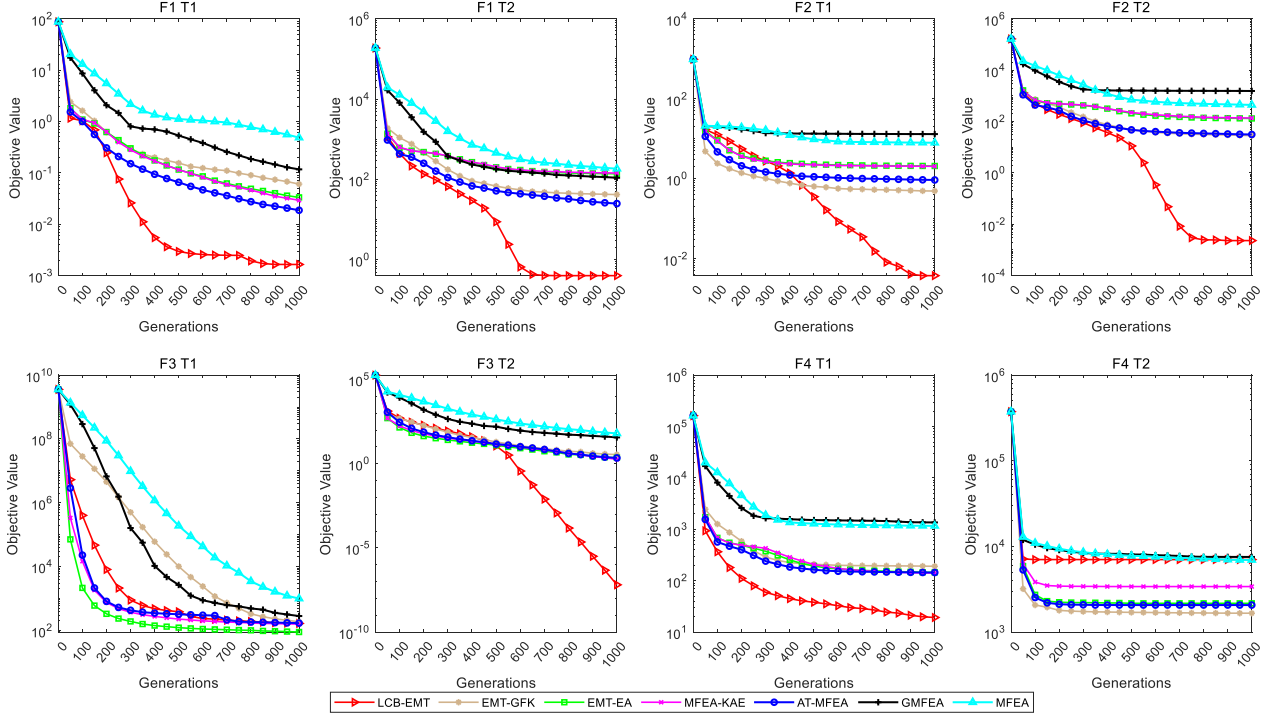


Figure 6.3: Convergence traces of average objective values obtained by MFEA, EMT-GFK, EMT-EA, GMFEA, AT-MFEA, MFEA-KAE, and LCB-EMT on the F series single-objective multitasking instances (F1~F4).

EMT achieved the highest number (11 out of 18) of the best results. Notably, LCB-EMT was found to be statistically significantly better than the compared algorithms in more than half of the cases. Compared to the classical single-objective multitasking benchmark, the F series single-objective multitasking benchmark is more challenging due to the distribution of global optima. LCB-EMT still demonstrated superior performance on this benchmark. Specifically, LCB-EMT achieved the best results in 10 out of 16 test functions, indicating its ability to converge on optimal solutions for the majority of the test functions. These findings demonstrate that LCB-EMT is a promising algorithm for solving single-objective multitasking optimization problems.

Secondly, the performance of our proposed algorithm is promising regardless of the inter-task

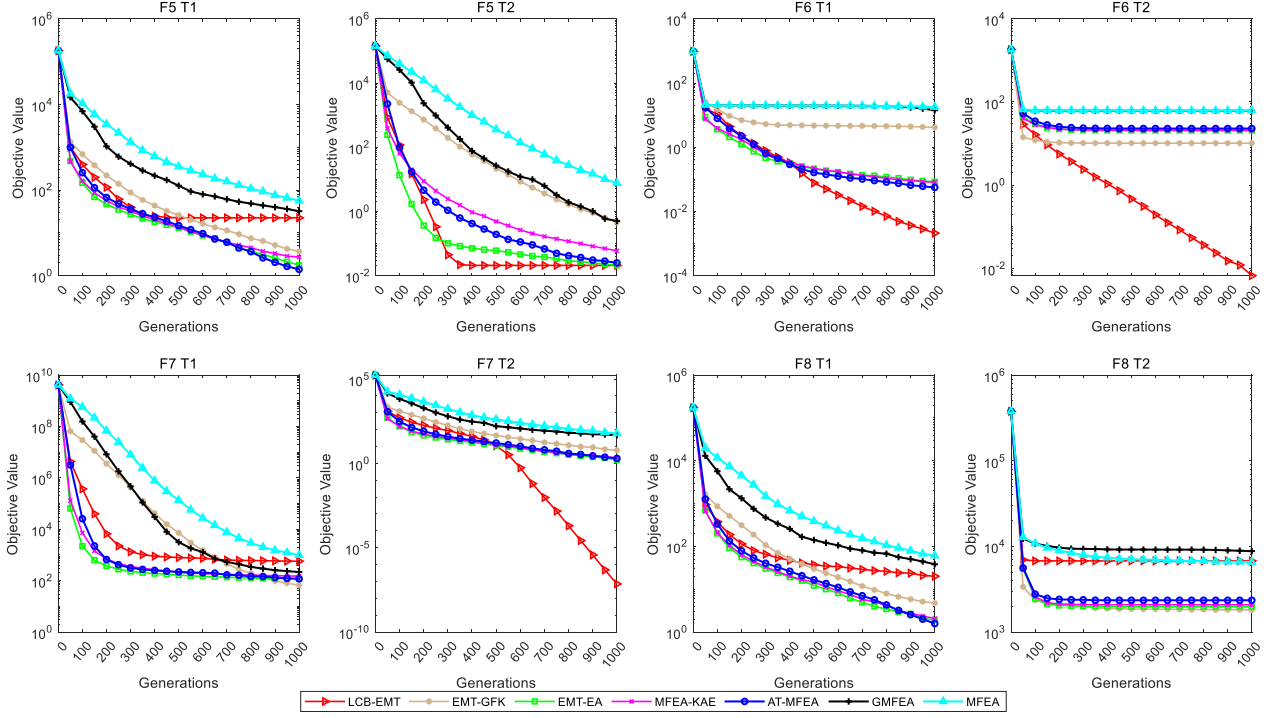


Figure 6.4: Convergence traces of average objective values obtained by MFEA, EMT-GFK, EMT-EA, GMFEA, AT-MFEA, MFEA-KAE, and LCB-EMT on the F series single-objective multitasking instances (F5-F8).

similarity. This is because our LCB-EMT algorithm does not rely solely on inter-task similarity to transfer solutions. Instead, it evaluates the quality of solutions by leveraging the inter-task similarity and distribution information. Therefore, the effectiveness of LCB-EMT is not limited by the similarity between tasks, making it suitable for solving various multitasking optimization problems.

Lastly, we plot the convergence traces of average objective values obtained by the seven compared algorithms over 1000 generations. The convergence traces on the F1-F8 test instances are shown in Fig. 6.3 and Fig. 6.4. From the convergence traces, we can find that, compared with other methods, the objective values of our algorithm can continue to decline while other algorithms easily fall into local optimum. This is because our method transfers the solutions

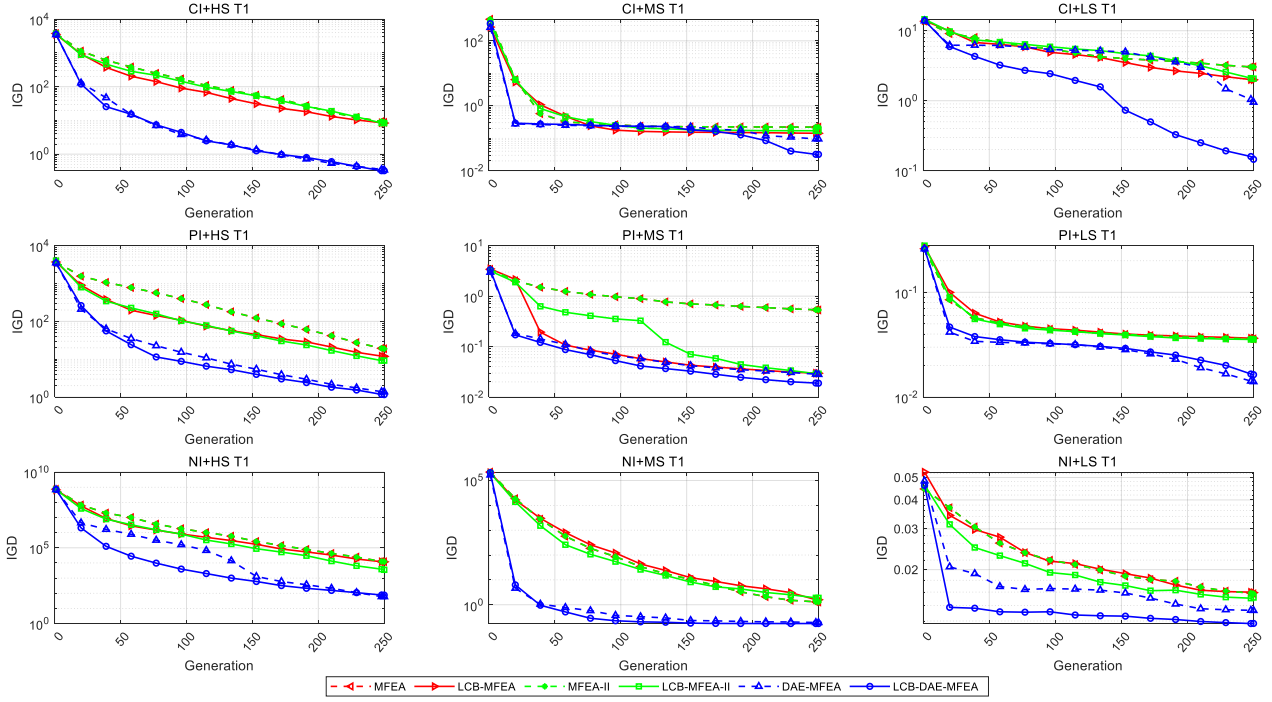


Figure 6.5: Convergence traces of IGD values obtained by MFEA, LCB-MFEA, MFEA-II, LCB-MFEA-II, DAE-MFEA, and LCB-DAE-MFEA on task 1 of multiobjective multitasking instances.

that can guide the population to escape from the local optima.

2) Multiobjective multitasking benchmark: To examine the generality of the proposed solution selection model and its scalability on the multiobjective optimization problems, we incorporated the solution selection approach into MFEA, MFEA-II, and DAE-MFEA [193] and conducted experiments on a multiobjective multitasking benchmark. These algorithms were renamed LCB-MFEA, LCB-MFEA-II, and LCB-DAE-MFEA, respectively. For these algorithms, we configure the population size as 100 and the maximum generation as 250.

The results, presented in Table 6.3, show that LCB-MFEA, LCB-MFEA-II, and LCB-DAE-MFEA outperform their counterparts in terms of IGD performance. However, the improvement of LCB-DAE-MFEA is insignificant compared to DAE-MFEA on half of the test problems. This

Table 6.3: Mean IGD values and standard deviation obtained by ablation algorithms on the multiobjective multitask problems.

Problem	Task	MFEA	LCB-MFEA	MFEA-II	LCB-MFEA-II	DAE-MFEA	LCB-DAE-MFEA
CI+HS	T1	8.78e+00±1.40e+00(=)	8.36e+00±4.52e-01	8.78e+00±1.40e+00(=)	8.36e+00±1.00e+00	3.40e-01±5.16e-02(=)	3.23e-01±2.66e-01
	T2	4.99e-01±4.67e-02(=)	5.04e-01±4.15e-02	4.99e-01±4.67e-02(=)	5.09e-01±4.78e-02	1.16e-01±7.47e-03(=)	1.05e-01±4.02e-02
CI+MS	T1	2.20e-01±2.05e-02(+)	1.42e-01±1.16e-01	2.20e-01±2.05e-02(+)	1.70e-01±2.66e-02	9.52e-02±1.08e-01(=)	3.18e-02±1.57e-02
	T2	1.03e-02±1.15e-03(=)	1.36e-02±8.93e-03	1.03e-02±1.15e-03(-)	1.56e-02±7.03e-03	1.03e-02±5.55e-03(+)	6.21e-03±1.17e-03
CI+LS	T1	3.04e+00±2.68e-01(+)	1.99e+00±3.80e-01	3.04e+00±2.68e-01(+)	2.07e+00±1.84e-01	9.52e-01±8.05e-01(+)	1.45e-01±2.02e-02
	T2	7.61e-03±8.55e-05(=)	7.54e-03±8.61e-04	7.61e-03±8.55e-05(+)	5.60e-03±1.21e-03	3.24e-03±1.69e-03(+)	9.66e-04±6.78e-05
PI+HS	T1	1.89e+01±4.37e+00(+)	1.19e+01±1.06e+00	1.89e+01±4.37e+00(+)	9.22e+00±1.92e+00	1.39e+00±5.33e-01(=)	1.16e+00±5.91e-01
	T2	3.12e+01±5.06e+00(+)	2.60e+01±9.13e-01	3.12e+01±5.06e+00(+)	2.10e+01±2.66e+00	1.04e+01±5.13e-01(+)	9.49e+00±1.26e+00
PI+MS	T1	5.32e-01±5.37e-02(+)	2.97e-02±6.68e-03	5.32e-01±5.37e-02(+)	2.91e-02±8.19e-03	2.81e-02±1.82e-03(+)	1.89e-02±2.23e-03
	T2	7.49e+01±3.32e+00(+)	9.86e-02±1.60e-01	7.49e+01±3.32e+00(+)	3.04e-01±5.15e-01	2.03e-04±1.52e-05(+)	1.95e-04±6.70e-06
PI+LS	T1	3.60e-02±2.17e-04(-)	3.65e-02±2.42e-04	3.60e-02±2.17e-04(+)	3.55e-02±1.41e-04	1.40e-02±1.23e-03(-)	1.64e-02±2.99e-03
	T2	6.19e-01±9.12e-03(+)	6.00e-01±6.73e-03	6.19e-01±9.12e-03(-)	6.34e-01±1.88e-03	4.50e-01±5.52e-02(=)	5.00e-01±9.52e-02
NI+HS	T1	1.21e+04±4.26e+03(=)	1.19e+04±2.44e+03	1.21e+04±4.26e+03(+)	3.53e+03±9.56e+02	5.90e+01±5.51e+01(=)	7.54e+01±1.94e+01
	T2	8.99e+00±2.31e+00(=)	9.04e+00±1.78e+00	8.99e+00±2.31e+00(+)	5.06e+00±7.10e-01	3.69e-01±3.15e-01(=)	4.49e-01±6.78e-02
NI+MS	T1	1.28e+00±3.25e-01(-)	1.59e+00±1.23e-01	1.28e+00±3.25e-01(-)	1.89e+00±9.40e-01	1.93e-01±6.67e-03(+)	1.75e-01±1.09e-03
	T2	3.66e-01±3.28e-01(+)	1.56e-01±4.73e-02	3.66e-01±3.28e-01(=)	3.39e-01±3.03e-01	7.08e-03±7.17e-03(=)	2.91e-03±2.72e-03
NI+LS	T1	1.57e-02±6.74e-04(=)	1.59e-02±9.88e-04	1.57e-02±6.74e-04(=)	1.52e-02±1.31e-04	1.33e-02±1.94e-04(+)	1.17e-02±1.03e-04
	T2	6.47e-01±1.80e-03(+)	6.43e-01±6.26e-04	6.47e-01±1.80e-03(+)	6.44e-01±1.62e-03	1.69e-01±1.84e-02(-)	6.42e-01±9.93e-03
+/-/=/-		9/7/2		11/4/3		8/8/2	

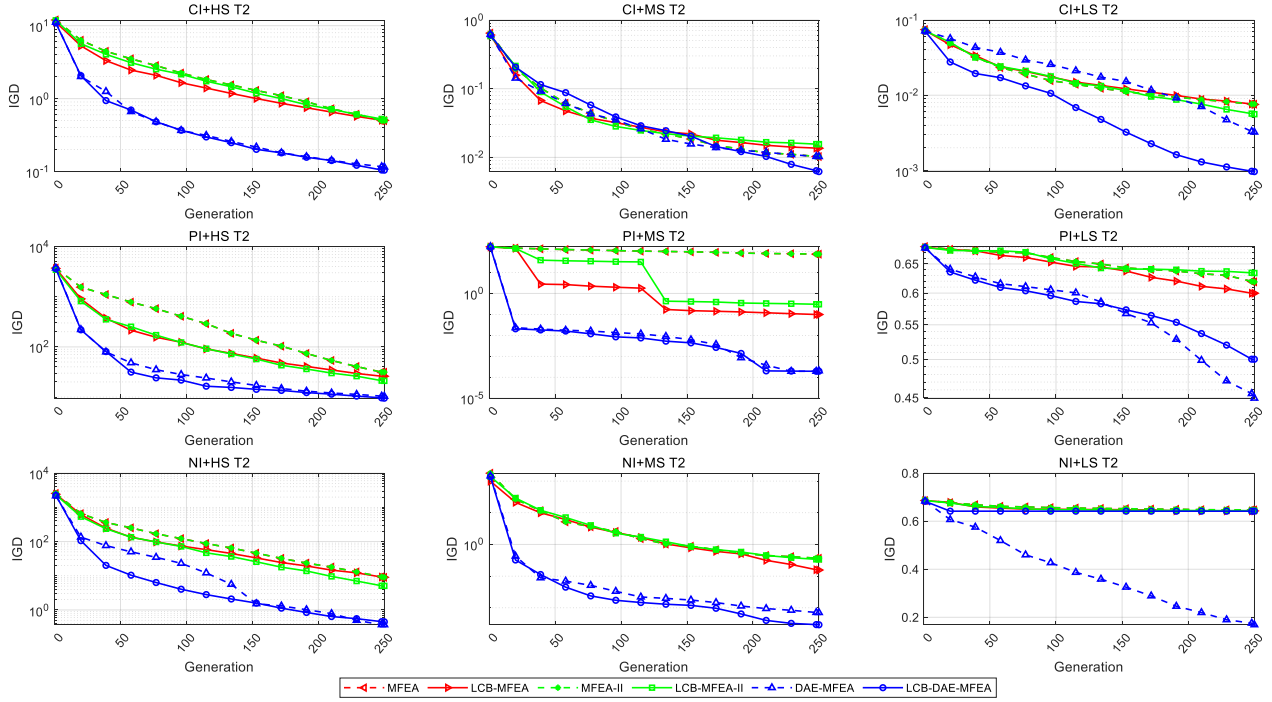


Figure 6.6: Convergence traces of IGD values obtained by MFEA, LCB-MFEA, MFEA-II, LCB-MFEA-II, DAE-MFEA, and LCB-DAE-MFEA on task 2 of multiobjective multitasking instances.

is because DAE-MFEA combines various domain adaptation methods to transfer solutions, thus obtaining a set of transferred solutions with diversity and can better explore and exploit the search space of the target task. In addition, the convergence traces of IGD values obtained by MFEA, LCB-MFEA, MFEA-II, LCB-MFEA-II, DAE-MFEA, and LCB-DAE-MFEA on the multiobjective multitasking benchmark are presented in Fig. 6.5 and Fig. 6.6.

Based on the experimental results presented above, our proposed solution selection approach exhibits promising performance not only on the single-objective multitasking benchmarks but also on the multiobjective multitasking benchmark.

Table 6.4: Mean objective values and standard deviation obtained by ablation algorithms on the F series single-objective multitask problems.

Problem	Task	SOEA	w / o σ, λ	w / o λ	LCB-EMT	I. R. to w/o λ
F1	T1	5.06e-01 $\pm$ 3.13e-01(+)	8.03e-03 $\pm$ 1.31e-02(+)	2.94e-04$\pm$5.79e-04(-)	1.65e-03 $\pm$ 4.92e-03	-
	T2	1.66e+03 $\pm$ 2.06e+02(+)	1.24e+01 $\pm$ 2.47e+01(+)	6.81e-03$\pm$1.28e-02(-)	4.09e-01 $\pm$ 8.00e-01	-
F2	T1	2.08e+01 $\pm$ 2.01e+01(+)	7.59e-03 $\pm$ 7.34e-03(+)	1.01e-02 $\pm$ 1.45e-02(+)	3.80e-03$\pm$6.53e-03	62.38%
	T2	1.84e+03 $\pm$ 4.06e+02(+)	4.62e-02 $\pm$ 5.59e-02(+)	4.46e-02 $\pm$ 4.42e-02(+)	2.32e-03$\pm$3.73e-03	94.80%
F3	T1	8.70e+04 $\pm$ 1.03e+05(+)	7.33e+01$\pm$4.99e+01(-)	7.45e+02 $\pm$ 2.53e+02(+)	1.65e+02 $\pm$ 1.42e+02	77.85%
	T2	1.98e+03 $\pm$ 4.25e+02(+)	4.34e-05 $\pm$ 4.76e-05(+)	6.15e+01 $\pm$ 1.23e+02(+)	6.39e-08$\pm$1.64e-08	99.99%
F4	T1	1.80e+03 $\pm$ 3.11e+02(+)	2.00e+01 $\pm$ 5.30e+00(=)	8.96e+01 $\pm$ 2.97e+01(+)	1.92e+01$\pm$3.47e+00	78.57%
	T2	3.49e+05 $\pm$ 1.23e+03(+)	7.03e+03 $\pm$ 8.66e-01(=)	1.05e+04 $\pm$ 2.53e+00(+)	7.02e+03$\pm$9.65e-13	33.14%
F5	T1	1.73e+03 $\pm$ 3.96e+02(+)	4.26e+01 $\pm$ 1.13e+01(=)	1.58e+02 $\pm$ 5.56e+01(+)	3.14e+01$\pm$2.26e+01	80.13%
	T2	8.14e-01 $\pm$ 4.39e-01(+)	4.39e+00 $\pm$ 7.51e+00(+)	5.84e+01 $\pm$ 6.80e+01(+)	2.08e-02$\pm$1.96e-02	99.96%
F6	T1	1.39e+01 $\pm$ 9.16e+00(+)	1.61e-03$\pm$6.67e-04(=)	8.74e-03 $\pm$ 2.82e-03(+)	2.10e-03 $\pm$ 1.08e-03	75.97%
	T2	2.11e+01 $\pm$ 2.88e+00(+)	6.13e-03$\pm$3.39e-03(=)	8.41e-03 $\pm$ 2.09e-02(+)	6.68e-03 $\pm$ 7.19e-03	20.57%
F7	T1	1.13e+04 $\pm$ 2.85e+03(+)	2.78e+02$\pm$3.55e+02(-)	4.32e+02 $\pm$ 1.74e+02(=)	5.71e+02 $\pm$ 1.07e+03	=
	T2	1.78e+03 $\pm$ 2.58e+02(+)	5.45e-06 $\pm$ 1.07e-05(+)	4.26e-07 $\pm$ 1.38e-07(+)	7.11e-08$\pm$3.66e-08	83.31%
F8	T1	1.93e+03 $\pm$ 3.81e+02(+)	2.11e+01 $\pm$ 2.92e+00(=)	9.78e+01 $\pm$ 2.59e+01(+)	2.01e+01$\pm$4.61e+00	79.45%
	T2	3.49e+05 $\pm$ 2.13e+03(+)	6.79e+03 $\pm$ 6.54e+00(=)	1.02e+04 $\pm$ 4.02e+02(+)	6.77e+03$\pm$6.92e+00	33.63%
+/-/-		16/0/0	7/7/2	13/1/2		

6.4.3 Ablation Studies

In this subsection, we perform ablation experiments to demonstrate the effectiveness of the proposed LCB-EMT. One of the important features of the proposed work is that we combine the prediction expectation value μ , the prediction uncertainty σ , and the inter-task similarity metric λ to evaluate the quality of transferred solutions for the target task. To verify the effectiveness of each metric, we decompose the proposed algorithm into two ablation algorithms, w / o σ, λ and w / o λ , where w / o σ, λ and w / o λ respectively use the following equations to calculate $SLCB$ value for evaluating solutions for knowledge transfer,

$$\begin{aligned}
 SLCB(x) &= f(x), \\
 SLCB(x) &= f(x) - \sigma(x).
 \end{aligned}
 \tag{6.22}$$

Table 6.5: Mean objective values and standard deviation obtained by ablation algorithms on the classical single-objective multitask problems.

