mathematics

MDPI

Article

Reinforcement Learning Enhanced Multi-Objective Social Network Search Algorithm for Engineering Design Problems

Wei Peng [1], Zihan Li [2], Ji Li [2] and Guoqing Hu [1,*]

[1] School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510640, China
[2] School of Aeronautical Manufacturing and Mechanical Engineering, Nanchang Hangkong University, Nanchang 330063, China
* Correspondence: gqhu@scut.edu.cn

Abstract

To address real-world engineering design optimization problems, this study proposes a reinforcement learning enhanced multi-objective social network search algorithm (QMOSNS), which represents a novel approach for solving multi-objective optimization problems. QMOSNS utilizes Halton sequences for population initialization to enhance the diversity of the initial population. A multi-objective archive mechanism is implemented to store Pareto-optimal solutions and select parental individuals through a reassigned fitness evaluation strategy. Furthermore, Q-learning is incorporated to adaptively select mutation operators, thereby dynamically balancing the algorithm's exploration and exploitation capabilities. QMOSNS was rigorously evaluated through 50 prominent case studies, including 22 unconstrained multi-objective benchmark problems, 18 constrained multi-objective benchmark problems, and 10 multi-objective engineering design problems, to comprehensively validate its computational capabilities and effectiveness. Moreover, statistical results obtained using consistent performance metrics were compared with those of other highly regarded algorithms to ensure a fair and objective performance assessment. The comparative results show that QMOSNS is robust and superior in handling a wide variety of multi-objective problems. This study underscores the efficacy of integrating reinforcement learning with social intelligence for tackling complex multi-objective optimization in engineering and computational domains.

Keywords: multi-objective optimization; social network search; pareto optimal solution; Q-learning; initialization

MSC: 68W50; 65K10; 49K30

check for updates

Academic Editor: Hsien-Chung Wu

Received: 8 October 2025
Revised: 5 November 2025
Accepted: 10 November 2025
Published: 11 November 2025

Citation: Peng, W.; Li, Z.; Li, J.; Hu, G. Reinforcement Learning Enhanced Multi-Objective Social Network Search Algorithm for Engineering Design Problems. *Mathematics* **2025**, *13*, 3613. https://doi.org/10.3390/math13223613

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Multi-objective optimization (MOO) is a subfield of multi-criteria decision making that deals with mathematical problems involving the simultaneous optimization of multiple objective functions. Unlike single-objective optimization problems that seek a single optimal solution based on a single criterion, MOO requires considering and balancing two or more conflicting objectives [1]. MOO is of significant practical importance in real-world applications, spanning disciplines such as economics, science, and engineering [2–4]. It provides a systematic framework to support decision-makers in navigating complex trade-off environments and identifying optimal solutions that satisfy multiple competing

requirements. The task of MOO algorithms is to find the best trade-off solution set among multiple objectives, known as the Pareto optimal set [5].

Generally, the use of stochastic optimization algorithms to address multi-objective optimization problems (MOPs) can be categorized into two main classes: priori and posterior [6]. A priori method primarily involves transforming MOPs into new single-objective optimization problems through a set of weighting coefficients. Single-objective optimization algorithms can then readily solve this transformed problem after objective aggregation. Typically, domain experts determine these weights based on empirical knowledge, adjusting the weighting of each objective according to their relative importance [7]. However, this approach has notable limitations: Firstly, weight determination often relies on the decision-maker's subjective judgment, lacking unified objective criteria. Different decision-makers varying perceptions of objective importance may lead to inconsistent weight assignments. Furthermore, its weighted aggregation approach makes it difficult to optimize all objectives at once. Consequently, a single run cannot obtain the complete Pareto-optimal set [8]. In a posteriori approach, multiple objectives are simultaneously preserved and optimized, allowing for the acquisition of the complete Pareto optimal solution set in a single optimization run. The Pareto optimal set provides decision-makers with a wide range of options, enabling them to make informed choices among multiple solutions based on specific requirements [9]. This highlights the importance of solution diversity in MOO. Moreover, this approach can effectively handle any type of Pareto front. The algorithms based on a posteriori method are highly prevalent in the literature [10,11].