Problem	Task	SOEA	w / o σ, λ	w / o λ	LCB-EMT	I. R. to w / o λ
CI+HS	T1	3.42e-01±5.14e-01(+)	3.08e-03±6.16e-03(+)	5.91e-07±1.17e-06(+)	1.31e-07±2.58e-07	77.80%
	T2	1.75e+03±4.15e+02(+)	1.19e+00±2.37e+00(+)	2.96e-01±5.85e-01(+)	1.53e-03±4.28e-03	99.48%
CI+MS	T1	1.47e+01±8.73e+00(+)	2.37e-03±3.13e-03(+)	9.25e-04±5.21e-04(=)	8.07e-04±9.11e-04	12.76%
	T2	1.86e+03±4.08e+02(+)	5.41e-01±4.18e-03(=)	5.87e-01±1.47e-02(=)	5.48e-01±1.63e+00	6.64%
CI+LS	T1	9.68e+00±3.71e+00(+)	5.30e-03±1.49e-03(=)	6.15e-03±1.76e-03(=)	5.13e-03±4.60e-03	16.59%
	T2	3.48e+05±3.26e+03(+)	6.77e+03±5.10e+03(=)	6.77e+03±4.18e+03(+)	5.27e+03±2.98e+03	22.16%
PI+HS	T1	1.73e+03±2.65e+02(+)	3.63e+01±1.86e+01(=)	3.51e+01±2.86e+01(=)	2.92e+01±2.90e+01	16.81%
	T2	8.69e-01±3.01e-01(+)	4.16e+00±8.27e+00(+)	1.92e+02±3.82e+02(+)	2.74e-02±2.51e-02	99.99%
PI+MS	T1	1.78e+01±2.17e+01(+)	5.18e-03±6.54e-04(=)	4.53e-02±4.78e-02(+)	4.47e-03±8.84e-04	90.13%
	T2	2.93e+04±2.32e+04(+)	4.97e+01±2.89e+01(=)	1.58e+02±1.62e+02(+)	4.79e+01±5.33e+01	69.68%
PI+LS	T1	1.24e+01±8.91e+00(+)	4.67e-06±4.40e-06(-)	1.26e-06±1.31e-06(-)	3.05e-05±6.29e-05	-
	T2	2.21e+01±5.51e+00(+)	8.23e-03±1.18e-02(-)	6.12e-03±3.75e-03(-)	9.31e-01±1.84e+00	-
NI+HS	T1	2.25e+04±2.41e+04(+)	2.30e+02±2.41e+02(=)	3.17e+02±7.92e+01(=)	3.47e+02±2.87e+02	=
	T2	1.77e+03±2.68e+02(+)	7.83e-07±1.43e-06(+)	1.05e+01±2.09e+01(+)	5.55e-08±2.90e-08	99.99%
NI+MS	T1	7.14e-01±1.17e+00(+)	3.96e-10±7.77e-10(+)	1.47e-13±2.14e-13(-)	1.96e-12±5.09e-12	-
	T2	2.08e+01±2.79e+00(+)	1.91e+00±3.78e+00(+)	1.82e-01±3.61e-01(+)	2.98e-03±3.50e-03	98.36%
NI+LS	T1	1.87e+03±4.22e+02(+)	1.53e+01±2.93e+00(=)	1.90e+01±8.52e+00(=)	1.50e+01±9.38e+00	21.05%
	T2	3.48e+05±2.87e+03(+)	6.77e+03±5.59e+00(=)	6.77e+03±3.33e+03(+)	5.26e+03±2.98e+03	22.30%
+/-/-		18/0/0	7/9/2	9/6/3		

Besides, we developed a simple multitasking algorithm, SOEA, as the baseline, where SOEA uses two separate solvers with DE and the genetic algorithm to optimize the two tasks.

The experimental results of SOEA, w / o σ, λ , w / o λ , and LCB-EMT on the F series benchmark and classical benchmark can be seen in Table 6.4 and Table 6.5, respectively. As can be observed from the experimental results, we can first find that all knowledge transfer-based methods exhibit their effectiveness over SOEA without knowledge transfer. Moreover, LCB-EMT can perform knowledge transfer more effectively than w / o σ, λ and w / o λ in most cases. However, we find that the performance of LCB-EMT is worse than that of w / o λ on F1, F6, and F7 instances. This is because, in these EMT problems, the inter-task similarity value λ obtained by our algorithm is far from the real one, and we will verify it later.

Inter-task similarity λ determines the intensity of knowledge transfer (i.e., the number of solutions that can be transferred) and is a significant factor in affecting performance. The λ_{gen} and mean λ over all generations are depicted in Fig. 6.7. Compared with the similarity metric obtained by Spearman's rank correlation coefficient, we can see that the estimated λ obtained by our proposed method is close to the similarity obtained by Spearman's rank correlation coefficient in most cases. However, the λ estimated by LCB-EMT on F1, F6, and F7 is far from the similarity metric obtained by Spearman's rank correlation coefficient. Therefore, this may be the reason for the performance degradation of LCB-EMT on F1, F6, and F7 instances.

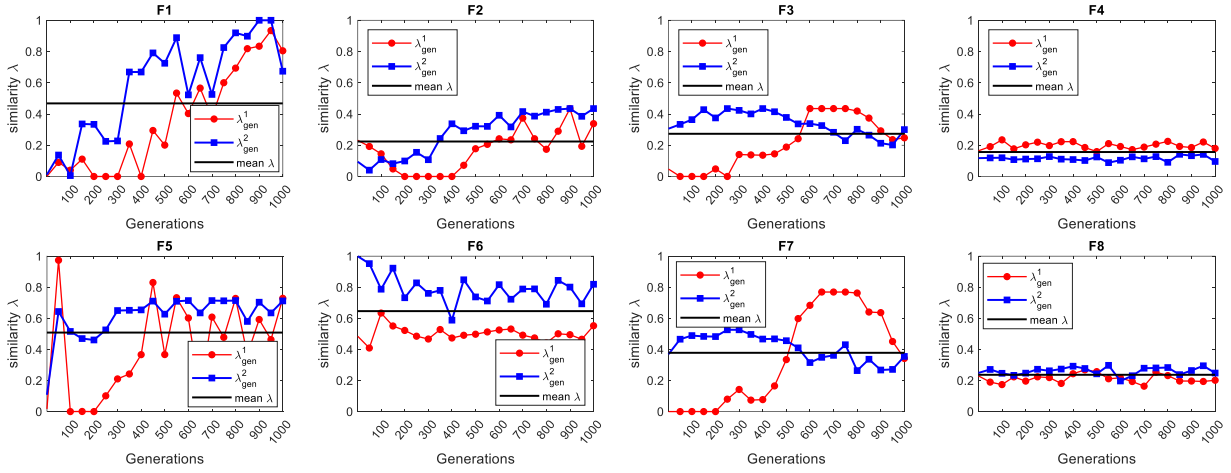


Figure 6.7: Inter-task similarity λ estimated by the proposed LCB-EMT. λ_{gen}^1 and λ_{gen}^2 denote the estimated similarity on the first task and the second task under the generation index gen , respectively, and $\text{mean } \lambda = \frac{\sum_{gen} (\lambda_{gen}^1 + \lambda_{gen}^2)}{2 \sum_{gen} 1}$.

To show the relationship between the algorithm performance and the inter-task similarity, we have calculated the LCB-EMT's improvement rate to the counterpart $\bar{w} / o \lambda$ (I. R. to $w / o \lambda$ in Table 6.4 and Table 6.5), so that the relative performance changes of the proposed inter-task similarity based LCB on various similarity can be demonstrated. From Table 6.5, it is obvious that the HS (high similarity) and MS (medium similarity) groups often achieve better results than the LS (low similarity) group on the classical single-objective multitasking benchmark.

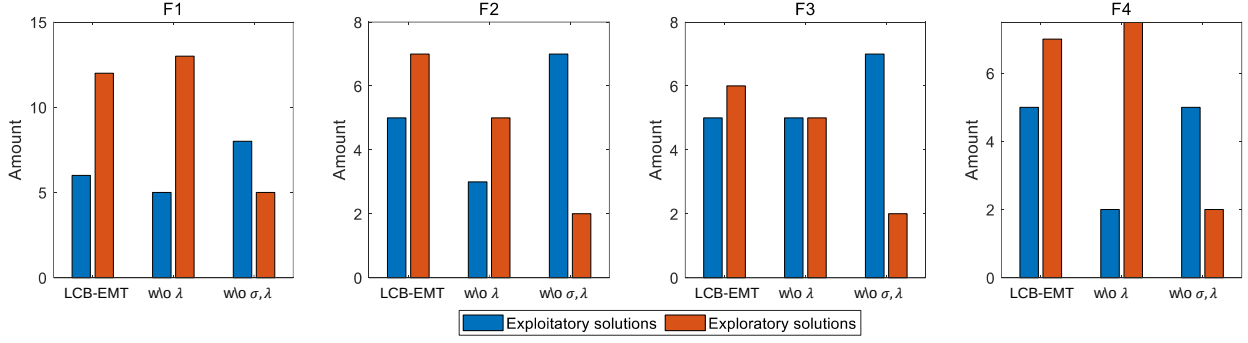


Figure 6.8: The average number of exploratory solutions and exploitative solutions on F1~F4 instances when the knowledge transfer is triggered.

This indicates that the inter-task similarity λ encourages the population to explore the search space and help to escape the local optima under the HS and MS cases, which matches our intuition shown in Fig. 1. From Table 6.4, LCB-EMT achieves great improvement in most cases, except for F1 with the inter-task similarity of 1. This may be due to the inaccurate similarity estimation (shown in Fig. 6.7), which causes the promising solutions not to be transferred to explore the target search space of F1.

The intuition behind the proposed LCB-EMT is that the solutions in less-explored regions have lower SLCB values, i.e., lower objective values or larger uncertainty. Therefore, transferring solutions to these regions has the potential to improve population diversity or convergence. We defined two types of solutions, exploratory solutions and exploitative solutions, and then counted the number of solutions that can improve population diversity or convergence to verify the effectiveness of the proposed LCB-EMT.

We propose a metric, dubbed *Div*, to identify whether a solution is an exploratory solution. The *Div* metric indicates the diversity of a population P ,

$$Div(P) = \frac{\sum_i \sum_j \|x_i - x_j\|_2}{|P|}, \quad (6.23)$$

where x_i is the decision variable of individual P_i . The larger the $Div(P)$, the wider the distribution of solutions in the decision space, and the easier P is to escape from the local optima.

If transferring a solution can induce a larger Div , we regard it as an exploratory solution. On the other hand, if transferring a solution can improve population convergence, we regard it as an exploitative solution.

Fig. 6.8 shows our statistical results of compared algorithms in selecting exploratory solutions and exploitative solutions. We can observe that $w/o \sigma, \lambda$ often transfer more exploitative solutions, while $w/o \lambda$ can transfer more exploratory solutions. LCB-EMT can balance the two types of solutions, thus achieving the balance of exploration-exploitation to suppress negative transfer triggered by local optima and slow convergence. Specifically, the proposed LCB-EMT is not biased toward a certain type of solutions for knowledge transfer, thus preventing the population from over-exploration or over-exploitation. $w/o \sigma, \lambda$ selects solutions based on the objective values, thus likely transferring exploitative solutions and easily leading to local optima. $w/o \lambda$ pays more attention to prediction uncertainty than $w/o \sigma, \lambda$, thus transferring more exploratory solutions than $w/o \sigma, \lambda$. Compared with $w/o \sigma, \lambda$ (or $w/o \lambda$), LCB-EMT can transfer more exploratory solutions (or exploitative solutions), thus balancing exploration and exploitation.

6.4.4 Generality Analysis

In this subsection, we aim to demonstrate the generality of our proposed solution selection approach with several existing single-population-based EMT algorithms on the two single-objective multitasking benchmarks. Specifically, we incorporate the solution selection approach into existing MFEA-based algorithms, GMFEA, AT-MFEA, and MFEA-KAE, and rename them as LCB-GMFEA, LCB-AT-MFEA, and LCB-MFEA-KAE, respectively. The results are presented in Tables 6.6 and 6.7, respectively. We can observe that LCB-GMFEA, LCB-AT-MFEA, and LCB-MFEA-KAE demonstrate comparable or superior performance to their counterparts on most of the test problems, with only a few instances where they performed worse. These findings provide evidence of the generality of our proposed LCB-based solution selection

approach for enhancing the performance of existing EMT algorithms.

6.4.5 Real-World Applications

The beneficiation process is an industrial process of extracting useful material from raw ore to obtain qualified concentrates [194, 9]. In particular, raw ore can be processed in multiple operational environments with different conditions [9]. Due to the similarity between different operational environments, these tasks can be simultaneously optimized by exploiting the latent synergies between them. For each optimization task under a specified operational condition, we aim to maximize the concentrate yield $C(x)$, metal recovery $M(x)$, and concentrate grade $G(x)$ under one constraint low-intensity magnetic separation $S(x)$,

$$\{\max C(x), \max M(x), \max G(x)\} \quad s.t. \quad S(x) \leq S_{upper}, \quad (6.24)$$

where $x = (x_1, \dots, x_I)$ is the decision variable denoting the quantity of the i -th raw ore. S_{upper} is the upper bound for low-intensity magnetic separation. $C(x)$, $M(x)$, $G(x)$, and $S(x)$ can be mathematical formulated as follows respectively,

$$C(x) = \sum_{i=1}^I \frac{(1 - u_i)x_i}{k_{1,i}} + \sum_{i=1}^I \frac{u_i x_i}{k_{2,i}}, \quad (6.25)$$

where $k_{1,i}$ and $k_{2,i}$ represent the beneficiation ratio of powder ore and lump ore, respectively. u_i is the ratio of lump ore obtained from raw ore.

$$M(x) = \frac{\sum_{i=1}^I (1 - u_i)\beta_{1,i}x_i/k_{1,i} + \sum_{i=1}^I u_i\beta_{2,i}x_i/k_{2,i}}{\sum_{i=1}^I \alpha_i x_i}, \quad (6.26)$$

where α_i is the grade of raw ore, and β_i is the grade of concentrate ore.

$$G(x) = \frac{\sum_{i=1}^I (1 - u_i)\beta_{1,i}x_i/k_{1,i} + \sum_{i=1}^I u_i\beta_{2,i}x_i/k_{2,i}}{\sum_{i=1}^I (1 - u_i)x_i/k_{1,i} + \sum_{i=1}^I u_i x_i/k_{2,i}}. \quad (6.27)$$

Table 6.6: Mean objective values and standard deviation obtained by ablation algorithms on the F series single-objective multitask problems.

Problem	Task	AT-MFEA	LCB-AT-MFEA	KAE-MFEA	LCB-KAE-MFEA	GMFEA	LCB-GMFEA
F1	T1	1.51e-02±2.16e-03(+)	7.99e-03±5.71e-03	2.86e-02±6.30e-03(+)	1.35e-02±2.51e-03	8.39e-02±2.98e-02(+)	4.80e-02±1.09e-02
	T2	3.32e+01±1.03e+01(+)	1.42e+01±1.17e+01	1.47e+02±1.32e+01(+)	3.43e+01±7.74e+00	8.50e+01±7.20e+01(=)	8.51e+01±3.37e+01
F2	T1	1.74e+00±4.08e-01(+)	8.59e-01±7.25e-01	2.21e+00±3.47e-02(+)	2.02e+00±1.12e-01	1.32e+01±1.48e+00(+)	1.10e+01±1.93e+00
	T2	2.33e+01±1.19e+01(=)	2.41e+01±2.34e+01	1.26e+02±9.04e+00(+)	3.66e+01±1.15e+01	1.56e+03±6.67e+02(+)	8.91e+02±4.17e+02
F3	T1	1.85e+02±2.23e+01(+)	1.75e+02±2.24e+01	1.77e+02±4.06e+01(=)	1.65e+02±2.14e+01	2.28e+02±4.04e+01(+)	1.41e+02±3.48e+01
	T2	1.39e+00±8.28e-01(-)	1.80e+00±8.06e-01	1.91e+00±1.73e-01(=)	2.03e+00±2.99e-01	1.65e+01±1.51e+00(=)	2.26e+01±7.00e+00
F4	T1	1.62e+02±2.19e+01(+)	1.37e+02±2.18e+01	1.57e+02±3.30e+01(+)	1.26e+02±1.57e+01	1.15e+03±8.66e+01(+)	9.60e+02±8.17e+01
	T2	2.09e+03±4.56e+02(=)	2.30e+03±2.63e+02	3.74e+03±1.21e+03(+)	2.32e+03±9.48e+02	6.52e+03±1.02e+03(=)	7.55e+03±3.55e+02
F5	T1	1.44e+00±4.75e-01(+)	6.47e-01±6.45e-02	2.86e+00±1.04e+00(+)	1.52e+00±4.39e-01	2.07e+01±4.75e-01(+)	1.80e+01±3.82e+00
	T2	3.60e-02±1.89e-02(+)	2.58e-02±1.31e-02	6.71e-02±2.99e-02(+)	2.99e-02±3.23e-03	4.38e-02±1.98e-02(+)	3.04e-02±1.40e-02
F6	T1	5.33e-02±5.93e-03(+)	4.94e-02±2.34e-03	8.09e-02±6.01e-03(+)	6.46e-02±8.85e-03	1.79e+00±5.73e-02(+)	1.59e+00±1.84e-01
	T2	3.53e+01±8.59e+00(+)	2.24e+01±4.16e+00	2.31e+01±1.67e+00(-)	2.93e+01±3.50e+00	5.78e+01±2.44e+00(=)	5.89e+01±2.31e+00
F7	T1	9.80e+01±5.23e+01(+)	4.97e+01±3.64e-01	1.58e+02±2.43e+00(+)	7.17e+01±3.28e+01	3.16e+02±1.60e+02(+)	1.84e+02±3.28e+01
	T2	1.56e+00±1.27e+00(=)	9.61e-01±5.44e-01	2.30e+00±8.44e-01(+)	1.59e+00±5.49e-01	1.80e+01±1.74e+00(-)	2.25e+01±4.04e+00
F8	T1	2.10e+00±4.99e-01(+)	1.35e+00±5.89e-01	2.39e+00±5.83e-01(+)	1.30e+00±4.96e-01	2.05e+01±2.79e+00(+)	1.64e+01±8.75e-01
	T2	2.34e+03±3.60e+02(-)	2.69e+03±2.50e+02	2.02e+03±6.78e+02(-)	2.62e+03±4.74e+02	7.11e+03±1.11e+02(=)	7.31e+03±7.65e+02
+/-/-		11/3/2		12/2/2		10/5/1	

Table 6.7: Mean objective values and standard deviation obtained by ablation algorithms on the classical single-objective multitask problems.

Problem	Task	AT-MFEA	LCB-AT-MFEA	KAE-MFEA	LCB-KAE-MFEA	GMFEA	LCB-GMFEA
CI+HS	T1	2.50e-03 ±3.09e-03(-)	3.88e-03±4.19e-03	4.43e-03±3.98e-03(+)	1.22e-04 ±4.91e-05	3.72e-02±1.72e-02(+)	6.56e-03 ±2.78e-03
	T2	8.23e-01±9.00e-01(+)	5.26e-01 ±7.20e-01	1.66e+00±9.40e-01(+)	2.65e-01 ±1.41e-01	1.79e+01±6.96e+00(+)	1.21e+01 ±3.17e+00
CI+MS	T1	1.58e+00±8.30e-02(=)	1.35e+00 ±3.98e-01	2.38e+00±3.90e-02(+)	1.60e+00 ±8.85e-02	1.93e+01±3.04e-01(=)	1.91e+01 ±2.65e-01
	T2	2.00e+00 ±1.24e+00(=)	2.02e+00±1.99e+00	1.74e+00±9.30e-01(=)	1.66e+00 ±9.82e-01	2.07e+01±2.73e+00(+)	1.80e+01 ±1.90e+00
CI+LS	T1	1.59e+00±3.44e-01(+)	1.09e+00 ±2.05e-01	2.02e+00±1.31e-01(+)	1.69e+00 ±7.82e-02	2.00e+01 ±1.24e-02(=)	2.00e+01±2.23e-02
	T2	2.01e+03 ±5.56e+02(=)	2.24e+03±1.31e+02	2.27e+03 ±1.69e+02(-)	2.69e+03±1.19e+02	6.73e+03 ±3.51e+02(=)	6.92e+03±6.80e+02
PI+HS	T1	1.88e-01±6.99e-02(+)	1.09e-01 ±4.79e-02	2.01e+00±6.12e-01(+)	2.76e-01 ±1.23e-02	6.65e+00±4.96e+00(+)	3.86e+00 ±2.02e+00
	T2	3.79e-03±1.41e-03(+)	2.22e-03 ±9.59e-04	5.15e-02±1.27e-02(+)	5.52e-03 ±2.07e-04	1.39e-01±1.07e-01(+)	7.85e-02 ±4.14e-02
PI+MS	T1	1.71e+00±4.08e-01(+)	5.97e-01 ±4.84e-01	2.34e+00±4.53e-01(+)	1.52e+00 ±3.03e-01	1.91e+01±1.70e-01(=)	1.90e+01 ±1.13e-01
	T2	1.61e+02 ±3.44e+01(=)	1.58e+03±2.47e+03	1.43e+02 ±2.89e+01(-)	7.88e+02±1.03e+03	2.97e+02±2.32e+01(+)	2.18e+02 ±7.25e+01
PI+LS	T1	1.71e+00±1.27e-01(+)	1.45e+00 ±2.01e-01	2.37e+00±2.47e-01(+)	1.57e+00 ±2.39e-01	1.95e+01±4.04e-01(=)	1.91e+01 ±1.63e-01
	T2	2.81e-01 ±7.78e-02(=)	2.96e-01±6.30e-02	4.60e-01 ±5.27e-02(-)	9.54e-01±7.43e-02	2.50e+00±5.05e-01(+)	1.48e+00 ±1.62e-01
NI+HS	T1	9.22e+01±3.77e+01(+)	5.90e+01 ±1.79e+01	2.29e+02±1.34e+01(+)	7.25e+01 ±4.11e+01	1.68e+02±4.81e+01(=)	1.65e+02 ±5.25e+01
	T2	1.17e+00±5.98e-01(=)	8.17e-01 ±7.27e-01	2.30e+00±1.39e+00(=)	1.82e+00 ±8.12e-01	2.10e+01 ±3.05e+00(-)	2.29e+01±2.09e+00
NI+MS	T1	4.15e-04±3.15e-04(+)	2.08e-05 ±7.20e-06	1.41e-02±6.79e-03(+)	8.41e-05 ±3.06e-05	4.32e-02±6.08e-02(+)	2.21e-03 ±5.14e-04
	T2	2.63e-01±5.89e-02(+)	1.26e-01 ±3.38e-02	5.09e-01±4.33e-02(+)	2.54e-01 ±4.14e-02	1.32e+00±2.24e-01(+)	8.45e-01 ±1.29e-01
NI+LS	T1	1.59e+00±7.91e-01(+)	9.67e-01 ±6.25e-01	1.29e+00 ±3.19e-01(-)	1.80e+00±3.74e-01	2.06e+01±2.63e+00(+)	1.62e+01 ±1.81e+00
	T2	1.95e+03 ±1.72e+02(=)	2.13e+03±3.74e+02	3.34e+03±5.16e+02(+)	2.06e+03 ±5.24e+02	6.97e+03±5.77e+02(+)	4.37e+03 ±2.86e+03
+/-/-		10/8/0		12/2/4		11/6/1	

Table 6.8: Five conditions for raw ore allocation of mineral processing.

Series Number	Parameters	Condition 1	Condition 2	Condition 3	Condition 4	Condition 5
1	Capacity (N_k)	80.60	58.56	36.52	36.52	36.52
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
2	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
3	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	2.08	2.08	2.08	2.08	2.08
4	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	25.37	23.66
5	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08
6	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08
7	Capacity (N_k)	80.60	80.60	80.60	80.60	80.60
	Running Time (T_k)	27.08	27.08	27.08	27.08	27.08

The constraint regarding low-intensity magnetic separation $S(x)$ can be denoted as,

$$S(x) = \frac{\eta_b \sum_{i=1}^I u_i x_i}{\sum_{k=1}^K N_k T_k}, \quad (6.28)$$

where N_k and T_k denote the operating number and running time of the equipment with the series number k , respectively.

We deploy the raw ore allocation to five environments with different operational conditions and simultaneously optimize the five tasks. The environmental parameters N_k and T_k for the five operational conditions are enumerated in Table 6.8. The raw ore properties, such as α_i , $\beta_{1,i}$, $\beta_{2,i}$, u_i , $k_{1,i}$, and $k_{2,i}$ are listed in Table Table 5.6. η_b and S_{upper} are set to 82% and 80, respectively. To handle the constraint of the problem, we simply regard the constraint violation value as an additional objective [63].

In this application, five tasks (ROA1-ROA5) need to be optimized simultaneously. For each task, we treat this task as the target task and the other four tasks as the source tasks, and

Table 6.9: Mean HV values and standard deviation obtained by different algorithms on the ROA problems.

Problem	MFEA	EMT-EA	GMFEA	AT-MFEA	MFEA-KAE	EMT-GFK	LCB-EMT
ROA1	1.10e-01±2.02e-02(+)	1.28e-01±1.94e-02(+)	1.25e-01±5.15e-02(+)	1.33e-01±3.30e-02(+)	1.29e-01±2.23e-02(+)	1.30e-01±1.72e-02(+)	1.42e-01±1.92e-02
ROA2	1.08e-01±3.10e-02(+)	1.20e-01±2.50e-02(+)	1.17e-01±6.43e-02(+)	1.30e-01±2.07e-02(+)	1.23e-01±2.91e-02(+)	1.19e-01±2.55e-02(+)	1.39e-01±5.03e-02
ROA3	1.11e-01±1.38e-02(+)	1.28e-01±3.32e-02(+)	1.17e-01±5.09e-02(+)	1.33e-01±1.19e-02(=)	1.27e-01±3.06e-02(+)	1.17e-01±1.90e-02(+)	1.40e-01±1.53e-02
ROA4	1.07e-01±2.45e-02(+)	1.21e-01±3.19e-02(+)	1.21e-01±3.88e-02(+)	1.34e-01±3.74e-02(=)	1.24e-01±2.05e-02(+)	1.30e-01±2.28e-02(=)	1.36e-02±1.38e-02
ROA5	1.12e-01±4.11e-02(+)	1.30e-01±4.50e-02(+)	1.24e-01±2.40e-02(+)	1.33e-01±2.58e-02(+)	1.30e-01±3.96e-02(+)	1.33e-01±2.81e-02(+)	1.42e-01±2.04e-02
+/-/-	5/0/0	5/0/0	5/0/0	3/2/0	5/0/0	4/1/0	

thus we can have four pairs of target-source relationships. Since our TGP model only works on one target task and one source task, when knowledge transfer is triggered, we perform the knowledge transfer on each pair of target-source relationships. For each pair of target-source relationships, data from this pair of the target and source tasks are used to train a TGP model for solution transfer. Based on the transferred solutions from all pairs of relationships and the original solutions from the target task, parent solutions are selected by the selection operator for the next generation of the target task.

To endow LCB-EMT with the ability to deal with multiobjective optimization problems, we extend the Eq. (6.16) to the summation of mean prediction value and the uncertainty over multiple objective functions,

$$SLCB(x) = \sum_{m=1}^M f_m(x) - \lambda_{gen} \sigma_m(x), \quad (6.29)$$

where $f_m(x)$ and $\sigma_m(x)$ denote the prediction objective value and the prediction standard variance of the solution x on the m -th objective function.

We set the population size and the maximum number of evaluations to 100 and 10,000, respectively. We use the Hyper Volume (HV) metric to evaluate the quality of solutions obtained by our algorithm. We trigger knowledge transfer after a fixed interval generation $TG = 15$. The statistical results of HV for five different operational conditions (ROA1-ROA5) are reported in Table 6.9. The results demonstrate that the proposed LCB-EMT algorithm outperformed

other state-of-the-art methods under all operational conditions, as measured by HV values. This demonstrates the effectiveness and superiority of our approach in solving the raw ore allocation problem under various operational conditions.

6.5 Conclusion

This work proposed a solution selection approach to select high-quality solutions for positive transfer in EMT. Unlike existing approaches that often overlook leveraging information from the target task in selecting solutions, the proposed LCB metric makes use of the solution distribution of the target task to transfer solutions with higher potential improvement on the target task, thereby boosting positive knowledge transfer. The performance of the proposed approach has been extensively evaluated through comprehensive experimental studies on two single-objective multitasking benchmarks, one multiobjective multitasking benchmark, and one real-world multiobjective multitasking application. The results have demonstrated the effectiveness and generality of our proposed solution selection approach. There are several possible avenues for future research in the area of EMT. One direction is to extend the proposed solution selection approach to discrete spaces by transforming the discrete representations into continuous representations or exploring machine learning methods suitable for discrete representations. In addition, a wider range of machine learning techniques for measuring inter-task similarity and evaluating the quality of solutions can be explored in the future, as these metrics play a critical role in effective knowledge transfer. By leveraging advances in machine learning, we believe the capabilities of multitasking optimization algorithms on a broader range of real-world problems can be enhanced.

Chapter 7

Knowledge-Driven Multiobjective Combinatorial Optimization for Explainable Fair Graph Neural Networks

7.1 Motivation

To facilitate algorithmic fairness of machine learning in high-stakes applications, fair learning algorithms have recently attracted significant attention [195, 196, 197]. Recently, several studies have been proposed towards fair GNNs due to their superior predictive performance on high-stakes applications [198, 199, 200, 103]. On the other hand, *explainability*, a well-known principle to gain insights into the models, could help system developers and machine learning practitioners to debug the model to ensure algorithmic fairness by identifying data biases in training data [201, 202, 203, 204]. For example, Gopher [205] identifies the root of bias in the

context of tabular data by identifying samples of biased patterns. BIND [206] quantifies the node’s influences on the unfairness of GNNs based on the probabilistic distribution disparity. REFEREE [207] attempts to provide instance-level explanations for the bias of a certain node.

A primary issue of fair machine learning is that biases are usually introduced during the data collection. In particular, the topological structure of graph data might magnify the bias through the message-passing mechanisms [200, 208]. However, existing explainable works are not specifically for model-level fairness diagnosis of GNNs to identify such topological structures and could not answer the following questions: *What subgraph patterns of the graph cause the biased behavior of GNNs, and what actions could machine learning practitioners take toward a fairer GNN?* To complement prior explainable approaches, this paper attempts to produce compact, diagnostic, and actionable subgraph patterns that are responsible for the model’s biased behavior. Taking crime forecasting systems as an example (see Fig. 7.1 (a)), individuals within the same race (the yellow and blue nodes represent two races, respectively) are often connected in dense communities, where edges denote social connections. Particularly, such dense connection patterns may exacerbate algorithmic bias toward one specific race during the message-passing in GNNs [200, 208]. Therefore, diagnostic explanations are desirable to track certain connection patterns to suppress the exacerbation of algorithmic bias. Once the root of bias is identified, actionable explanations are provided for modifying the connection patterns, thus reducing the data bias and rectifying biased behaviors.

Intuitively, identifying a subset of edges and nodes from the entire graph to constitute subgraph patterns as explanations can be formulated as a combinatorial optimization problem. Nevertheless, it is challenging to solve the problem for the following three reasons: **1) Large-scale search space:** The combinations of edges and nodes for an explanation are exponential, and the explicit expressions of the gradients are difficult to obtain. It is, therefore, non-trivial to search from a large-scale space without the gradient information. **2) Multiobjective optimization:** Three objectives need to be optimized, including *Compactness*: the explanations should be

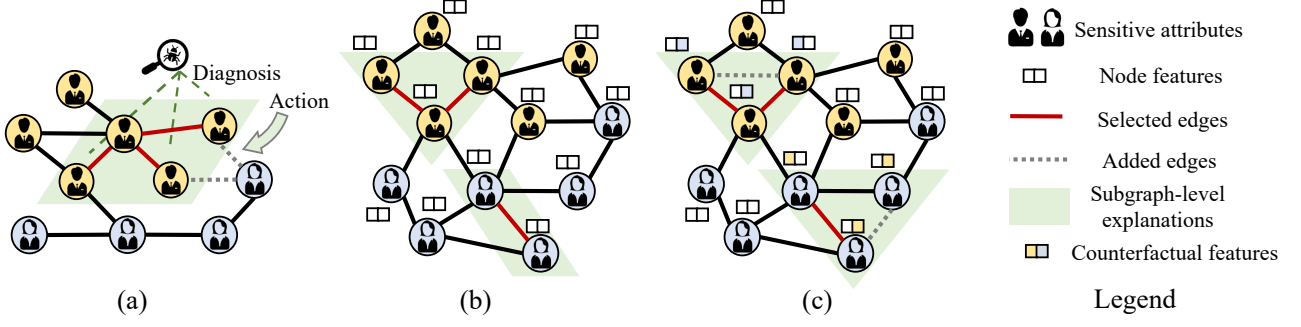


Figure 7.1: (a) An example of diagnostic explanations and actionable explanations on graphs. (b) Biased subgraphs as diagnostic explanations are identified by selecting a subset of edges and nodes to locate the root cause of the biased behavior of a GNN model. (c) By repairing the biased subgraphs (both features and structures), counterfactual debiased subgraphs as actionable explanations are generated to help the system developers and machine learning practitioners to reduce the data bias, thus rectifying discriminatory behavior.

compact enough so that humans can easily interpret them and take further actions [209]; *Bias attribution*: The generated explanations should be able to capture the root of bias or rectify biased behaviors at large; *Accuracy*: Taking the generated explanations to modify the graph for fairness should not impair the propagation of useful information and thus does not affect model predictive performance. **3) Intensive computation**: Evaluating the three objective functions on the modified graph requires GNN retraining, and repetitively retraining the GNN towards a fair one is computation-intensive.

Inspired by the distinctive competency of EAs for large-scale combinatorial optimization problems [210, 109, 5], we develop a dedicated multiobjective EA-based framework for generating diagnostic and actionable explanations. The generated explanations can satisfy the following three objectives simultaneously: *compactness*, *bias attribution*, and *accuracy*. Moreover, we propose an influenced nodes-based gradient approximation to address the problem of expensive evaluation costs for updating the GNN model. Our proposed approximation algorithm could avoid retraining the GNN from scratch when evaluating the GNN on the modified graph. Ex-

tensive experiments on three benchmarking datasets are conducted to evaluate the generated diagnostic and actionable explanations. The experimental results demonstrate that the generated explanations can identify the root of bias and rectify biased behaviors. Specifically, we empirically show that the generated explanations can alleviate bias and reduce the statistical parity by 3% $\sim$ 9% compared to vanilla GNNs while achieving comparable classification performance. In addition, we provide the theoretical fairness risk bound of the proposed method, indicating that using actionable explanations to rectify the root of bias can theoretically improve fairness.

7.2 Preliminaries

In this section, we present the concepts of fair node classification and the related notions about diagnostic and actionable explanations.

Notations: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ denotes an attributed graph with a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$, where $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{V}|})^\top \in \mathbb{R}^{|\mathcal{V}| \times d}$ is a set of node features. Each node i might have a sensitive attribute $s_i \in \{0, 1\}$. Taking binary classification as an example, fair node classification aims to ensure that the classifier $f : \mathbf{x} \rightarrow y$ makes fair predictions devoid of discrimination with respect to the sensitive attribute s , where $y \in \{0, 1\}$ denotes the node label.

Statistical parity and equal opportunity [211, 212] are two widely used definitions for fairness.

Definition 10. *Statistical parity (SP) [211] requires the classifier f to make identical prediction $\hat{y} \in \{0, 1\}$ while having different values of sensitive attribute s , $\Pr(\hat{y}|s = 1) = \Pr(\hat{y}|s = 0)$.*

Definition 11. *Equal opportunity (EO) [212] requires the predictions of both groups with different values of the attribute to have the same true positive rate, $\Pr(\hat{y}|y = 1, s = 1) = \Pr(\hat{y}|y = 1, s = 0)$.*

Based on SP and EO, the fairness metrics can be defined as [213, 214], $\Delta_{SP} = |\Pr(\hat{y}|s = 1) - \Pr(\hat{y}|s = 0)|$, $\Delta_{EO} = |\Pr(\hat{y}|y = 1, s = 1) - \Pr(\hat{y}|y = 1, s = 0)|$. The smaller the value of Δ_{SP} and Δ_{EO} , the fairer the model is.

Problem definition: We introduce a new formulation of explanation generation for fairness by formulating the problem as a multiobjective combinatorial optimization problem,

$$\langle \min \mathcal{C}, \max \mathcal{B}, \max \mathcal{F} \rangle, \quad (7.1)$$

where three objective functions (*i.e.*, *compactness* $\mathcal{C}$, *bias attribution* $\mathcal{B}$, and *accuracy* $\mathcal{F}$) are simultaneously optimized to generate subgraph patterns. Specifically, we aim to search biased subgraphs as diagnostic explanations $\mathcal{G}' = \{\mathcal{G}'_i\}$ for locating the bias of graph data (see Fig. 7.1 (b)). Furthermore, by modifying the biased subgraphs, we generate counterfactual debiased subgraphs as actionable explanations $\mathcal{G}'' = \{\mathcal{G}''_i\}$ to help the system developers to mitigate the bias (see Fig. 7.1 (c)). Formally, we introduce the definitions of *compactness* and *bias attribution* of the generated subgraph patterns as shown below.

Definition 12. (*Compactness*) Given a subgraph $\mathcal{G}_i^{(\cdot)}$ ($\mathcal{G}_i^{(\cdot)}$ denotes $\mathcal{G}'_i$ or $\mathcal{G}''_i$) with the node set $\mathcal{V}_i^{(\cdot)}$ and the edge set $\mathcal{E}_i^{(\cdot)}$, the compactness of the subgraph $\mathcal{G}_i^{(\cdot)}$ can be defined as,

$$\mathcal{C}(\mathcal{G}_i^{(\cdot)}) = \frac{|\mathcal{E}_i^{(\cdot)}|}{|\mathcal{V}_i^{(\cdot)}|}. \quad (7.2)$$

The fewer edges included in the explanation, the easier it is for humans to interpret the subgraph [209].

Biased subgraph $\mathcal{G}'_i$ is expected to be removed to suppress the propagation of biased information. Here, we use *bias attribution* to measure the improvement of fairness when subgraph $\mathcal{G}'_i$ is removed from the raw graph.

Definition 13. (*Bias attribution of a biased subgraph*) Given a subgraph $\mathcal{G}'_i$, the bias attribution of $\mathcal{G}'_i$ can be defined as,

$$\mathcal{B}(\mathcal{G}'_i) = \mathcal{M}(\mathcal{G}; \theta) - \mathcal{M}(\mathcal{G} \setminus \mathcal{G}'_i; \theta'), \quad (7.3)$$

where $\mathcal{M}(\mathcal{G} \setminus \mathcal{G}'_i; \theta')$ means the fairness value (*e.g.*, Δ_{SP}) of the classifier $f_{\theta'}$ trained on the modified graph $\mathcal{G} \setminus \mathcal{G}'_i$. The larger the value of $\mathcal{B}(\mathcal{G}'_i)$, the greater the $\mathcal{G}'_i$ toward bias.

By repairing the biased subgraph $\mathcal{G}'_i$, counterfactual debiased subgraph $\mathcal{G}''_i$ can be generated as actionable explanations. We extend the concept of bias attribution to the counterfactual debiased subgraph $\mathcal{G}''_i$.

Definition 14. (*Bias attribution of a counterfactual debiased subgraph*) Given a counterfactual debiased subgraph $\mathcal{G}''_i$, the bias attribution of $\mathcal{G}''_i$ can be defined as,

$$\mathcal{B}(\mathcal{G}''_i) = \mathcal{M}(\mathcal{G}; \theta) - \mathcal{M}((\mathcal{G} \setminus \mathcal{G}'_i) \cup \mathcal{G}''_i; \theta''), \quad (7.4)$$

where θ'' is the classifier trained on modified graph $(\mathcal{G} \setminus \mathcal{G}'_i) \cup \mathcal{G}''_i$. In this work, both node features and structure can be revised in $\mathcal{G}'_i$ to generate $\mathcal{G}''_i$. The larger the value of $\mathcal{B}(\mathcal{G}''_i)$, the greater ability of $\mathcal{G}''_i$ to reduce bias. We expect the counterfactual debiased subgraph should be as compact as possible so that the generated subgraphs are actionable for system developers to rectify the biased behaviors.

On top of bias attribution, we define the model-level (un)fairness explanations to guide the fairness debugging for system developers.

Definition 15. (*Model-level explanations for (un)fairness*) Let f_{θ} denote a GNN classification model, our goal is to investigate what subgraph patterns $\mathcal{G}'$ from the raw graph $\mathcal{G}$ account for the unfairness of the GNN model at large and what counterfactual subgraph patterns $\mathcal{G}''$ can improve the fairness of the model. The obtained subgraph patterns $\mathcal{G}'$ and $\mathcal{G}''$ can be respectively treated as the attribution of the unfairness and fairness of the model,

$$\begin{aligned} \mathcal{G}' &= \arg \max_{\{\mathcal{G}'_i\}} \mathcal{B}(\mathcal{G}'_i), \text{ s.t. } \mathcal{B}(\mathcal{G}'_i) > 0 \\ \mathcal{G}'' &= \arg \max_{\{\mathcal{G}''_i\}} \mathcal{B}(\mathcal{G}''_i), \text{ s.t. } \mathcal{B}(\mathcal{G}''_i) > 0. \end{aligned} \quad (7.5)$$

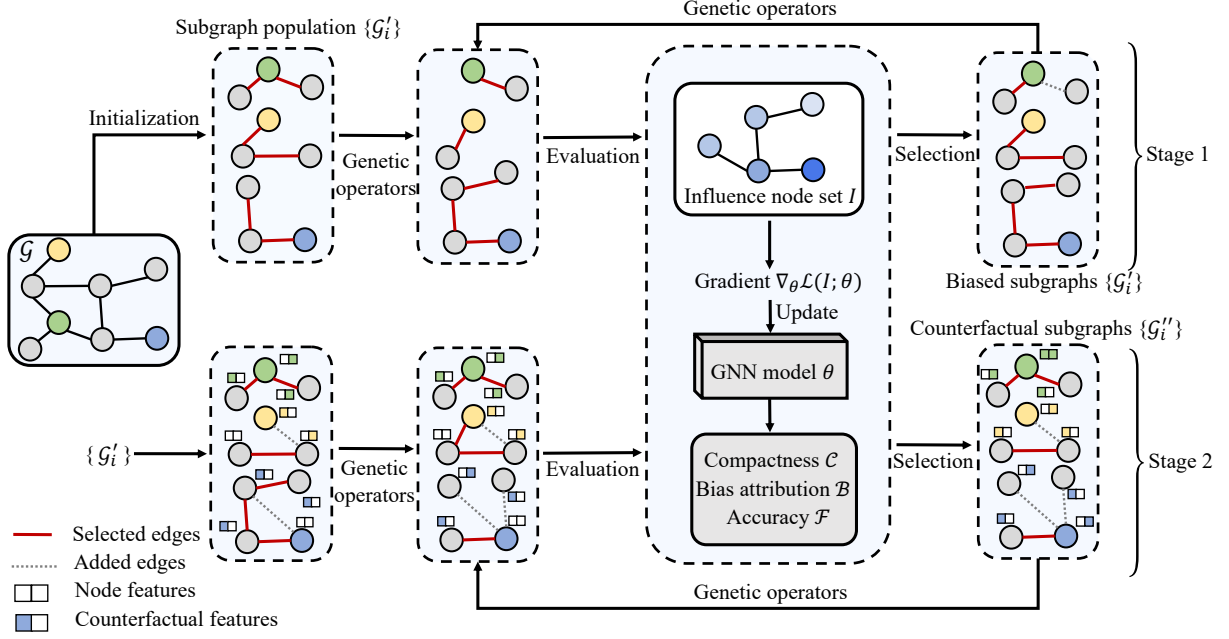


Figure 7.2: Two stages in the proposed *Geb*: The top branch (stage one) aims to identify biased subgraphs as diagnostic explanations for locating the root of data bias. The bottom branch (stage two) aims to generate counterfactual debiased subgraphs as actionable explanations for mitigating the bias.

We formulate the model-level explanation generation problem as Eq.(7.1) to maintain the explainability, fairness, and classification performance simultaneously.

7.3 Proposed Framework

The proposed framework for *Generating explanations for bias (Geb)* is illustrated in Fig. 7.2. The framework includes two stages: identification of biased subgraphs for locating the bias patterns (top branch), and generation of counterfactual debiased subgraphs for mitigating the bias (bottom branch).

7.3.1 Identification of Biased Subgraphs

Due to the remarkable competency of EAs in addressing combinatorial optimization problems and other complex optimization problems [42, 53, 215, 35, 8, 105], this work developed a multiobjective EA to solve the explanation generation problem. EAs search from a set of initial solutions through genetic operators to create new sets of better solutions to search toward the optima. Firstly, a set of subgraphs is randomly sampled from the graph data as the initial population. By performing the genetic operators, *i.e.*, crossover and mutation, to the population, a set of newly generated subgraphs is produced. Then, three objective functions, *i.e.*, the compactness and bias attribution of each subgraph as well as the model accuracy, are evaluated. After that, the subgraphs with better objective values are selected as the parent population for the next generation. The above processes are repeatedly conducted until the terminal condition (*e.g.*, the maximum number of generations) is met, and the subgraphs of the last generation are kept as the diagnostic explanations. The major components of our multiobjective EA are elaborated below:

Population: The population refers to a set of candidate subgraphs $\mathcal{G}' = \{\mathcal{G}'_i\}$, where $i = 1, \dots, I$, and I is the population size. In this work, we randomly sample I nodes and their 1-hop neighbors as the initial subgraphs.

Crossover: Crossover operators play a critical role in generating new solutions with better convergence by mixing the attributes of two solutions, which enables the newly produced solutions to have beneficial combinations of attributes of their parent. In this work, two candidate subgraphs are randomly selected as the parent subgraphs from the population pool to perform the crossover operator so that an offspring subgraph $\mathcal{G}'_{of}$ can be generated. The crossover operator mixes the edges of the two selected subgraphs $\mathcal{G}'_1 = \{\mathcal{V}'_1, \mathcal{E}'_1\}$ and $\mathcal{G}'_2 = \{\mathcal{V}'_2, \mathcal{E}'_2\}$,

$$\mathcal{G}'_{of} = (\mathcal{V}'_{of}, (\mathcal{E}'_1 \cap \mathcal{E}'_2) \cup \text{Pr}(\mathcal{E}'_1 \setminus \mathcal{E}'_2) \cup \text{Pr}(\mathcal{E}'_2 \setminus \mathcal{E}'_1)), \quad (7.6)$$

where $\mathcal{V}'_{of}$ is the set of nodes that are connected by the edges in $\mathcal{G}'_{of}$, and $\text{Pr}(\cdot)$ randomly outputs

a subset of edges from the candidate edges. As such, the common edges in both parents that may amplify bias will be kept in offspring, and potentially biased edges of one parent are randomly kept.

Mutation: By modifying the partial attributes in solutions, mutation operators introduce randomness and variability into the population during the evolution process, which can enhance the diversity of the population and help the population to escape from the local optima. In our design, for each candidate subgraph, we randomly add one edge that connects the subgraph to its adjacent nodes or randomly drop one edge from the subgraph to generate a new subgraph.

Objective Function Evaluation: Each subgraph in the population pool will get objective values that measure the quality of the generated biased subgraph. Specifically, the compactness $\mathcal{C}(\mathcal{G}'_i)$ and the bias attribution $\mathcal{B}(\mathcal{G}'_i)$ of a subgraph $\mathcal{G}'_i$ are evaluated by Eq. (7.2) and Eq. (7.3), respectively. The model accuracy $\mathcal{F}$ is the AUC score obtained by the GNN trained on the modified graph $\mathcal{G} \setminus \mathcal{G}'_i$.

Selection: Given the objective values of each subgraph, we use the non-dominated sort to select the non-dominated solutions as the parent population for the next generation [216].

7.3.2 Influenced Nodes-Based Gradient Approximation

It is extremely computationally expensive to retrain the GNN model to evaluate the objective functions on the modified graph. To address the computational challenge, we assume that modifying the local structure of a graph may not have a great impact on the distant nodes, and the GNN model can be simply updated on the influenced nodes affected by $\mathcal{G}'$. The rationality comes from the widely used local-dependence assumption for graph data [217]. Based on this assumption, we proposed an influenced nodes-based gradient approximation to update the GNN model with a lower computational cost. First, we give the definition of the influenced nodes,

Definition 16. (*Influenced nodes*) Assume the change of node u 's embedding is $\Delta \mathbf{h}_u$ after

modifying the graph data, the set of influenced nodes can be defined as,

$$\mathcal{I} = \{u | \arg \max_i \sigma(\mathbf{h}_u)[i] \neq \arg \max_i \sigma(\mathbf{h}_u + \Delta \mathbf{h}_u)[i]\}, \quad (7.7)$$

where $u \in \Omega_l(\mathcal{G}')$, $\Omega_l(\mathcal{G}')$ is the node set of l -hop neighbors of all nodes in $\mathcal{G}'$, l is the layer number of the studied GNN, σ is the softmax function, $\mathbf{h}_u$ is the embedding of the node u before modifying the graph, and $\sigma(\mathbf{h}_u)[i]$ is the predicted probability of the node u belonging to the class i . The definition indicates that those nodes from $\Omega_l(\mathcal{G}')$ whose classification results are different after modifying the graph are the influenced nodes. We update the GNN model on those influenced nodes instead of the whole graph to save computational costs.

Another issue is estimating the change of node u 's embedding, $\Delta \mathbf{h}_u$. We design a fast approximation algorithm to estimate the change of a node's embedding. For each node $u \in \Omega_l(\mathcal{G}')$, the change $\Delta \mathbf{h}_u$ depends on the l -hop neighbors $\Omega_l(u)$ and the l -hop removed neighbors $\bar{\Omega}_l(u)$.

$$\Delta \mathbf{h}_u = \sum_{v \in \Omega_l(u)} s_v^l \mathbf{h}_v - \sum_{v \in \bar{\Omega}_l(u)} s_v^l \mathbf{h}_v. \quad (7.8)$$

For each node u , s_u^l can be calculated as,

$$s_u^l = \frac{\sum_{v \in (\Omega_l(u) \cup \bar{\Omega}_l(u) \cup u)} s_v^{l-1}}{d_u + 1}, s_u^0 = 1, s_v^0 = 0, v \neq u, \quad (7.9)$$

where d_u is the degree of the node u . The intuition behind the influence scores is that the greater the distance between a node v to u , the smaller the influence score s_v of $\mathbf{h}_v$, and the less information is propagated from $\mathbf{h}_v$ for changing $\mathbf{h}_u$.

For a better understanding, we provide an illustrative example to demonstrate how to calculate the influence score s_v of one neighbor node $v \in \Omega_l(u)$ and how to use the influence scores to calculate $\Delta \mathbf{h}_u$. As shown in Fig. 7.3 (a), given a node u and the layer number $l = 2$ of the studied GNN, a raw subgraph is sampled from a graph. After performing mutation and crossover, a new subgraph is obtained in Fig. 7.3 (b). Because v_1 is removed from l -hop neighbors of u and v_2, v_3, v_4 is the l -hop neighbors of u , we get $\bar{\Omega}_l(u) = \{v_1\}$ and $\Omega_l(u) =$

$\{v_2, v_3, v_4\}$. Next, we calculate the influence scores from $l = 0$ to $l = 2$. Now, we start with $l = 0$. According to Eq. (7.9), the influence scores are $s_u^0 = 1$ and $s_{v_1}^0 = s_{v_2}^0 = s_{v_3}^0 = s_{v_4}^0 = 0$. We marked the influence scores below the nodes. Then, according to Eq. (7.9) again, the influence scores of $l = 1$ are shown in Fig. 7.3 (c). Continually, we get the influence scores of $l = 2$ as shown in Fig. 7.3 (d). Finally, with the influence scores in Fig. 7.3 (d), $\Delta \mathbf{h}_u$ can be calculated by Eq. (7.8).

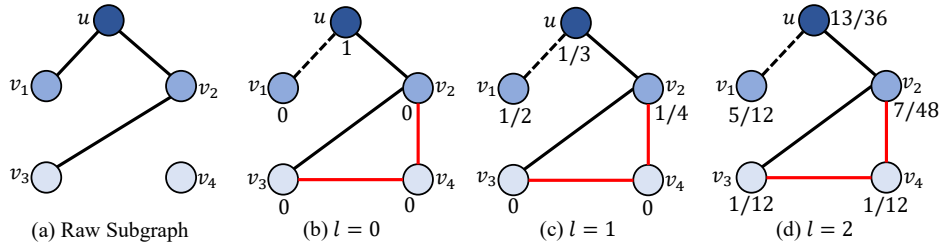


Figure 7.3: The illustrative examples of calculating the influence scores. The red line represents the newly selected edge in the subgraph, while the dashed line represents the edge that is removed from the subgraph. The number below the node denotes the influence score.

After estimating $\Delta \mathbf{h}$, the influenced nodes $\mathcal{I}$ can be identified by Eq. (7.7). Assuming that model parameters do not change significantly when a small subset of data points is changed, the model parameters can get a minimal change [218], thus the parameters of the GNN model can be updated on the set of influenced nodes by a single step of gradient descent,

$$\theta' = \theta - \eta \left(\nabla_{\theta} \mathcal{L}(\mathcal{G}, \theta) - \frac{1}{|\mathcal{I}|} \nabla_{\theta} \mathcal{L}(\mathcal{I}, \theta) \right), \quad (7.10)$$

where η is the learning rate for the gradient step and θ is the model trained on the raw graph $\mathcal{G}$. In this way, based on the trained model θ , we update the GNN to evaluate the objective functions instead of retraining the GNN from scratch.

7.3.3 Generation of Counterfactual Debaised Subgraphs

After identifying the biased subgraphs, we further seek to provide counterfactual debaised subgraphs as actionable explanations for system developers to repair such roots of bias. Taking the biased subgraphs $\mathcal{G}'$ as the initial subgraphs, the counterfactual debaised subgraphs $\mathcal{G}''$ are also generated by the proposed multiobjective EA. Due to the counterfactual nature, different from stage one, the mutation operator in stage two can add previously non-existent edges in the raw graph to generate counterfactual structures. The model accuracy is the AUC score obtained by the classifier trained on $(\mathcal{G} \setminus \mathcal{G}'_i) \cup \mathcal{G}''_i$. Other components in the multiobjective EA are the same as those of stage one.

In addition to generating counterfactual structures, we also seek to generate counterfactual features. Given a node $v \in \mathcal{G}''_i$, the counterfactual features $\mathbf{x}_v^c$ can be obtained by minimizing the fairness constraint $\mathcal{L}_{cov}$ with the gradient descent algorithm,

$$\mathbf{x}_v^c = \Pi(\mathbf{x}_v - \eta \nabla_{\mathbf{x}_v^c} \mathcal{L}_{cov}), \quad (7.11)$$

where Π projects the generated features back to the feasible domain, and $\mathcal{L}_{cov}$ is the absolute covariance between the sensitive attribute s and the prediction $\hat{y}$ [219],

$$\mathcal{L}_{cov} = |\mathbb{E}[(s - \mathbb{E}(s))(\hat{y} - \mathbb{E}(\hat{y}))]|. \quad (7.12)$$

The computation of the gradient $\nabla_{\mathbf{x}_v^c} \mathcal{L}_{cov}$ is complicated. Here, we derive the fairness constraint with respect to $\mathbf{x}_v^c$ via the chain rule,

$$\frac{\partial \mathcal{L}_{cov}}{\partial \mathbf{x}_v^c} = \frac{\partial \mathcal{L}_{cov}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{x}_v^c}, \quad (7.13)$$

where $\frac{\partial \mathcal{L}_{cov}}{\partial \theta}$ is the gradient w.r.t $\mathcal{L}_{cov}$, and we denote it as $\nabla_{\theta} \mathcal{L}_{cov}$. Then, $\frac{\partial \theta}{\partial \mathbf{x}_v^c}$ can be obtained by [220],

$$\frac{\partial \theta}{\partial \mathbf{x}_v^c} = -H_{\theta}^{-1} \frac{\partial^2 \mathcal{L}_{cov}}{\partial \theta \partial \mathbf{x}_v^c}, \quad (7.14)$$

where H_θ is the Hessian of the loss. By combining Eq. (7.13) and Eq. (7.14), the gradient of the loss w.r.t. the generated counterfactual features $\mathbf{x}_v^c$ is,

$$\frac{\partial \mathcal{L}_{cov}}{\partial \mathbf{x}_v^c} = -(\nabla_\theta \mathcal{L}_{cov})^\top H_\theta^{-1} \frac{\partial^2 \mathcal{L}_{cov}}{\partial \theta \partial \mathbf{x}_v^c}. \quad (7.15)$$

After generating offspring subgraphs with counterfactual structures and features, the evolutionary process is iteratively performed until the terminal condition is satisfied. The subgraphs of the last generation are served as the actionable explanations for diluting the bias.

7.4 Theoretical Analysis

In this section, we provide the fairness risk bound to illustrate the effectiveness of the generated counterfactual debiased subgraphs in improving fairness. Firstly, The expectation fairness risk Δ_{SP} and empirical fairness risk $\tilde{\Delta}_{SP}$ of a classifier f are introduced,

$$\begin{aligned} \Delta_{SP}(f, \mathcal{G}) &= \\ &|\mathbb{E}_{S=s^+}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1] - \mathbb{E}_{S=s^-}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1]|, \\ \tilde{\Delta}_{SP}(f, \mathcal{G}) &= \\ &|\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1) - \mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1)|, \end{aligned} \quad (7.16)$$

where $\mathcal{G}$ is a graph coming from the domain $\mathcal{Z}$, and $\mathbb{I}$ is an indicator function.

Theorem 2. Assume that $\mathcal{F}$ is a function class consisting of functions with range $[a, b]$, for any classifier $f \in \mathcal{F}$ with a graph $\mathcal{G}$, if the difference between the empirical fairness risk $\tilde{\Delta}_{SP}$ and the expectation fairness risk Δ_{SP} can be bounded by $\tau_{\Delta_{SP}}$. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the expectation fairness risk bound Δ_{SP} is given by,

$$\begin{aligned} \Delta_{SP}(f, \mathcal{G}) &\leq \tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)}) + D_{\mathcal{F}}(\mathcal{Z}^{(ub)}, \mathcal{Z}^{(bi)}) \\ &\quad + \tau_{\Delta_{SP}} + \sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}}, \end{aligned} \quad (7.17)$$

where $\mathcal{G}$ is the raw graph from the domain $\mathcal{Z}^{(bi)}$ with inherent data bias, $\mathcal{G}^{(ce)}$ is the debiasing graph, $\mathcal{Z}^{(ub)}$ is the domain of ideal debiased data, and n is the number of nodes.

It should be noted that the model f in $\Delta_{SP}(f, \mathcal{G})$ and $\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)})$ are not exactly the same, as the parameters of f have been updated on $\mathcal{G}^{(ce)}$. While according to Eq. (7.10), they can be approximately equivalent. The theoretical analysis shows that the expectation fairness risk $\Delta_{SP}(f, \mathcal{G})$ is bounded by the empirical fairness risk $\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)})$, the difference $D_{\mathcal{F}}(\mathcal{Z}^{(ub)}, \mathcal{Z}^{(bi)})$ between the two domains $\mathcal{Z}^{(ub)}$ and $\mathcal{Z}^{(bi)}$, and two additional terms $\sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}}$ and $\tau_{\Delta_{SP}}$, suggesting that generating counterfactual debiased subgraphs and reducing the inherent bias degree of the collected data (i.e., $D_{\mathcal{F}}(\mathcal{Z}^{(ub)}, \mathcal{Z}^{(bi)})$) can reduce the upper bound of the fairness risk. This work uses the generated counterfactual debiased subgraphs to produce a debiasing graph $\mathcal{G}^{(ce)}$ with a fair classifier f to minimize the empirical fairness risk $\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)})$, thus bounding the expectation fairness risk $\Delta_{SP}(f, \mathcal{G})$.

Next, we provide the fairness risk bound to illustrate the effectiveness of using counterfactual unbiased subgraphs to improve fairness. Firstly, the extended McDiamid's inequality is introduced here.

Theorem 3. (*Extended McDiamid's Inequality [221]*). Given independent domains $\mathcal{Z}^{(k)} (1 \leq k \leq K)$, for any k , let $S^{m_k} = \{z_i^{(k)}\}_{i=1}^{m_k}$ be m_k independent random variables taking values from the domain $\mathcal{Z}^{(k)}$. Assume that the function $H : (\mathcal{Z}^{(1)})^{m_1} \times \dots \times (\mathcal{Z}^{(K)})^{m_K} \rightarrow \mathbb{R}$ satisfies the condition of bounded difference: for all $1 \leq k \leq K$ and $1 \leq i \leq m_k$,

$$\sup_{S^{m_1}, \dots, S^{m_K}, z_i^{(k)}} |\mathcal{H} - \mathcal{H}'| \leq c_i^{(k)}, \quad (7.18)$$

where

$$\begin{aligned} \mathcal{H} &= H(S^{m_1}, \dots, S^{m_{k-1}}, \\ &\quad z_1^{(k)}, \dots, z_i^{(k)}, \dots, z_{m_k}^{(k)}, \dots, S^{m_{k+1}}, \dots, S^{m_K}), \\ \mathcal{H}' &= H(S^{m_1}, \dots, S^{m_{k-1}}, \\ &\quad z_1^{(k)}, \dots, z_i'^{(k)}, \dots, z_{m_k}^{(k)}, \dots, S^{m_{k+1}}, \dots, S^{m_K}), \end{aligned} \quad (7.19)$$

Then, for any $\xi > 0$,

$$\begin{aligned} & \Pr\{H(S^{m_1}, \dots, S^{m_K}) - \mathbb{E}(H(S^{m_1}, \dots, S^{m_K})) \geq \xi\} \\ & \leq \exp\left(-2\xi^2 / \sum_{k=1}^K \sum_{i=1}^{m_k} (c_i^{(k)})^2\right). \end{aligned} \quad (7.20)$$

For the sake of simplicity, we assume that $\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1) \geq \mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1)$ and $\mathbb{E}_{S=s^+}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1] \geq \mathbb{E}_{S=s^-}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1]$. According to Theorem 3, we employ a function $\mathcal{H}$ to measure the discrepancy between the debiasing graph domain $\mathcal{Z}^{(ce)}$ and the raw graph domain $\mathcal{Z}^{(bi)}$ containing inherent bias, respectively.

$$\begin{aligned} & \mathcal{H}(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)}) \\ &= \sup_{f \in \mathcal{F}} |\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)}) - \Delta_{SP}(f, \mathcal{G})| \\ &= \sup_{f \in \mathcal{F}} |\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1) - \mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1) \\ & \quad - \mathbb{E}_{S=s^+}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1] + \mathbb{E}_{S=s^-}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1]| \\ &= \sup_{f \in \mathcal{F}} |\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1) - \mathbb{E}_{S=s^+}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1] \\ & \quad - (\mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1) - \mathbb{E}_{S=s^-}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1])|, \end{aligned} \quad (7.21)$$

where $\mathcal{G}$ is the raw graph from the domain $\mathcal{Z}^{(bi)}$, $\mathcal{G}^{(ce)}$ is the debiasing graph from the domain $\mathcal{Z}^{(ce)}$, $\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)})$ is the empirical fairness risk Δ_{SP} obtained by a classifier f with the debiasing graph, $\Delta_{SP}(f, \mathcal{G})$ is the expectation fairness risk Δ_{SP} obtained by the classifier f with the raw graph, and $\mathbb{I}$ is the indicator function. We hereafter denote $\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1)$, $\mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}^{(ce)}}(\mathbf{x}) = 1)$, $\mathbb{E}_{S=s^+}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1]$, and $\mathbb{E}_{S=s^-}[f_{\mathbf{x} \sim \mathcal{G}}(\mathbf{x}) = 1]$ as $\tilde{R}_+^{(ce)}$, $\tilde{R}_-^{(ce)}$, $R_+^{(bi)}$, and $R_-^{(bi)}$, respectively.

Given $\gamma \in [0, 1]$, we simply consider the fairness on the debiasing graph $\mathcal{G}^{(ce)}$ to be the linear combination of the fairness on the raw graph $\mathcal{G}$ and the fairness on the ideal unbiased graph

$\mathcal{G}^{(ub)} \in \mathcal{Z}^{(ub)}$ [222], thus we have,

$$\begin{aligned}\tilde{R}_+^{(ce)} &= \gamma \tilde{R}_+^{(bi)} + (1 - \gamma) \tilde{R}_+^{(ub)}, \\ \tilde{R}_-^{(ce)} &= \gamma \tilde{R}_-^{(bi)} + (1 - \gamma) \tilde{R}_-^{(ub)},\end{aligned}\tag{7.22}$$

where $\tilde{R}_+^{(ub)}$ and $\tilde{R}_-^{(ub)}$ denote $\mathbb{I}_{S=s^+}(f_{\mathbf{x} \sim \mathcal{G}^{(ub)}}(\mathbf{x}) = 1)$ and $\mathbb{I}_{S=s^-}(f_{\mathbf{x} \sim \mathcal{G}^{(ub)}}(\mathbf{x}) = 1)$, respectively. Then, Eq. (7.21) can be rewritten as,

$$\begin{aligned}& H(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)}) \\ &= \sup_{f \in \mathcal{F}} |\gamma \tilde{R}_+^{(bi)} + (1 - \gamma) \tilde{R}_+^{(ub)} - R_+^{(bi)} \\ &\quad - (\gamma \tilde{R}_-^{(bi)} + (1 - \gamma) \tilde{R}_-^{(ub)} - R_-^{(bi)})| \\ &= \sup_{f \in \mathcal{F}} |\gamma (\tilde{R}_+^{(bi)} - \tilde{R}_-^{(bi)}) - (R_+^{(bi)} - R_-^{(bi)}) \\ &\quad + (1 - \gamma) (\tilde{R}_+^{(ub)} - \tilde{R}_-^{(ub)})| \\ &= \sup_{f \in \mathcal{F}} |\gamma \tilde{\Delta}_{SP}(f, \mathcal{G}) + (1 - \gamma) \tilde{\Delta}_{SP}(f, \mathcal{G}^{(ub)}) - \Delta_{SP}(f, \mathcal{G})|.\end{aligned}\tag{7.23}$$

Then, according to Theorem 3, for any $\xi > 0$, we have,

$$\Pr\{H(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)}) - \mathbb{E}(H(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)})) \geq \xi\} \leq \exp\left(\frac{-4n\xi^2}{(b-a)^2}\right).\tag{7.24}$$

With probability at least $1 - (\delta/4)$, we have,

$$\begin{aligned}
& H(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)}) \\
& \leq \mathbb{E}[H(\mathcal{Z}^{(ce)}, \mathcal{Z}^{(bi)})] + \sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}} \\
& \leq \gamma \sup_{f \in \mathcal{F}} |\tilde{\Delta}_{SP}(f, \mathcal{G}) - \Delta_{SP}(f, \mathcal{G})| \\
& \quad + (1 - \gamma) \mathbb{E}(\sup_{f \in \mathcal{F}} |\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ub)}) \\
& \quad \quad - \Delta_{SP}(f, \mathcal{G}^{(ub)}) + \Delta_{SP}(f, \mathcal{G}^{(ub)}) - \Delta_{SP}(f, \mathcal{G})|) \\
& \quad + \sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}} \tag{7.25} \\
& \leq \gamma \sup_{f \in \mathcal{F}} |\tilde{\Delta}_{SP}(f, \mathcal{G}) - \Delta_{SP}(f, \mathcal{G})| \\
& \quad + (1 - \gamma) \mathbb{E} \left(\sup_{f \in \mathcal{F}} |\tilde{\Delta}_{SP}(f, \mathcal{G}^{(ub)}) - \Delta_{SP}(f, \mathcal{G}^{(ub)})| \right) \\
& \quad + (1 - \gamma) \sup_{f \in \mathcal{F}} |\Delta_{SP}(f, \mathcal{G}^{(ub)}) - \Delta_{SP}(f, \mathcal{G})| \\
& \quad + \sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}}.
\end{aligned}$$

For any classifier $f \in \mathcal{F}$ with the graph $\mathcal{G} \sim \mathcal{Z}$, we assume the difference between the empirical fairness risk $\tilde{\Delta}_{SP}(f, \mathcal{G})$ and the expectation fairness risk $\Delta_{SP}(f, \mathcal{G})$ can be bounded by $\tau_{\Delta_{SP}}$. Then, the summation of the first supremum term and the second supremum term in Eq. (7.25) is bounded by $\tau_{\Delta_{SP}}$. The quantity in the third supremum term in Eq. (7.25) can be termed as,

$$\sup_{f \in \mathcal{F}} |\Delta_{SP}(f, \mathcal{G}^{(ub)}) - \Delta_{SP}(f, \mathcal{G})| = D_{\mathcal{F}}(\mathcal{Z}^{(ub)}, \mathcal{Z}^{(bi)}), \tag{7.26}$$

which measures the difference between the two domains. Overall, we have with probability $1 - \delta$,

$$\begin{aligned}
\Delta_{SP}(f, \mathcal{G}) & \leq \tilde{\Delta}_{SP}(f, \mathcal{G}^{(ce)}) + D_{\mathcal{F}}(\mathcal{Z}^{(ub)}, \mathcal{Z}^{(bi)}) \\
& \quad + \tau_{\Delta_{SP}} + \sqrt{\frac{(b-a)^2 \ln(4/\delta)}{4n}}, \tag{7.27}
\end{aligned}$$

This completes the proof.

Table 7.1: The comparisons of our proposed *Geb* with the baselines (%).

Method	AUC ($\uparrow$)			Δ_{SP} ($\downarrow$)			Δ_{EO} ($\downarrow$)		
	NBA	Pokec-z	Pokec-n	NBA	Pokec-z	Pokec-n	NBA	Pokec-z	Pokec-n
GCN	78.3$\pm$0.3	77.2$\pm$0.1	75.1$\pm$0.2	7.9 $\pm$ 1.3	9.9 $\pm$ 1.1	9.6 $\pm$ 0.9	17.8 $\pm$ 2.6	9.1 $\pm$ 0.6	12.8 $\pm$ 1.3
GAT	78.2 $\pm$ 0.6	76.7 $\pm$ 0.1	75.1 $\pm$ 0.2	10.2 $\pm$ 2.5	9.1 $\pm$ 0.9	9.4 $\pm$ 0.7	15.9 $\pm$ 4.0	8.4 $\pm$ 0.6	12.0 $\pm$ 1.5
ALFR	71.5 $\pm$ 0.3	71.3 $\pm$ 0.3	67.7 $\pm$ 0.5	2.3 $\pm$ 0.9	2.8 $\pm$ 0.5	3.1 $\pm$ 0.5	3.2 $\pm$ 1.5	1.1 $\pm$ 0.4	3.9 $\pm$ 0.6
ALFR-e	72.9 $\pm$ 1.0	74.0 $\pm$ 0.7	71.9 $\pm$ 0.3	4.7 $\pm$ 1.8	5.8 $\pm$ 0.4	4.1 $\pm$ 0.5	4.7 $\pm$ 1.7	2.8 $\pm$ 0.8	4.6 $\pm$ 1.6
Debias	71.3 $\pm$ 0.7	71.4 $\pm$ 0.6	67.9 $\pm$ 0.7	2.5 $\pm$ 1.5	1.9 $\pm$ 0.6	2.4 $\pm$ 0.7	3.1 $\pm$ 1.9	1.9 $\pm$ 0.4	2.6 $\pm$ 1.0
Debias-e	72.9 $\pm$ 1.2	74.2 $\pm$ 0.7	71.7 $\pm$ 0.7	5.3 $\pm$ 0.9	4.7 $\pm$ 1.0	3.6 $\pm$ 0.2	3.1 $\pm$ 1.3	3.0 $\pm$ 1.4	4.4 $\pm$ 1.2
FCGE	73.6 $\pm$ 1.5	71.0 $\pm$ 0.2	69.5 $\pm$ 0.4	2.9 $\pm$ 1.0	3.1 $\pm$ 0.5	4.1 $\pm$ 0.8	3.0 $\pm$ 1.2	1.7 $\pm$ 0.6	5.5 $\pm$ 0.9
FairGCN	77.0 $\pm$ 0.3	76.7 $\pm$ 0.2	74.9 $\pm$ 0.4	1.0 $\pm$ 0.5	0.9 $\pm$ 0.5	0.8 $\pm$ 0.2	1.2 $\pm$ 0.4	1.7 $\pm$ 0.2	1.1 $\pm$ 0.5
FairGAT	77.5 $\pm$ 0.7	76.5 $\pm$ 0.2	74.9 $\pm$ 0.4	0.7 $\pm$ 0.5	0.5 $\pm$ 0.3	0.6 $\pm$ 0.3	0.7 $\pm$ 0.3	0.8 $\pm$ 0.3	0.8 $\pm$ 0.2
NT-FairGNN	77.0 $\pm$ 0.3	76.7 $\pm$ 0.3	74.9 $\pm$ 0.4	1.0 $\pm$ 0.5	1.0 $\pm$ 0.4	0.8 $\pm$ 0.2	1.2 $\pm$ 0.4	1.6 $\pm$ 0.2	1.1 $\pm$ 0.3
GE-REFEREE	75.6 $\pm$ 0.9	-	-	2.8 $\pm$ 0.3	-	-	0.8 $\pm$ 0.3	-	-
GEAR	70.0 $\pm$ 1.0	70.1 $\pm$ 1.5	69.1 $\pm$ 1.1	2.0 $\pm$ 0.5	2.2 $\pm$ 0.2	2.0 $\pm$ 0.7	0.9 $\pm$ 0.3	3.4 $\pm$ 0.3	6.0 $\pm$ 0.4
NIFTY	67.7 $\pm$ 0.8	70.9 $\pm$ 1.1	68.5 $\pm$ 1.5	3.0 $\pm$ 0.8	0.8 $\pm$ 0.5	4.7 $\pm$ 0.5	5.8 $\pm$ 0.9	4.0 $\pm$ 1.3	6.7 $\pm$ 0.7
Geb-NIFTY (Ours)	71.3 $\pm$ 0.5	72.8 $\pm$ 1.0	70.9 $\pm$ 1.1	0.6 $\pm$ 0.4	0.5 $\pm$ 0.3	0.9 $\pm$ 0.2	1.3 $\pm$ 0.5	0.8 $\pm$ 0.4	1.2 $\pm$ 0.5
Geb-GCN (Ours)	74.6 $\pm$ 0.8	71.4 $\pm$ 0.2	72.6 $\pm$ 0.4	0.5 $\pm$ 0.4	0.3$\pm$0.2	6.3 $\pm$ 0.4	0.6 $\pm$ 0.2	0.4$\pm$0.2	4.3 $\pm$ 0.6
Geb-GAT (Ours)	74.8 $\pm$ 0.6	73.1 $\pm$ 0.2	72.9 $\pm$ 0.6	0.4$\pm$0.3	0.3$\pm$0.2	6.3 $\pm$ 0.8	0.6 $\pm$ 0.2	0.4$\pm$0.1	5.1 $\pm$ 0.3
Geb-FairGCN (Ours)	74.9 $\pm$ 0.6	72.3 $\pm$ 0.2	72.4 $\pm$ 0.2	0.4$\pm$0.3	0.4 $\pm$ 0.3	0.2$\pm$0.1	0.5$\pm$0.1	0.4$\pm$0.2	0.3 $\pm$ 0.1
Geb-FairGAT (Ours)	74.9 $\pm$ 0.7	71.0 $\pm$ 0.7	74.8 $\pm$ 0.2	0.5 $\pm$ 0.3	0.6 $\pm$ 0.3	0.2$\pm$0.1	0.6 $\pm$ 0.1	0.6 $\pm$ 0.3	0.2$\pm$0.1

7.5 Experiments

7.5.1 Experimental Settings

We evaluate *Geb* on three real-world graph datasets: NBA [219], Pokec-z [223], and Pokec-n [223]. To investigate the flexibility of the proposed *Geb*, we incorporate it into GCN [224], GAT [225], NIFTY [226], FairGCN [219], and FairGAT [219], and rename them as *Geb*-GCN, *Geb*-GAT, *Geb*-NIFTY, *Geb*-FairGCN, and *Geb*-FairGAT, respectively. We compare the proposed *Geb* with several state-of-the-art methods: ALFR and ALFR-e [227], Debias and Debias-e [228], FCGE [229], FairGCN [219], FairGAT [219], NT-FairGNN [200], GE-REFEREE [207], GEAR [230], and NIFTY [226]. We use the AUC metric to measure the node classification performance. Δ_{SP} and Δ_{EO} are used to measure the fairness.

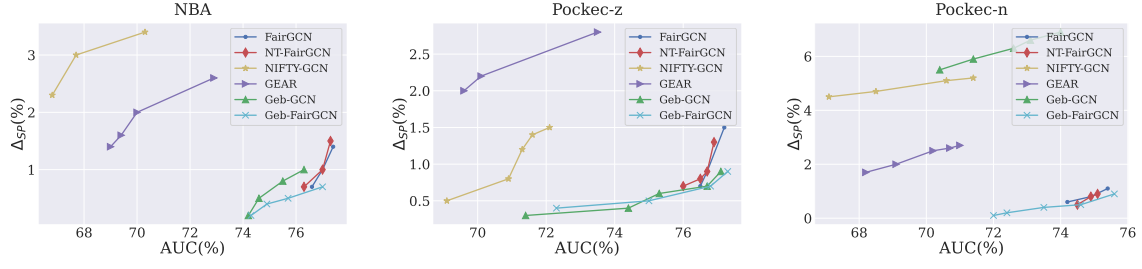


Figure 7.4: The obtained Pareto front on NBA, Pokec-z, and Pokec-n datasets, where the lower-right corner is desirable.

7.5.2 Comparisons with Baselines

Table 7.1 shows the performance comparison of the compared algorithms. The best performance is highlighted in bold. Experimental results show that *Geb*-GCN, *Geb*-GAT, *Geb*-NIFTY, *Geb*-FairGCN, and *Geb*-FairGAT outperform their vanilla counterparts in terms of the fairness metrics Δ_{SP} and Δ_{EO} , *e.g.*, compared to vanilla counterparts, the proposed *Geb* reduces Δ_{SP} by 3% ~ 9% without losing too much classification performance.

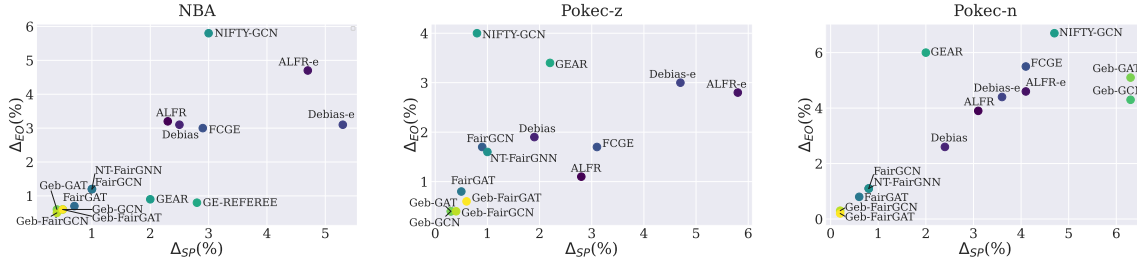


Figure 7.5: Visualization of fairness performance on NBA, Pokec-z, and Pokec-n datasets. The lower left corner, the fairer of the GNN model.

We also plot the Pareto front obtained by FairGCN, NT-FairGNN, NIFTY-GCN, GEAR, *Geb*-GCN, and *Geb*-FairGCN, as shown in Fig. 7.4. It should be noted that our proposed *Geb* can provide a set of Pareto solutions in one run. As for the compared algorithms, we ran multiple times to obtain the Pareto solutions. Obviously, the proposed *Geb* often arrive at the lower-right

corner, *i.e.*, high accuracy, low Δ_{SP} . That is, the solutions obtained by *Geb* Pareto dominate most of the solutions obtained by the compared algorithms. Moreover, we visualize the fairness performance of compared algorithms in terms of Δ_{SP} and Δ_{EO} in Fig. 7.5. Together with these experimental results, the effectiveness of applying the generated actionable explanations in enhancing fairness and maintaining classification performance can be demonstrated.

Table 7.2: The compactness values of the explanations generated by *Geb*, GEAR, and NIFTY, respectively.

Method	NBA	Pokec-z	Pokec-n
NIFTY-GCN	49.36	16.52	16.52
GEAR	49.41	16.53	16.53
<i>Geb</i> (Ours)	8.25	7.34	7.62

Interpretability Comparisons. In the field of explainable GNNs, subgraph-based explanations are often extracted for qualitative evaluation, and these explanations are used to verify whether the subgraph matches human knowledge. For example, some key substructures in molecules are extracted as the main factors causing molecular properties, and such key substructures should align with the chemists’ intuition or ground truth [231, 204, 209]. For quantitative evaluation, the compactness metric is used to measure the ratio of edges to nodes within the subgraph-based explanations [204, 209]. The intuition behind this metric is that the sparser the subgraph, the lower the burden on human understanding of the explanations, and the higher the interpretability.

In this work, we also follow the above commonly used evaluation protocols, *i.e.* extracting subgraph-based explanations for qualitative and quantitative evaluation. Specifically, we take the counterfactual graphs as the subgraph-based explanations. The compactness of explanations generated by *Geb*, GEAR, and NIFTY are presented in Table 7.2. It can be observed that *Geb* produces more compact explanations compared to GEAR and NIFTY. This is because NIFTY globally perturbs the structure to generate counterfactuals, making the explanations

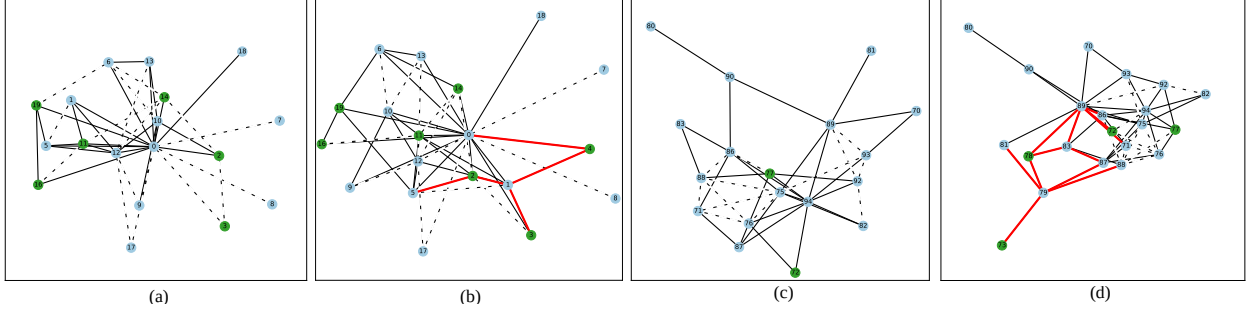


Figure 7.6: Explanation visualization on NBA dataset, where different node’s color denotes the nodes with different sensitive attributes. The solid black line, dashed line, and solid red line denote the raw edge, the selected edge in the biased subgraph, and the added edge in the counterfactual debiased subgraph, respectively. (a) The identified biased subgraph of the first 20 nodes. (b) The counterfactual debiased subgraph is generated based on the biased subgraph in (a). (c) The identified biased subgraph of the 70-th~ 90-th nodes. (d) The counterfactual debiased subgraph is generated based on the biased subgraph in (c). The explanation visualization indicates that nodes with the same sensitive attributes could amplify biases and breaking such connections can suppress the amplification of bias.

non-compact for humans to interpret. GEAR only perturbs features to output an entire graph instead of a set of compact subgraphs.

Moreover, two diagnostic explanations and two actionable explanations generated by *Geb* on the NBA dataset are plotted in Fig. 7.6 (a) and Fig. 7.6 (c), and Fig. 7.6 (b) and Fig. 7.6 (d), respectively. We can observe that the diagnostic explanations tend to remove positive edges (i.e., the edge that connects with the same sensitive attribute) to suppress the magnification of bias. Further, the actionable explanations are likely to add negative edges (i.e., the edge that connects with different sensitive attributes) to dilute bias. The explanation visualization aligns with human intuition, *i.e.*, nodes with the same sensitive attributes tend to be interconnected and could amplify biases. Therefore, the visualized subgraphs constitute explanations that guide the machine learning practitioners to accordingly curate the graph data to reduce bias.

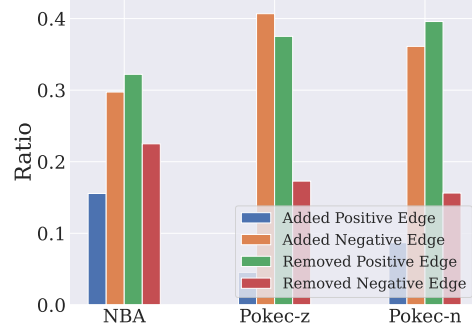


Figure 7.7: The ratio of each type of edge to all edges, where the actionable explanations produced by *Geb*-GCN is used to mitigate bias.

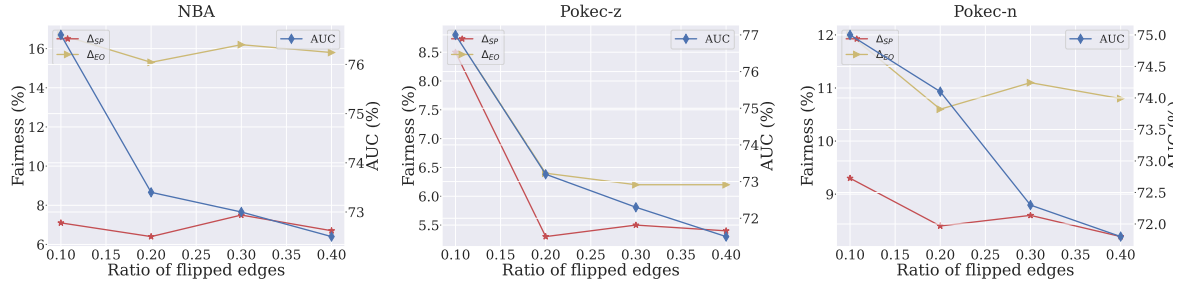


Figure 7.8: The change of the fairness performance and the AUC scores obtained by GCN when randomly flipping positive edges to negative edges with varied ratios.

In addition to visualizing explanatory subgraphs, we quantitatively analyze four types of edges (i.e., added positive edges, added negative edges, removed positive edges, and removed negative edges) to illustrate the behavior of *Geb* in reducing bias. From Fig. 7.7, we confirm that *Geb* tends to add negative edges and remove positive edges. Furthermore, inspired by this finding, we randomly flip positive edges to negative edges to see whether fairness can be improved while maintaining accuracy. The fairness evaluation and classification results are shown in Fig. 7.8. Notably, randomly flipping the positive edges may not necessarily improve fairness and maintain the classification performance, illustrating the rationality of using EAs to search biased patterns heuristically.

7.5.3 Ablation Studies

To verify the effectiveness of the generated biased subgraphs, we analyzed the performance of *Geb* at stage one (*Geb*-GCN-S1 and *Geb*-GAT-S1), *i.e.*, directly removing the biased subgraphs. It should be noted that only the edges of the biased subgraphs are removed from the graph data, but the nodes are preserved. From Table 7.3, we can see if we only remove the biased subgraph from the graph without performing stage two to repair them, the bias can also be reduced to some extent. This phenomenon shows that there are certain subgraph patterns in the graph that can magnify the bias, and the biased structures can be located by stage one of the proposed *Geb*. However, compared to *Geb*-GCN and *Geb*-GAT, directly removing the biased subgraphs can result in more loss of useful topology information, thus degrading the AUC scores more.

Table 7.3: The comparisons of our proposed *Geb* with its ablation versions (%)

Method	AUC ($\uparrow$)			Δ_{SP} ($\downarrow$)			Δ_{EO} ($\downarrow$)		
	NBA	Pokec-z	Pokec-n	NBA	Pokec-z	Pokec-n	NBA	Pokec-z	Pokec-n
GCN	78.3$\pm$0.3	77.2$\pm$0.1	75.1$\pm$0.2	7.9 $\pm$ 1.3	9.9 $\pm$ 1.1	9.6 $\pm$ 0.9	17.8 $\pm$ 2.6	9.1 $\pm$ 0.6	12.8 $\pm$ 1.3
GAT	78.2 $\pm$ 0.6	76.7 $\pm$ 0.1	75.1$\pm$0.2	10.2 $\pm$ 2.5	9.1 $\pm$ 0.9	9.4 $\pm$ 0.7	15.9 $\pm$ 4.0	8.4 $\pm$ 0.6	12.0 $\pm$ 1.5
<i>Geb</i> -GCN-S1	74.2 $\pm$ 0.6	74.7 $\pm$ 0.7	72.5 $\pm$ 0.6	1.8 $\pm$ 0.9	2.1 $\pm$ 0.6	6.5 $\pm$ 1.2	1.7 $\pm$ 1.0	1.8 $\pm$ 0.8	7.1 $\pm$ 0.4
<i>Geb</i> -GAT-S1	73.9 $\pm$ 0.8	73.2 $\pm$ 0.5	71.5 $\pm$ 1.2	2.5 $\pm$ 1.7	3.3 $\pm$ 0.4	6.4 $\pm$ 1.2	1.3 $\pm$ 0.8	3.4 $\pm$ 0.7	6.1 $\pm$ 1.1
<i>Geb</i> -GCN w/o Infl	73.2 $\pm$ 0.5	73.0 $\pm$ 0.4	71.2 $\pm$ 0.2	0.3$\pm$0.2	0.2$\pm$0.1	4.3 $\pm$ 0.3	0.6 $\pm$ 0.2	0.5 $\pm$ 0.4	4.1$\pm$0.3
<i>Geb</i> -GAT w/o Infl	74.3 $\pm$ 0.5	72.9 $\pm$ 0.5	72.1 $\pm$ 0.3	0.3$\pm$0.2	0.5 $\pm$ 0.4	5.2 $\pm$ 1.2	0.6 $\pm$ 0.1	0.8 $\pm$ 0.4	5.0 $\pm$ 1.0
<i>Geb</i> -GCN-FS	73.5 $\pm$ 0.4	73.2 $\pm$ 0.3	71.0 $\pm$ 0.3	0.3$\pm$0.2	0.3 $\pm$ 0.1	4.0$\pm$0.5	0.6 $\pm$ 0.2	0.4$\pm$0.2	4.3 $\pm$ 0.3
<i>Geb</i> -GAT-FS	74.0 $\pm$ 0.5	73.0 $\pm$ 0.3	71.8 $\pm$ 0.5	0.4 $\pm$ 0.2	0.4 $\pm$ 0.2	5.1 $\pm$ 1.5	0.5$\pm$0.2	0.7 $\pm$ 0.3	4.8 $\pm$ 0.7
<i>Geb</i> -GCN w/o cf	74.0 $\pm$ 0.4	75.1 $\pm$ 0.3	72.5 $\pm$ 0.6	1.6 $\pm$ 0.6	3.0 $\pm$ 0.6	6.7 $\pm$ 1.1	0.6 $\pm$ 0.2	3.3 $\pm$ 0.7	6.2 $\pm$ 1.5
<i>Geb</i> -GAT w/o cf	74.2 $\pm$ 0.5	74.8 $\pm$ 0.3	71.7 $\pm$ 1.7	1.7 $\pm$ 0.2	2.5 $\pm$ 0.2	6.3 $\pm$ 1.2	0.6 $\pm$ 0.3	2.8 $\pm$ 0.6	6.0 $\pm$ 1.3
<i>Geb</i> -GCN	74.6 $\pm$ 0.8	71.4 $\pm$ 0.2	72.6 $\pm$ 0.4	0.5 $\pm$ 0.4	0.3 $\pm$ 0.2	6.3 $\pm$ 0.4	0.6 $\pm$ 0.2	0.4$\pm$0.2	4.3 $\pm$ 0.6
<i>Geb</i> -GAT	74.8 $\pm$ 0.6	73.1 $\pm$ 0.2	72.9 $\pm$ 0.6	0.4 $\pm$ 0.3	0.3 $\pm$ 0.2	6.3 $\pm$ 0.8	0.6 $\pm$ 0.2	0.4$\pm$0.1	5.1 $\pm$ 0.3

To validate the effectiveness of the proposed influenced nodes-based gradient approximation, we evaluate the performance of *Geb* w/o Infl (i.e., based on the latest parameters, update the model with a few steps of gradient descent) and *Geb*-FS (i.e., train the model from scratch). From Table 7.3, it can be found that *Geb* w/o Infl and *Geb*-FS can achieve performance that

Table 7.4: The running time of the compared algorithms on NBA, Pokec-z, and Pokec-n datasets (seconds).

Method	NBA	Pokec-z	Pokec-n
FairGCN	5.34e+1	1.37e+2	1.24e+2
FairGAT	5.20e+1	1.36e+2	1.24e+2
NT-FairGNN	5.32e+1	1.40e+2	1.18e+2
GE-REFEREE	1.80e+1	OOM	OOM
NIFTY-GCN	5.21e+1	1.07e+2	9.71e+1
GEAR	6.38e+1	1.54e+2	1.60e+2
<i>Geb</i> -GCN w/o Infl	2.19e+4	5.86e+4	5.83e+4
<i>Geb</i> -GCN-FS	5.75e+5	1.50e+6	1.33e+6
<i>Geb</i> -GCN	1.49e+3	5.01e+4	5.01e+4

is not significantly different from the original ones. Besides, we evaluate the computational cost of *Geb*-GCN, and *Geb*-GAT, *Geb*-GCN-FS, *Geb*-GCN w/o Infl, *Geb*-GAT w/o Infl, and several baseline algorithms. As shown in Table 7.4, we can confirm that the proposed influenced nodes-based gradient approximation can cut running time to 10% $\sim$ 93% without significantly affecting classification performance and fairness. Though *Geb* consumes more running time than other baselines, we argue that it is a one-time cost. We want to highlight that *Geb* can achieve competitive results in terms of fairness and AUC scores and provide explainability — an essential aspect that other baselines do not have. To examine the impact of the generated counterfactual features, we evaluate the performance of *Geb* without the counterfactual features, and the corresponding ablation algorithms are renamed as *Geb* w/o cf. From Table 7.3, we can see that counterfactual features can further enhance fairness.

7.5.4 Sensitivity Studies

To investigate the impact of genetic operators, we plot the AUC scores and the Δ_{SP} values under varying crossover rates and mutation rates in Fig. 7.9 and Fig. 7.10, respectively.

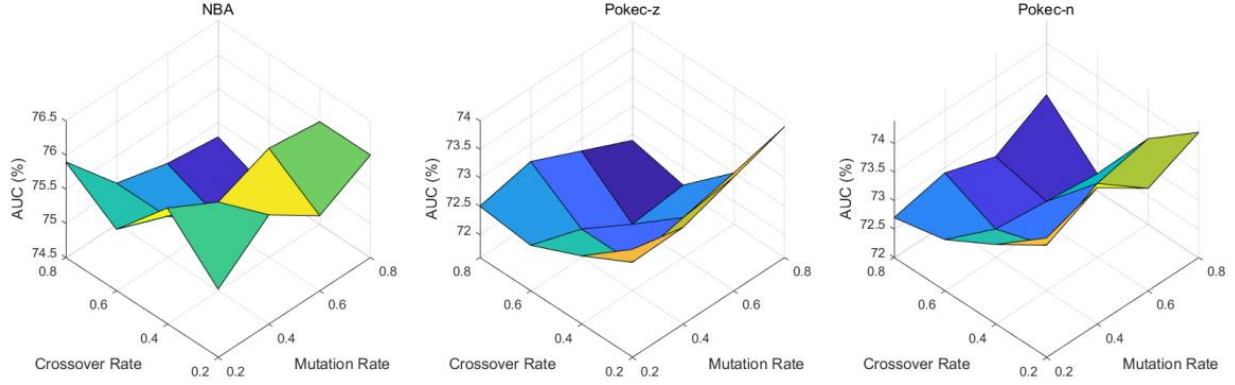


Figure 7.9: The impact of different crossover rates and mutation rates of *Geb*-GCN on AUC performance under NBA, Pokec-z, and Pokec-n datasets.

Generally, a mutation rate that is too high and a crossover rate that is too low will result in poor search performance.

Generation Distance (GD) [232] is used to measure the quality of the obtained solutions in terms of convergence. A smaller GD indicates that the solution set obtained by the EA is closer to the theoretical optima.

$$\mathbf{GD} = \frac{1}{|\mathbf{S}|} \left(\sum_{i=1}^{|\mathbf{S}|} \|\mathbf{S}_i - \mathbf{r}\| \right)^{1/2}, \quad (7.28)$$

where $\mathbf{S}$ is the obtained non-dominated solution set, and $\mathbf{r}$ is an ideal point, *i.e.*, the theoretically optimal values of the three objectives. In this work, $\mathbf{r}$ is set to $\langle 0.5, 1, 1 \rangle$.

The convergence traces in terms of GD values on NBA, Pokec-z, and Pokec-n datasets are plotted in Fig. 7.11. Although the GD values indicate that the proposed EA has converged when the population size is set to 40, the algorithm may suffer from the local optima. We believe more efficient search algorithms are worth studying to improve the search performance of multiobjective optimization toward Pareto optimal solutions.

Population size is a key parameter in the multiobjective EA. A large population size can lead to an increase in computational resources, while a small population size can lead to the failure

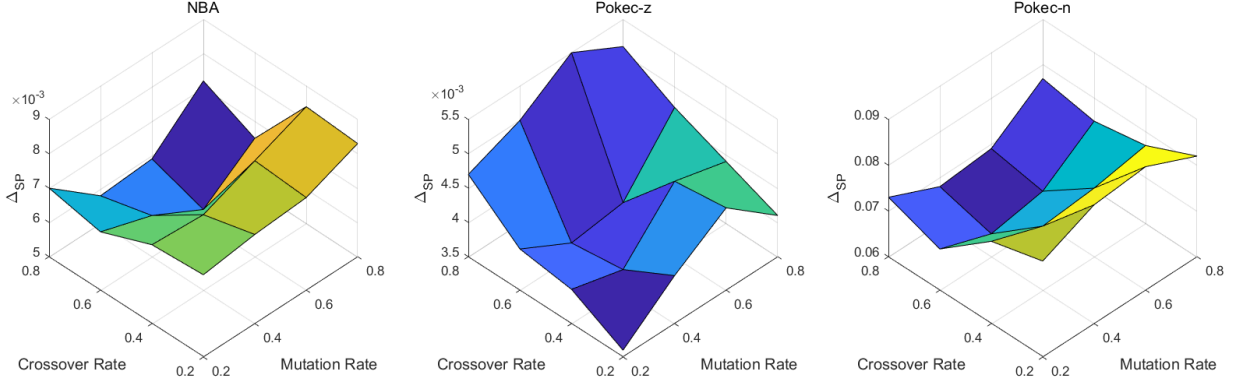


Figure 7.10: The impact of different crossover rates and mutation rates of *Geb*-GCN on Δ_{sp} values under NBA, Pokec-z, and Pokec-n datasets. A mutation rate that is too high and a crossover rate that is too low will degrade the ability to search for counterfactual debiased subgraphs.

of searching for promising solution traits. To investigate the performance with respect to the population size, we plot the fairness values and the AUC under varying population sizes in Fig. 7.12. It can be observed that the proposed *Geb* can achieve acceptable performance in terms of AUC and fairness when the population size is set to 40, indicating that promising traits for identifying the biased behaviors have emerged in the population. Therefore, for the sake of saving computational resource consumption, this work sets the population size to 40.

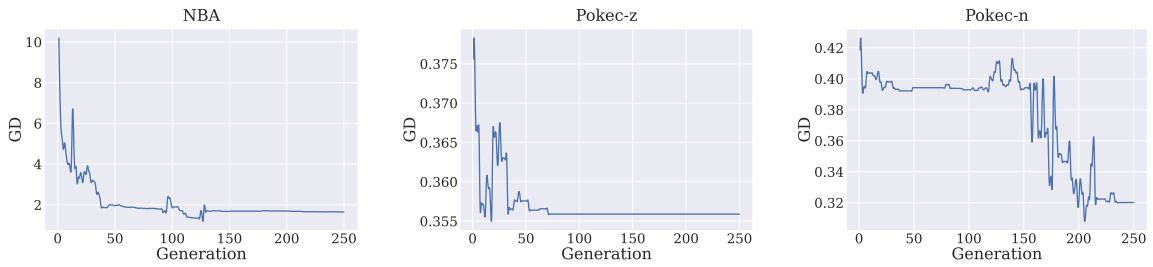


Figure 7.11: The convergence traces of the proposed EA on NBA, Pokec-z, and Pokec-n datasets.

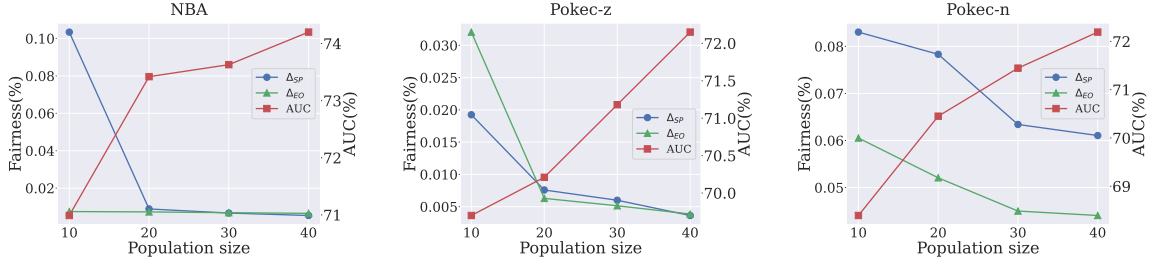


Figure 7.12: Analysis of varying population sizes of *Geb*-GCN on NBA, Pokec-z, and Pokec-n datasets, which indicates a small population size can lead to the failure of searching the promising solution traits.

7.6 Conclusion

This work advanced a model-agnostic fair GNN framework to generate compact and interpretable model-level explanations for locating the source of bias and rectifying the biased behavior. In particular, a multiobjective EA is designed to generate diagnostic explanations and actionable explanations. The former explains the root cause of the biased behavior of the GNNs, highlighting nodes or edges causing the bias. The latter provides counterfactuals on how to rectify this bias, demonstrating the specific nodes or edges that should be added or removed. To address the computational challenge in the EA, we proposed an influenced nodes-based gradient approximation to update the GNN models. We also provided a theoretical fairness risk bound to illustrate the effectiveness of *Geb* in reducing the bias. The experimental results exhibit promising advancements in terms of classification performance, fairness, and interpretability.

Chapter 8

Summary and Future Work

8.1 Summary

This thesis has elaborated on a series of research on knowledge-driven EAs for complex optimization problems, including large-scale multiobjective optimization, dynamic multiobjective optimization, dynamic constrained multiobjective optimization, multitasking optimization, and a fair graph learning task involving multiobjective optimization. The main contributions of this thesis can be summarized as the following five aspects.

1. In Chapter 3, to solve large-scale multiobjective optimization problems, a knowledge-driven offspring generation operator is designed. Typically, by learning from the manifold constructed by the non-dominated solutions, a dedicated generative adversarial network can generate high-quality offspring solutions that are distributed on the manifold. In addition, three interpolation strategies are incorporated with the generative adversarial network to manipulate the low-dimensional latent variables to generate more diverse offspring. With the above design, high-quality offspring solutions can be generated, mitigating the issue of search inefficiency in large-scale optimization problems.

2. In Chapter 4, to solve dynamic multiobjective optimization problems, a knowledge-driven computational resource allocation strategy is proposed to adaptively adjust search strategies in dynamic environments. The work proposed a Monte Carlo tree policy that reduces the computational waste on improper evolutionary operators and suboptimal subpopulations, enabling the evolutionary strategies to adapt to the characteristics of the changing environment. Specifically, a Monte Carlo tree-assisted computational resource allocation strategy is proposed to activate promising subpopulations and allocate more computing resources to them. Moreover, the activated subpopulations can flexibly select the evolutionary operators to improve the search efficiency of the new environment. This work first considers resource allocation problems in dynamic environments, advancing dynamic optimization algorithms toward more complex and practical scenarios.
3. In Chapter 5, to solve dynamic constrained multiobjective optimization problems, a spatial-temporal knowledge-based dynamic response strategy is devised to accelerate the search for feasible solutions in dynamic environments. This work establishes two synergistic tasks: one task is focused on the unconstrained search space to maintain diversity, while the other involves searching the constrained search space to focus on feasibility. Particularly, the spatial and temporal knowledge transfer modules are designed to reuse the knowledge of the dynamic environment and constrained/unconstrained search space, thus accelerating the search for the two synergistic tasks. This work first introduces the multitask paradigm into dynamic constrained multiobjective optimization, and a test suite including 14 test problems with various characteristics is proposed to advance DCMOPs toward real-world cases.
4. In Chapter 6, to solve multitask optimization problems, a knowledge-driven solution selection method for evolutionary multitasking is developed to transfer high-quality solutions for the target optimization problems. In this work, the transfer Gaussian process is employed to model the solution distribution of both the source task and target task, mit-

igating the negative transfer caused by the lack of exploiting the solution distribution. In addition, an inter-task similarity-based lower confidence bound metric is designed to estimate the potential improvement of the transferred solutions on the target task, enabling the algorithm to identify high-quality solutions to the target task. The proposed solution selection approach can be seamlessly incorporated into most existing EMT algorithms and promote positive knowledge transfer.

5. In Chapter 7, by formulating the fair graph learning problems as multiobjective optimization problems, a dedicated multiobjective EA is designed to ensure the explainability, fidelity, and fairness of GNNs in high-stake applications. The work can produce compact, diagnostic, and actionable subgraph patterns that are responsible for the GNNs' biased behavior. An influenced nodes-based gradient approximation is proposed to reduce evaluation costs in fair graph learning to avoid retraining the graph neural networks. This work first utilizes multiobjective optimization algorithms to balance interpretability, fairness, and fidelity in fair graph learning.

8.2 Future Work

While the research presented in this thesis has made progress in the field of knowledge-driven evolutionary optimization, there are several avenues for future work that merit exploration.

One direction is to further enhance the problem-solving capabilities of the proposed framework by incorporating more advanced machine-learning techniques, e.g., large language models [233]. This could involve the development of more sophisticated models that can accurately capture the complex relationships between problem features and optimal solutions. In addition, the integration of deep learning architectures could enable the framework to learn high-level representations of the problem landscape, allowing for more effective exploration and exploitation strategies.

Another area for future research is the extension of knowledge-driven evolutionary optimization to handle a broader range of optimization problems. Adapting the proposed methods to handle these more complex problem formulations would expand the applicability of the framework and its ability to tackle real-world applications across diverse domains, such as engineering design, finance, healthcare, or logistics. By leveraging diverse forms of problem-relevant information, knowledge-driven evolutionary optimization could further improve its ability to navigate complex search spaces and converge to high-quality solutions more efficiently.

Lastly, the scalability and computational efficiency of the knowledge-driven evolutionary optimization algorithms could be investigated. The strategies for leveraging parallel architectures, cloud computing platforms, or graphics processing unit acceleration could expedite the optimization process to handle computationally intensive tasks [234], unleashing the full potential of the knowledge-driven approach.

References

- [1] Ram Bhusan Agrawal, Kalyanmoy Deb, and Ram Bhushan Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9(3):115–148, 1994.
- [2] Kangjia Qiao, Kunjie Yu, Boyang Qu, Jing Liang, Hui Song, and Caitong Yue. An evolutionary multitasking optimization framework for constrained multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 26(2):263–277, 2022.
- [3] Kalyanmoy Deb, Udaya Bhaskara Rao N., and S. Karthik. Dynamic multi-objective optimization and decision-making using modified nsga-ii: A case study on hydro-thermal power scheduling. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization*, pages 803–817, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
- [4] Kay Chen Tan, Yoong Han Chew, and Loo Hay Lee. A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855–885, 2006.
- [5] Zhenzhong Wang, Kai Ye, Min Jiang, Junfeng Yao, Neal N. Xiong, and Gary G. Yen. Solving hybrid charging strategy electric vehicle based dynamic routing problem via evolutionary multi-objective optimization. *Swarm and Evolutionary Computation*, page 100975, 2021.

-
- [6] Zhenzhong Wang, Qingyuan Zeng, Wanyu Lin, Min Jiang, and Kay Chen Tan. Generating diagnostic and actionable explanations for fair graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19):21690–21698, Mar. 2024.
 - [7] Liang Feng, Qingxia Shang, Yaqing Hou, Kay Chen Tan, and Yew-Soon Ong. Multi-space evolutionary search for large-scale optimization with applications to recommender systems. *IEEE Transactions on Artificial Intelligence*, 4(1):107–120, 2023.
 - [8] Zhenzhong Wang, Lulu Cao, Liang Feng, Min Jiang, and Kay Chen Tan. Evolutionary multitask optimization with lower confidence bound-based solution selection strategy. *IEEE Transactions on Evolutionary Computation*, 2024.
 - [9] Jinliang Ding, Cuie Yang, Qiong Xiao, Tianyou Chai, and Yaochu Jin. Dynamic evolutionary multiobjective optimization for raw ore allocation in mineral processing. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(1):36–48, 2019.
 - [10] Lulu Cao, Yufei Liu, Zhenzhong Wang, Dejun Xu, Kai Ye, Kay Chen Tan, and Min Jiang. An interpretable approach to the solutions of high-dimensional partial differential equations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20640–20648, 2024.
 - [11] Kay Chen Tan, Liang Feng, and Min Jiang. Evolutionary transfer optimization - a new frontier in evolutionary computation research. *IEEE Computational Intelligence Magazine*, 16(1):22–33, 2021.
 - [12] Songbai Liu, Qiuzhen Lin, Jianqiang Li, and Kay Chen Tan. A survey on learnable evolutionary algorithms for scalable multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 27(6):1941–1961, 2023.
 - [13] C. He, R. Cheng, Y. Tian, X. Zhang, K. C. Tan, and Y. Jin. Paired offspring generation for constrained large-scale multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2020.

- [14] Ye Tian, Xiutao Zheng, Xingyi Zhang, and Yaochu Jin. Efficient large-scale multiobjective optimization based on a competitive swarm optimizer. *IEEE Transactions on Cybernetics*, 50(8):3696–3708, 2020.
- [15] Ye Tian, Langchun Si, Xingyi Zhang, Ran Cheng, Cheng He, Kay Tan, and Yaochu Jin. Evolutionary large-scale multi-objective optimization: A survey. *ACM Computing Surveys*, 06 2021.
- [16] L. M. Antonio and C. A. C. Coello. Use of cooperative coevolution for solving large scale multiobjective optimization problems. In *2013 IEEE Congress on Evolutionary Computation*, pages 2758–2765, June 2013.
- [17] X. Zhang, Y. Tian, R. Cheng, and Y. Jin. A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 22(1):97–112, Feb 2018.
- [18] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong. A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables. *IEEE Transactions on Evolutionary Computation*, 20(2):275–298, April 2016.
- [19] H. Hong, K. Ye, M. Jiang, and K.C. Tan. Solving large-scale multi-objective optimization via probabilistic prediction model. In *11th Edition of International Conference Series on Evolutionary Multi-Criterion Optimization*, 2021.
- [20] Zhenyu Yang, Ke Tang, and Xin Yao. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15):2985–2999, 2014.
- [21] Cheng He, Lianghao Li, Ye Tian, Xingyi Zhang, Ran Cheng, Yaochu Jin, and Xin Yao. Accelerating large-scale multiobjective optimization via problem reformulation. *IEEE Transactions on Evolutionary Computation*, 23(6):949–961, 2019.

-
- [22] Heiner Zille, Hisao Ishibuchi, Sanaz Mostaghim, and Yusuke Nojima. A framework for large-scale multiobjective optimization based on problem transformation. *IEEE Transactions on Evolutionary Computation*, 22(2):260–275, 2017.
 - [23] He Cheng, Shihua Huang, Ran Cheng, Kay Chen Tan, and Yaochu Jin. Evolutionary multiobjective optimization driven by generative adversarial networks (gans). *IEEE Transactions on Cybernetics*, 2020.
 - [24] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff. A multiobjective evolutionary algorithm using gaussian process-based inverse modeling. *IEEE Transactions on Evolutionary Computation*, 19(6):838–856, Dec 2015.
 - [25] Y. Tian, X. Zhang, C. Wang, and Y. Jin. An evolutionary algorithm for large-scale sparse multi-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2019.
 - [26] Shouyong Jiang and Shengxiang Yang. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 21(1):65–82, 2017.
 - [27] Gan Ruan, Guo Yu, Jinhua Zheng, Juan Zou, and Shengxiang Yang. The effect of diversity maintenance on prediction in dynamic multi-objective optimization. *Applied Soft Computing*, 58:631–647, 2017.
 - [28] Ruochen Liu, Luyao Peng, Jiangdi Liu, and Jing Liu. A diversity introduction strategy based on change intensity for evolutionary dynamic multiobjective optimization. *Soft Computing*, 24(17):12789–12799, 2020.
 - [29] Danial Yazdani, Mohammad Nabi Omidvar, Jürgen Branke, Trung Thanh Nguyen, and Xin Yao. Scaling up dynamic optimization problems: A divide-and-conquer approach. *IEEE Transactions on Evolutionary Computation*, 24(1):1–15, 2020.

- [30] R. Chen, K. Li, and X. Yao. Dynamic multiobjectives optimization with a changing number of objectives. *IEEE Transactions on Evolutionary Computation*, 22(1):157–171, Feb 2018.
- [31] Zhengping Liang, Tiancheng Wu, Xiaoliang Ma, Zexuan Zhu, and Shengxiang Yang. A dynamic multiobjective evolutionary algorithm based on decision variable classification. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
- [32] Miao Rong, Dunwei Gong, Yong Zhang, Yaochu Jin, and Witold Pedrycz. Multidirectional prediction approach for dynamic multiobjective optimization problems. *IEEE Transactions on Cybernetics*, 49(9):3362–3374, 2019.
- [33] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 3, pages 1875–1882 Vol. 3, 1999.
- [34] Hidehiro Nakano, Masataka Kojima, and Arata Miyauchi. An artificial bee colony algorithm with a memory scheme for dynamic optimization problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 2657–2663, 2015.
- [35] Min Jiang, Zhenzhong Wang, Liming Qiu, Shihui Guo, Xing Gao, and Kay Chen Tan. A fast dynamic evolutionary multiobjective algorithm via manifold transfer learning. *IEEE Transactions on Cybernetics*, 51(7):3417–3428, 2021.
- [36] Dejun Xu, Min Jiang, Weizhen Hu, Shaozi Li, Renhu Pan, and Gary G. Yen. An online prediction approach based on incremental support vector machine for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2021.
- [37] Leilei Cao, Lihong Xu, Erik D. Goodman, Chunteng Bao, and Shuwei Zhu. Evolutionary dynamic multiobjective optimization assisted by a support vector regression predictor. *IEEE Transactions on Evolutionary Computation*, 24(2):305–319, 2020.

-
- [38] Xinxin Xu, Yanyan Tan, Wei Zheng, and Shengtao Li. Memory-enhanced dynamic multi-objective evolutionary algorithm based on lp decomposition. *Applied Sciences*, 8(9), 2018.
 - [39] Radhia Azzouz, Slim Bechikh, and Lamjed Ben Said. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. *Soft Computing*, 21(4):885–906, 2017.
 - [40] Jun Zhang, Zhi-hui Zhan, Ying Lin, Ni Chen, Yue-jiao Gong, Jing-hui Zhong, Henry S.H. Chung, Yun Li, and Yu-hui Shi. Evolutionary computation meets machine learning: A survey. *IEEE Computational Intelligence Magazine*, 6(4):68–75, 2011.
 - [41] Dan Guo, Xilu Wang, Kailai Gao, Yaochu Jin, Jinliang Ding, and Tianyou Chai. Evolutionary optimization of high-dimensional multiobjective and many-objective expensive problems assisted by a dropout neural network. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–14, 2021.
 - [42] Zhenzhong Wang, Haokai Hong, Kai Ye, Guang-En Zhang, Min Jiang, and Kay Chen Tan. Manifold interpolation for large-scale multiobjective optimization via generative adversarial networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(8):4631–4645, 2023.
 - [43] Cheng He, Shihua Huang, Ran Cheng, Kay Chen Tan, and Yaochu Jin. Evolutionary multiobjective optimization driven by generative adversarial networks (gans). *IEEE Transactions on Cybernetics*, 51(6):3129–3142, 2021.
 - [44] Kunjie Yu, Dezheng Zhang, Jing Liang, Ke Chen, Caitong Yue, Kangjia Qiao, and Ling Wang. A correlation-guided layered prediction approach for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2022.

- [45] Rethnaraj Rambabu, Prahlad Vadakkepat, Kay Chen Tan, and Min Jiang. A mixture-of-experts prediction framework for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 50(12):5099–5112, 2020.
- [46] Min Jiang, Zhongqiang Huang, Liming Qiu, Wenzhen Huang, and Gary Yen. Transfer learning based dynamic multiobjective optimization algorithms. *IEEE Transactions on Evolutionary Computation*, PP(99):1–1, 2017.
- [47] Huan Zhang, Jinliang Ding, Min Jiang, Kay Chen Tan, and Tianyou Chai. Inverse gaussian process modeling for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, pages 1–14, 2021.
- [48] Xiao-Fang Liu, Zhi-Hui Zhan, Tian-Long Gu, Sam Kwong, Zhenyu Lu, Henry Been-Lirn Duh, and Jun Zhang. Neural network-based information transfer for dynamic optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 31(5):1557–1570, 2020.
- [49] Qingyang Zhang, Shengxiang Yang, Shouyong Jiang, Ronggui Wang, and Xiaoli Li. Novel prediction strategies for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 24(2):260–274, 2020.
- [50] Arrchana Muruganantham, Kay Chen Tan, and Prahlad Vadakkepat. Evolutionary dynamic multiobjective optimization via kalman filter prediction. *IEEE Trans. Cybernetics*, 46(12):2862–2873, 2016.
- [51] Liang Feng, Wei Zhou, Weichen Liu, Yew-Soon Ong, and Kay Chen Tan. Solving dynamic multiobjective problem via autoencoding evolutionary search. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
- [52] Fei Zou, Gary G. Yen, and Chen Zhao. Dynamic multiobjective optimization driven by inverse reinforcement learning. *Information Sciences*, 575:468–484, 2021.

-
- [53] Min Jiang, Zhenzhong Wang, Haokai Hong, and Gary G. Yen. Knee point-based imbalanced transfer learning for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 25(1):117–129, 2021.
 - [54] Qi Zhao, Bai Yan, Yuhui Shi, and Martin Middendorf. Evolutionary dynamic multiobjective optimization via learning from historical search process. *IEEE Transactions on Cybernetics*, pages 1–12, 2021.
 - [55] Shu Luo, Linxuan Zhang, and Yushun Fan. Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning. *Computers and Industrial Engineering*, 159:107489, 2021.
 - [56] Yinan Guo, Guoyu Chen, Min Jiang, Dunwei Gong, and Jing Liang. A knowledge guided transfer strategy for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2022.
 - [57] Zhenzhong Wang, Min Jiang, Xing Gao, Liang Feng, Weizhen Hu, and Kay Chen Tan. Evolutionary dynamic multi-objective optimization via regression transfer learning. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2375–2381, 2019.
 - [58] Xiangxiang Zeng, Wen Wang, Cong Chen, and Gary G. Yen. A consensus community-based particle swarm optimization for dynamic community detection. *IEEE Transactions on Cybernetics*, 50(6):2502–2513, 2020.
 - [59] Min Jiang, Liming Qiu, Zhongqiang Huang, and Gary G. Yen. Dynamic multi-objective estimation of distribution algorithm based on domain adaptation and nonparametric estimation. *Information Sciences*, 435:203–223, 2018.
 - [60] Kunjie Yu, Jing Liang, Boyang Qu, Yong Luo, and Caitong Yue. Dynamic selection preference-assisted constrained multiobjective differential evolution. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–12, 2021.

- [61] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
- [62] Zhi-Zhong Liu, Bing-Chuan Wang, and Ke Tang. Handling constrained multiobjective optimization problems via bidirectional coevolution. *IEEE Transactions on Cybernetics*, 52(10):10163–10176, 2021.
- [63] Yonas Gebre Woldesenbet, Gary G. Yen, and Biruk G. Tessema. Constraint handling in multiobjective evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 13(3):514–525, 2009.
- [64] Radhia Azzouz, Slim Bechikh, Lamjed Ben Said, and Walid Trabelsi. Handling time-varying constraints and objectives in dynamic evolutionary multi-objective optimization. *Swarm and Evolutionary Computation*, 39:222–248, 2018.
- [65] Radhia Azzouz, Slim Bechikh, and Lamjed 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, GECCO '15*, page 615–622, New York, NY, USA, 2015. Association for Computing Machinery.
- [66] Qingda Chen, Jinliang Ding, Shengxiang Yang, and Tianyou Chai. A novel evolutionary algorithm for dynamic constrained multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 24(4):792–806, 2020.
- [67] Zhongwei Ma and Yong Wang. Shift-based penalty for evolutionary constrained multiobjective optimization and its application. *IEEE Transactions on Cybernetics*, 53(1):18–30, 2023.
- [68] T.P. Runarsson and Xin Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, 2000.

-
- [69] Gregorio Toscano, Ricardo Landa, Giomara Lárraga, and Guillermo Leguizamón. On the use of stochastic ranking for parent selection in differential evolution for constrained optimization. *Soft Computing*, 21(16):4617–4633, 2017.
- [70] Ye Tian, Yajie Zhang, Yansen Su, Xingyi Zhang, Kay Chen Tan, and Yaochu Jin. Balancing objective optimization and constraint satisfaction in constrained evolutionary multi-objective optimization. *IEEE Transactions on Cybernetics*, 52(9):9559–9572, 2022.
- [71] Zhun Fan, Wenji Li, Xinye Cai, Han Huang, Yi Fang, Yugen You, Jiajie Mo, Caimin Wei, and Erik Goodman. An improved epsilon constraint-handling method in moea/d for cmops with large infeasible regions. *Soft Computing*, 23(23):12491–12510, 2019.
- [72] María-Yaneli Ameca-Alducin, Maryam Hasani-Shoreh, Wilson Blaikie, Frank Neumann, and Efrén Mezura-Montes. A comparison of constraint handling techniques for dynamic constrained optimization problems. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2018.
- [73] Kangjia Qiao, Kunjie Yu, Boyang Qu, Jing Liang, Hui Song, Caitong Yue, Hongyu Lin, and Kay Chen Tan. Dynamic auxiliary task-based evolutionary multitasking for constrained multi-objective optimization. *IEEE Transactions on Evolutionary Computation*, 2022.
- [74] Chaoda Peng, Hai-Lin Liu, and Fangqing Gu. An evolutionary algorithm with directed weights for constrained multi-objective optimization. *Applied Soft Computing*, 60:613–622, 2017.
- [75] Tapabrata Ray, Hemant Kumar Singh, Amitay Isaacs, and Warren Smith. Infeasibility driven evolutionary algorithm for constrained optimization. In *Constraint-handling in evolutionary optimization*, pages 145–165. Springer, 2009.

- [76] Bo Yang Qu and Ponnuthurai Nagarathnam Suganthan. Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods. *Engineering Optimization*, 43(4):403–416, 2011.
- [77] Ashish Mani and C. Patvardhan. A novel hybrid constraint handling technique for evolutionary optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 2577–2583, 2009.
- [78] Ruwang Jiao, Bing Xue, and Mengjie Zhang. A multiform optimization framework for constrained multiobjective optimization. *IEEE Transactions on Cybernetics*, 2022.
- [79] Jing Liang, Xuanxuan Ban, Kunjie Yu, Boyang Qu, Kangjia Qiao, Caitong Yue, Ke Chen, and Kay Chen Tan. A survey on evolutionary constrained multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 27(2):201–221, 2023.
- [80] Chenyang Bu, Wenjian Luo, and Lihua Yue. Continuous dynamic constrained optimization with ensemble of locating and tracking feasible regions strategies. *IEEE Transactions on Evolutionary Computation*, 21(1):14–33, 2017.
- [81] Qingda Chen, Jinliang Ding, Gary G. Yen, Shengxiang Yang, and Tianyou Chai. Multi-population evolution based dynamic constrained multiobjective optimization under diverse changing environments. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2023.
- [82] Guoyu Chen, Yinan Guo, Min Jiang, Shengxiang Yang, Xiaoxiao Zhao, and Dunwei Gong. A subspace-knowledge transfer based dynamic constrained multiobjective evolutionary algorithm. *IEEE Transactions on Emerging Topics in Computational Intelligence*, pages 1–13, 2023.
- [83] Jian-Yu Li, Zhi-Hui Zhan, Kay Chen Tan, and Jun Zhang. A meta-knowledge transfer-based differential evolution for multitask optimization. *IEEE Transactions on Evolutionary Computation*, 26(4):719–734, 2022.

-
- [84] Tingyang Wei, Shibin Wang, Jinghui Zhong, Dong Liu, and Jun Zhang. A review on evolutionary multitask optimization: Trends and challenges. *IEEE Transactions on Evolutionary Computation*, 26(5):941–960, 2022.
- [85] Jinliang Ding, Cuie Yang, Yaochu Jin, and Tianyou Chai. Generalized multitasking for evolutionary optimization of expensive problems. *IEEE Transactions on Evolutionary Computation*, 23(1):44–58, 2019.
- [86] Liang Feng, Lei Zhou, Jinghui Zhong, Abhishek Gupta, Yew-Soon Ong, Kay-Chen Tan, and A. K. Qin. Evolutionary multitasking via explicit autoencoding. *IEEE Transactions on Cybernetics*, 49(9):3457–3470, 2019.
- [87] Kavitesh Kumar Bali, Yew-Soon Ong, Abhishek Gupta, and Puay Siew Tan. Multifactorial evolutionary algorithm with online transfer parameter estimation: Mfea-ii. *IEEE Transactions on Evolutionary Computation*, 24(1):69–83, 2020.
- [88] Dongrui Wu and Xianfeng Tan. Multitasking genetic algorithm (mtga) for fuzzy system optimization. *IEEE Transactions on Fuzzy Systems*, 28(6):1050–1061, 2020.
- [89] Yi Jiang, Zhi-Hui Zhan, Kay Chen Tan, and Jun Zhang. Block-level knowledge transfer for evolutionary multitask optimization. *IEEE Transactions on Cybernetics*, pages 1–14, 2023.
- [90] Kavitesh Kumar Bali, Abhishek Gupta, Liang Feng, Yew Soon Ong, and Tan Puay Siew. Linearized domain adaptation in evolutionary multitasking. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1295–1302, 2017.
- [91] Lei Zhou, Liang Feng, Abhishek Gupta, and Yew-Soon Ong. Learnable evolutionary search across heterogeneous problems via kernelized autoencoding. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2021.

- [92] Jun Zhang, Weien Zhou, Xianqi Chen, Wen Yao, and Lu Cao. Multisource selective transfer framework in multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 24(3):424–438, 2020.
- [93] Yi Jiang, Zhi-Hui Zhan, Kay Chen Tan, and Jun Zhang. A bi-objective knowledge transfer framework for evolutionary many-task optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2022.
- [94] Sheng-Hao Wu, Zhi-Hui Zhan, Kay Chen Tan, and Jun Zhang. Orthogonal transfer for multitask optimization. *IEEE Transactions on Evolutionary Computation*, 27(1):185–200, 2023.
- [95] Kangjia Qiao, Kunjie Yu, Boyang Qu, Jing Liang, Hui Song, Caitong Yue, Hongyu Lin, and Kay Chen Tan. Dynamic auxiliary task-based evolutionary multitasking for constrained multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 27(3):642–656, 2023.
- [96] Songbai Liu, Qiuzhen Lin, Liang Feng, Ka-Chun Wong, and Kay Chen Tan. Evolutionary multitasking for large-scale multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2022.
- [97] Yue Wu, Hangqi Ding, Maoguo Gong, A. K. Qin, Wenping Ma, Qiguang Miao, and Kay Chen Tan. Evolutionary multiform optimization with two-stage bidirectional knowledge transfer strategy for point cloud registration. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2022.
- [98] Ziyu Hu, Yulin Li, Hao Sun, and Xuemin Ma. Multitasking multiobjective optimization based on transfer component analysis. *Information Sciences*, 605:182–201, 2022.
- [99] Zedong Tang, Maoguo Gong, Yue Wu, Wenfeng Liu, and Yu Xie. Regularized evolutionary multitask optimization: Learning to intertask transfer in aligned subspace. *IEEE Transactions on Evolutionary Computation*, 25(2):262–276, 2021.

-
- [100] Zhengping Liang, Hao Dong, Cheng Liu, Weiqi Liang, and Zexuan Zhu. Evolutionary multitasking for multiobjective optimization with subspace alignment and adaptive differential evolution. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
- [101] Zefeng Chen, Yuren Zhou, Xiaoyu He, and Jun Zhang. Learning task relationships in evolutionary multitasking for multiobjective continuous optimization. *IEEE Transactions on Cybernetics*, pages 1–12, 2020.
- [102] Hao Li, Yew-Soon Ong, Maoguo Gong, and Zhenkun Wang. Evolutionary multitasking sparse reconstruction: Framework and case study. *IEEE Transactions on Evolutionary Computation*, 23(5):733–747, 2018.
- [103] Zhenzhong Wang, Lulu Cao, Wanyu Lin, Min Jiang, and Kay Chen Tan. Robust graph meta-learning via manifold calibration with proxy subgraphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(12):15224–15232, Jun. 2023.
- [104] Xiaoming Xue, Kai Zhang, Kay Chen Tan, Liang Feng, Jian Wang, Guodong Chen, Xing-gang Zhao, Liming Zhang, and Jun Yao. Affine transformation-enhanced multifactorial optimization for heterogeneous problems. *IEEE Transactions on Cybernetics*, pages 1–15, 2020.
- [105] Min Jiang, Zhenzhong Wang, Shihui Guo, Xing Gao, and Kay Chen Tan. Individual-based transfer learning for dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 51(10):4968–4981, 2021.
- [106] A. Ponsich, A. L. Jaimes, and C. A. C. Coello. A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *IEEE Transactions on Evolutionary Computation*, 17(3):321–344, June 2013.
- [107] Y. Zhou, G. G. Yen, and Z. Yi. Evolutionary shallowing deep neural networks at block levels. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2021.

- [108] ZP Stanko, T Nishikawa, and Scott R Paulinski. Large-scale multi-objective optimization for the management of seawater intrusion, santa barbara, ca. In *AGU Fall Meeting Abstracts*, pages H31G–1507, 2015.
- [109] Zhenyao Zhao, Min Jiang, Shihui Guo, Zhenzhong Wang, Fei Chao, and Kay Chen Tan. Improving deep learning based optical character recognition via neural architecture search. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–7, 2020.
- [110] J. Cheney. The application of optimisation methods to the design of large scale telecommunication networks. In *IEE Colloquium on Large-Scale and Hierarchical Systems*, pages 2/1–2/2, March 1988.
- [111] Handing Wang, Licheng Jiao, Ronghua Shang, Shan He, and Fang Liu. A memetic optimization strategy based on dimension reduction in decision space. *Evolutionary Computation*, 23(1):69–100, 2015.
- [112] S. Yang, Y. Tian, C. He, X. Zhang, K. C. Tan, and Y. Jin. A gradient-guided evolutionary approach to training deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2021.
- [113] C. Zhao and D. Guo. Particle swarm optimization algorithm with self-organizing mapping for nash equilibrium strategy in application of multiobjective optimization. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2020.
- [114] B. Wang, B. Xue, and M. Zhang. Surrogate-assisted particle swarm optimization for evolving variable-length transferable blocks for image classification. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2021.
- [115] Handing Wang and Yaochu Jin. A random forest-assisted evolutionary algorithm for data-driven constrained multiobjective combinatorial optimization of trauma systems. *IEEE Transactions on Cybernetics*, 50(2):536–549, 2020.

-
- [116] James J. Q. Yu, Wen Yu, and Jiatao Gu. Online vehicle routing with neural combinatorial optimization and deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3806–3817, 2019.
- [117] Thomas Barrett, William Clements, Jakob Foerster, and Alex Lvovsky. Exploratory combinatorial optimization with reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3243–3250, 2020.
- [118] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research*, page 105400, 2021.
- [119] Tiande Guo, Congying Han, Siqi Tang, and Man Ding. Solving combinatorial problems with machine learning methods. In *Nonlinear Combinatorial Optimization*, pages 207–229. Springer, 2019.
- [120] Bryan Wilder, Eric Ewing, Bistra Dilkina, and Milind Tambe. End to end learning and optimization on graphs. *Advances in Neural Information Processing Systems*, 32:4672–4683, 2019.
- [121] Marin Vlastelica Pogančić, Anselm Paulus, Vit Musil, Georg Martius, and Michal Ročlinek. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2019.
- [122] Huangke Chen, Ran Cheng, Jinming Wen, Haifeng Li, and Jian Weng. Solving large-scale many-objective optimization problems by covariance matrix adaptation evolution strategy with scalable small subpopulations. *Information Sciences*, 509:457–469, 2020.
- [123] Ruochen Liu, Rui Ren, Jin Liu, and Jing Liu. A clustering and dimensionality reduction based evolutionary algorithm for large-scale multi-objective problems. *Applied Soft Computing*, 89:106120, 2020.

- [124] Ye Tian, Chang Lu, Xingyi Zhang, Kay Chen Tan, and Yaochu Jin. Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks. *IEEE Transactions on Cybernetics*, 51(6):3115–3128, 2021.
- [125] Hillermeier and Claus. Nonlinear multiobjective optimization: A generalized homotopy approach. *Journal of the Operational Research Society*, 10.1007/978-3-0348-8280-4(2):246–247, 2001.
- [126] S. Mardle and K. M. Miettinen. Nonlinear multiobjective optimization. *Journal of the Operational Research Society*, 51(2):246, 1999.
- [127] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 597–613, Cham, 2016. Springer International Publishing.
- [128] G. S. Xie, Z. Zhang, G. Liu, F. Zhu, L. Liu, L. Shao, and X. Li. Generalized zero-shot learning with multiple graph adaptive generative networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2021.
- [129] Mahyar Khayatkhoei, Maneesh K. Singh, and Ahmed Elgammal. Disconnected manifold learning for generative adversarial networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7343–7353. Curran Associates, Inc., 2018.
- [130] Y. Shen, J. Gu, X. Tang, and B. Zhou. Interpreting the latent space of gans for semantic face editing. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9240–9249, 2020.
- [131] C. Wang, C. Xu, X. Yao, and D. Tao. Evolutionary generative adversarial networks. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2019.

-
- [132] Alec Radford and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *Computer Science*, 13(2):284–302, 2015.
- [133] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584. IEEE, 2015.
- [134] K. He and J. Sun. Convolutional neural networks at constrained time cost. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5353–5360, June 2015.
- [135] Q. Zhang, A. Zhou, and Y. Jin. Rm-meda: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1):41–63, Feb 2008.
- [136] Y. Tian, R. Cheng, X. Zhang, and Y. Jin. Platemo: A matlab platform for evolutionary multi-objective optimization [educational forum]. *IEEE Computational Intelligence Magazine*, 12(4):73–87, Nov 2017.
- [137] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In Marc Schoenauer, Kalyanmoy Deb, Günther Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature PPSN VI*, pages 849–858, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
- [138] H. Li and Q. Zhang. Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302, 2009.

- [139] Xingyi Zhang, Xiutao Zheng, Ran Cheng, Jianfeng Qiu, and Yaochu Jin. A competitive mechanism based multi-objective particle swarm optimizer with fast convergence. *Information Sciences*, 427:63 – 76, 2018.
- [140] Ran Cheng, Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. Test problems for large-scale multiobjective and many-objective optimization. *IEEE Transactions on Cybernetics*, 47(12):4108–4121, 2017.
- [141] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.
- [142] G. Ismayilov and H. R. Topcuoglu. Dynamic multi-objective workflow scheduling for cloud computing based on evolutionary algorithms. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pages 103–108, Dec 2018.
- [143] Fangfang Zhang, Yi Mei, Su Nguyen, and Mengjie Zhang. Multitask multiobjective genetic programming for automated scheduling heuristic learning in dynamic flexible job-shop scheduling. *IEEE Transactions on Cybernetics*, 53(7):4473–4486, 2023.
- [144] Lu Huang, Hai-shan Chen, and Ting-ting Hu. Survey on resource allocation policy and job scheduling algorithms of cloud computing1. *J. Softw.*, 8(2):480–487, 2013.
- [145] Durdana Habib, Habibullah Jamal, and Shoab A Khan. Employing multiple unmanned aerial vehicles for co-operative path planning. *International Journal of Advanced Robotic Systems*, 10(5):235, 2013.
- [146] Nasser R. Sabar, Say Leng Goh, Ayad Turkey, and Graham Kendall. Population-based iterated local search approach for dynamic vehicle routing problems. *IEEE Transactions on Automation Science and Engineering*, pages 1–11, 2021.

-
- [147] Nasser R. Sabar, Ashish Bhaskar, Edward Chung, Ayad Turkey, and Andy Song. A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion. *Swarm and Evolutionary Computation*, 44:1018–1027, 2019.
- [148] Xiaoshu Xiang, Ye Tian, Xingyi Zhang, Jianhua Xiao, and Yaochu Jin. A pairwise proximity learning-based ant colony algorithm for dynamic vehicle routing problems. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2021.
- [149] Danial Yazdani, Ran Cheng, Donya Yazdani, Jürgen Branke, Yaochu Jin, and Xin Yao. A survey of evolutionary continuous dynamic optimization over two decades—part a. *IEEE Transactions on Evolutionary Computation*, 25(4):609–629, 2021.
- [150] Danial Yazdani, Ran Cheng, Cheng He, and Jürgen Branke. Adaptive control of subpopulations in evolutionary dynamic optimization. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
- [151] Masoud Kamosi, Ali B. Hashemi, and M.R. Meybodi. A hibernating multi-swarm optimization algorithm for dynamic environments. In *2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC)*, pages 363–369, 2010.
- [152] M. Plessis and A. Engelbrecht. Differential evolution for dynamic environments with unknown numbers of optima. *Journal of Global Optimization*, 55(1):73–99, 2013.
- [153] Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.
- [154] Shouyong Jiang, Shengxiang Yang, Xin Yao, Kay Chen Tan, Marcus Kaiser, and Natalio Krasnogor. Benchmark problems for cec2018 competition on dynamic multiobjective optimisation. In *Proc. CEC Competition*, pages 1–18, 2018.

- [155] Jianqiang Li, Tao Sun, Qiuzhen Lin, Min Jiang, and Kay Chen Tan. Reducing negative transfer learning via clustering for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 26(5):1102–1116, 2022.
- [156] Qingda Chen, Jinliang Ding, Tianyou Chai, and Quanke Pan. Evolutionary optimization under uncertainty: The strategies to handle varied constraints for fluid catalytic cracking operation. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
- [157] Ilker Kucukoglu, Reginald Dewil, and Dirk Cattrysse. The electric vehicle routing problem and its variations: A literature review. *Computers and Industrial Engineering*, 161:107650, 2021.
- [158] Fengxia Wang, Min Huang, Shengxiang Yang, and Xingwei Wang. Penalty and prediction methods for dynamic constrained multi-objective optimization. *Swarm and Evolutionary Computation*, 80:101317, 2023.
- [159] Abhishek Gupta, Yew-Soon Ong, and Liang Feng. Multifactorial evolution: Toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 20(3):343–357, 2016.
- [160] M. Farina, K. Deb, and P. Amato. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Transactions on Evolutionary Computation*, 8(5):425–442, 2004.
- [161] Sen Bong Gee, Kay Chen Tan, and Hussein A. Abbass. A benchmark test suite for dynamic evolutionary multiobjective optimization. *IEEE Transactions on Cybernetics*, 47(2):461–472, 2017.
- [162] Zhongwei Ma and Yong Wang. Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons. *IEEE Transactions on Evolutionary Computation*, 23(6):972–986, 2019.

-
- [163] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622, 2014.
- [164] Jia-Peng Li, Yong Wang, Shengxiang Yang, and Zixing Cai. A comparative study of constraint-handling techniques in evolutionary constrained multiobjective optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 4175–4182, 2016.
- [165] Guoyu Chen, Yinan Guo, Yong Wang, Jing Liang, Dunwei Gong, and Shengxiang Yang. Evolutionary dynamic constrained multiobjective optimization: Test suite and algorithm. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2023.
- [166] Ke Li, Renzhi Chen, Guangtao Fu, and Xin Yao. Two-archive evolutionary algorithm for constrained multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 23(2):303–315, 2019.
- [167] Abhishek Gupta, Yew-Soon Ong, and Liang Feng. Insights on transfer optimization: Because experience is the best teacher. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1):51–64, 2018.
- [168] Lu Bai, Wu Lin, Abhishek Gupta, and Yew-Soon Ong. From multitask gradient descent to gradient-free evolutionary multitasking: A proof of faster convergence. *IEEE Transactions on Cybernetics*, 52(8):8561–8573, 2022.
- [169] Abhishek Gupta, Jacek Mańdziuk, and Yew-Soon Ong. Evolutionary multitasking in bi-level optimization. *Complex & Intelligent Systems*, 1(1-4):83–95, 2015.
- [170] Liang Feng, Yuxiao Huang, Lei Zhou, Jinghui Zhong, Abhishek Gupta, Ke Tang, and Kay Chen Tan. Explicit evolutionary multitasking for combinatorial optimization: A case study on capacitated vehicle routing problem. *IEEE Transactions on Cybernetics*, pages 1–14, 2020.

- [171] Jun Yi, Junren Bai, Haibo He, Wei Zhou, and Lizhong Yao. A multifactorial evolutionary algorithm for multitasking under interval uncertainties. *IEEE Transactions on Evolutionary Computation*, 24(5):908–922, 2020.
- [172] Fangfang Zhang, Yi Mei, Su Nguyen, Mengjie Zhang, and Kay Chen Tan. Surrogate-assisted evolutionary multitask genetic programming for dynamic flexible job shop scheduling. *IEEE Transactions on Evolutionary Computation*, 25(4):651–665, 2021.
- [173] John Park, Yi Mei, Su Nguyen, Gang Chen, and Mengjie Zhang. Evolutionary multitask optimisation for dynamic job shop scheduling using niched genetic programming. In *AI 2018: Advances in Artificial Intelligence: 31st Australasian Joint Conference, Wellington, New Zealand, December 11-14, 2018, Proceedings 31*, pages 739–751. Springer, 2018.
- [174] Thiago Rios, Bas van Stein, Thomas Bäck, Bernhard Sendhoff, and Stefan Menzel. Multitask shape optimization using a 3-d point cloud autoencoder as unified representation. *IEEE Transactions on Evolutionary Computation*, 26(2):206–217, 2021.
- [175] Yue Wu, Yibo Liu, Maoguo Gong, Peiran Gong, Hao Li, Zedong Tang, Qiguang Miao, and Wenping Ma. Multi-view point cloud registration based on evolutionary multitasking with bi-channel knowledge sharing mechanism. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2022.
- [176] Chen Yang, Qunjian Chen, Zexuan Zhu, Zhi-An Huang, Shulin Lan, and Liehuang Zhu. Evolutionary multitasking for costly task offloading in mobile edge computing networks. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2023.
- [177] Cuie Yang, Jinliang Ding, Yaochu Jin, Chengzhi Wang, and Tianyou Chai. Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes. *IEEE Transactions on Automation Science and Engineering*, 16(3):1046–1057, 2019.
- [178] Zhongjian Wu, Wu Lin, Huimei Tang, and Qiuzhen Lin. A multiobjective multitask evolutionary algorithm based on decomposition and multivariate gaussian distribution.

-
- In *2021 IEEE 7th International Conference on Cloud Computing and Intelligent Systems (CCIS)*, pages 104–108, 2021.
- [179] Liang Feng, Lei Zhou, Abhishek Gupta, Jinghui Zhong, Zexuan Zhu, Kay-Chen Tan, and Kai Qin. Solving generalized vehicle routing problem with occasional drivers via evolutionary multitasking. *IEEE Transactions on Cybernetics*, pages 1–14, 2019.
- [180] Yuan Yuan, Yew-Soon Ong, Abhishek Gupta, Puay Siew Tan, and Hua Xu. Evolutionary multitasking in permutation-based combinatorial optimization problems: Realization with tsp, qap, lop, and jsp. In *2016 IEEE Region 10 Conference (TENCON)*, pages 3157–3164, 2016.
- [181] Ye Tian, Shichen Peng, Xingyi Zhang, Tobias Rodemann, Kay Chen Tan, and Yaochu Jin. A recommender system for metaheuristic algorithms for continuous optimization based on deep recurrent neural networks. *IEEE Transactions on Artificial Intelligence*, 1(1):5–18, 2020.
- [182] Abhishek Gupta, Lei Zhou, Yew-Soon Ong, Zefeng Chen, and Yaqing Hou. Half a dozen real-world applications of evolutionary multitasking, and more. *IEEE Computational Intelligence Magazine*, 17(2):49–66, 2022.
- [183] Ray Lim, Abhishek Gupta, Yew-Soon Ong, Liang Feng, and Allan N Zhang. Non-linear domain adaptation in transfer evolutionary optimization. *Cognitive Computation*, 13(2):290–307, 2021.
- [184] B. Cao, S. J. Pan, Y. Zhang, D. Yeung, and Q. Yang. Adaptive transfer learning. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.
- [185] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

- [186] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.
- [187] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.
- [188] Dawei Zhan and Huanlai Xing. A fast kriging-assisted evolutionary algorithm based on incremental learning. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2021.
- [189] Zhenkun Wang, Qingfu Zhang, Yew-Soon Ong, Shunyu Yao, Haitao Liu, and Jianping Luo. Choose appropriate subproblems for collaborative modeling in expensive multiobjective optimization. *IEEE Transactions on Cybernetics*, 53(1):483–496, 2021.
- [190] Kaiqin Xu, Leshi Shu, Linjun Zhong, Ping Jiang, and Qi Zhou. A bi-fidelity bayesian optimization method for multi-objective optimization with a novel acquisition function. *Structural and Multidisciplinary Optimization*, 66(3):53, 2023.
- [191] Abhishek Gupta. Evolutionary multitasking for single-objective continuous optimization: Benchmark problems, performance metric, and baseline results. *Nanyang Technological University, Technical Report*, 06 2017.
- [192] Fuhao Gao, Weifeng Gao, Lingling Huang, Jin Xie, Hong Li, and Maoguo Gong. Evolutionary multitasking optimization enhanced by geodesic flow kernel. *IEEE Transactions on Emerging Topics in Computational Intelligence*, pages 1–10, 2023.
- [193] Wu Lin, Qiuzhen Lin, Liang Feng, and Kay Chen Tan. Ensemble of domain adaptation-based knowledge transfer for evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 2023.
- [194] Cuie Yang, Jinliang Ding, Yaochu Jin, Chengzhi Wang, and Tianyou Chai. Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes. *IEEE Transactions on Automation Science and Engineering*, 16(3):1046–1057, 2018.

-
- [195] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys*, 54(6):1–35, 2021.
- [196] Mingyang Wan, Daochen Zha, Ninghao Liu, and Na Zou. In-processing modeling techniques for machine learning fairness: A survey. *ACM Transactions on Knowledge Discovery from Data*, 17(3):1–27, 2023.
- [197] Zhimeng Jiang, Xiaotian Han, Chao Fan, Fan Yang, Ali Mostafavi, and Xia Hu. Generalized demographic parity for group fairness. In *International Conference on Learning Representations*, 2022.
- [198] Guangyin Jin, Qi Wang, Cunchao Zhu, Yanghe Feng, Jincan Huang, and Jiangping Zhou. Addressing crime situation forecasting task with temporal graph convolutional neural network approach. In *2020 12th International Conference on Measuring Technology and Mechatronics Automation*, pages 474–478. IEEE, 2020.
- [199] Nan Wang, Lu Lin, Jundong Li, and Hongning Wang. Unbiased graph embedding with biased graph observations. In *Proceedings of the ACM Web Conference 2022, WWW '22*, page 1423–1433, New York, NY, USA, 2022. Association for Computing Machinery.
- [200] Enyan Dai and Suhang Wang. Learning fair graph neural networks with limited and private sensitive attribute information. *IEEE Transactions on Knowledge and Data Engineering*, 2022.
- [201] Jing Ma, Ruocheng Guo, Saumitra Mishra, Aidong Zhang, and Jundong Li. Clear: Generative counterfactual explanations on graphs. In *Advances in Neural Information Processing Systems*, 2022.
- [202] Yingqiang Ge, Juntao Tan, Yan Zhu, Yinglong Xia, Jiebo Luo, Shuchang Liu, Zuohui Fu, Shijie Geng, Zelong Li, and Yongfeng Zhang. Explainable fairness in recommenda-

- tion. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 681–691, 2022.
- [203] Zuohui Fu, Yikun Xian, Ruoyuan Gao, Jieyu Zhao, Qiaoying Huang, Yingqiang Ge, Shuyuan Xu, Shijie Geng, Chirag Shah, Yongfeng Zhang, et al. Fairness-aware explainable recommendation over knowledge graphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 69–78, 2020.
- [204] Wanyu Lin, Hao Lan, and Baochun Li. Generative causal explanations for graph neural networks. In *International Conference on Machine Learning*, pages 6666–6679. PMLR, 2021.
- [205] Romila Pradhan, Jiongli Zhu, Boris Glavic, and Babak Salimi. Interpretable data-based explanations for fairness debugging. *Proceedings of the 2022 ACM SIGMOD International Conference on Management of Data*, 2022.
- [206] Yushun Dong, Song Wang, Jing Ma, Ninghao Liu, and Jundong Li. Interpreting unfairness in graph neural networks via training node attribution. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.
- [207] Yushun Dong, Song Wang, Yu Wang, Tyler Derr, and Jundong Li. On structural explanation of bias in graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD ’22, page 316–326, New York, NY, USA, 2022. Association for Computing Machinery.
- [208] Pouria Rouzrokh, Bardia Khosravi, Shahriar Faghani, Mana Moassefi, Diana V Vera Garcia, Yashbir Singh, Kuan Zhang, Gian Marco Conte, and Bradley J Erickson. Mitigating bias in radiology machine learning: 1. data handling. *Radiology: Artificial Intelligence*, 4(5):e210290, 2022.

-
- [209] Wanyu Lin, Hao Lan, Hao Wang, and Baochun Li. Orphicx: A causality-inspired latent variable model for interpreting graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13729–13738, 2022.
- [210] Xun Zhou, A. K. Qin, Maoguo Gong, and Kay Chen Tan. A survey on evolutionary construction of deep neural networks. *IEEE Transactions on Evolutionary Computation*, 25(5):894–912, 2021.
- [211] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226, 2012.
- [212] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 29, 2016.
- [213] Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. Data decisions and theoretical implications when adversarially learning fair representations. *2017 Workshop on Fairness, Accountability, and Transparency in Machine Learning*, 2017.
- [214] Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.
- [215] Xiaoyu He, Zibin Zheng, Zefeng Chen, and Yuren Zhou. Adaptive evolution strategies for stochastic zeroth-order optimization. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1271–1285, 2022.
- [216] Kay Chen Tan, Eik Fun Khor, and Tong Heng Lee. *Multiobjective evolutionary algorithms and applications*. Springer Science & Business Media, 2005.
- [217] Tailin Wu, Hongyu Ren, Pan Li, and Jure Leskovec. Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33:20437–20448, 2020.

- [218] Matthew Jagielski, Giorgio Severi, Niklas Pousette Harger, and Alina Oprea. Subpopulation data poisoning attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 3104–3122, New York, NY, USA, 2021. Association for Computing Machinery.
- [219] Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 680–688, 2021.
- [220] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017.
- [221] Chao Zhang, Lei Zhang, and Jieping Ye. Generalization bounds for domain adaptation. *Advances in neural information processing systems*, 4:3320, 2012.
- [222] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.
- [223] Lubos Takac and Michal Zabovsky. Data analysis in public social networks. In *International scientific conference and international workshop present day trends of innovations*, volume 1. Present Day Trends of Innovations Lamza Poland, 2012.
- [224] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.
- [225] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *International Conference on Learning Representations*, 2017.

-
- [226] Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pages 2114–2124. PMLR, 2021.
- [227] Harrison Edwards and Amos Storkey. Censoring representations with an adversary. *The International Conference on Learning Representations*, 2015.
- [228] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018.
- [229] Avishek Bose and William Hamilton. Compositional fairness constraints for graph embeddings. In *International Conference on Machine Learning*, pages 715–724. PMLR, 2019.
- [230] Jing Ma, Ruocheng Guo, Mengting Wan, Longqi Yang, Aidong Zhang, and Jundong Li. Learning fair node representations with graph counterfactual fairness. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM ’22*, page 695–703, New York, NY, USA, 2022. Association for Computing Machinery.
- [231] Zhenzhong Wang, Zehui Lin, Wanyu Lin, Ming Yang, Minggang Zeng, and Kay Chen Tan. Explainable molecular property prediction: Aligning chemical concepts with predictions via language models. *arXiv preprint arXiv:2405.16041*, 2024.
- [232] Jesse B Zydallis, David A Van Veldhuizen, and Gary B Lamont. A statistical comparison of multiobjective evolutionary algorithms including the momga-ii. In *Evolutionary Multi-Criterion Optimization*, pages 226–240. Springer, 2001.
- [233] Xingyu Wu, Sheng-hao Wu, Jibin Wu, Liang Feng, and Kay Chen Tan. Evolutionary computation in the era of large language model: Survey and roadmap. *arXiv preprint arXiv:2401.10034*, 2024.

- [234] Beichen Huang, Ran Cheng, Zhuozhao Li, Yaochu Jin, and Kay Chen Tan. Evox: A distributed gpu-accelerated framework for scalable evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 2024.