Moreover, posterior methods can be classified into three major categories: Multi-Objective Evolutionary Algorithms (MOEA) and Multi-Objective Swarm Algorithms (MOSA). In 1984, David Schaffer introduced the Vector Evaluated Genetic Algorithm (VEGA) by extending the original Genetic Algorithm (GA) to multi-objective problems [12]. This work marked the inception of evolutionary algorithms in multi-objective optimization contexts, and VEGA is widely recognized as the first Pareto-based MOEA. The core principle underlying most MOEAs is non-dominated sorting (NDS), which was adopted by the original Non-dominated Sorting Genetic Algorithm (NSGA) [13]. However, NSGA suffered from several limitations, including the absence of an explicit sharing parameter, lack of elitism consideration, and high computational cost of non-dominated sorting. To address these issues, its successor Non-dominated Sorting Genetic Algorithm II (NSGA-II) introduced three key innovations: a fast non-dominated sorting approach, a diversity preservation mechanism, and a crowded-comparison operator [14]. These improvements established NSGA-II as one of the most widely used and influential MOEAs globally. Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) represent another significant advancement in the field of MOEA [15]. This class of algorithms breaks down a multi-objective problem into separate subproblems. It then uses aggregation strategies to share information between neighboring subproblems, which optimizes computational efficiency. However, such methods may fail when addressing problems with discontinuous Pareto fronts and exhibit certain limitations in solving complex MOPs. In the domain of MOSAs, Coello Coello & Lechuga proposed the Multi-Objective Particle Swarm Optimization (MOPSO) algorithm by extending the conventional Particle Swarm Optimization (PSO) framework [16]. The approach incorporates mechanisms such as multi-objective function handling, mutation strategies, and a grid-based technique to enhance the distribution of solutions along the Pareto front. MOSAs represent a class of optimization techniques inspired by collective behaviors observed in human societies, animal populations, and physical phenomena [17]. Since the introduction of MOPSO, numerous novels and effective MOSAs have been continuously developed. Notable examples include the Multi-objective Grey Wolf Optimizer (MOGWO) [18], Multi-objective Water Cycle Algorithm (MOWCA) [19],

Decomposition-Based Multi-Objective Symbiotic Organism Search (MOSOS/D) [20], Multi-objective Artificial Vultures Optimization Algorithm (MOAVOA) [21], Multi-objective Group Learning Algorithm (MOGLA) [22], Multi-objective Stochastic Paint Optimizer (MOSPO) [23], Multi-objective Teaching-Learning-Based Optimizer (MOTLBO) [24], and Multi-objective Sine-Cosine Algorithm (MOSCA) [25]. These algorithms have made significant contributions to addressing complex MOPs by providing efficient and diverse solutions.

In MOO algorithms, parameter selection plays a critical role in modulating the balance between exploration and exploitation capabilities of evolutionary operators [26]. For instance, in the NSGA-II, the crossover and mutation probabilities directly affect convergence behavior (exploitation). Similarly, in MOEA/D, the size of the neighborhood and the aggregation function parameters determine the balance between global search and local intensification. Moreover, distinct operator types inherently possess characteristic exploration-exploitation properties [27]. Differential evolution operators are highly exploratory, using stochastic vectors to probe the search space. In contrast, simulated binary crossover and polynomial mutation are more exploitative, refining solutions through local search. These characteristics are fundamentally influenced by their respective mechanistic structures and interaction modes with the solution space. Therefore, it is crucial to intelligently adjust algorithm parameters or combine different operators during optimization. This is key to maintaining an effective balance between exploration and exploitation. This adaptive approach enables dynamic adjustment to problem characteristics and enhances the algorithm's ability to escape local optima while preserving convergence efficiency [28]. Reinforcement Learning (RL) is a machine learning paradigm focused on training agents to make optimal sequential decisions through interaction with an unknown environment [29]. In this learning process, an agent tries different actions and adjusts its strategy according to the results of these actions. Recent years have seen many studies successfully harness Reinforcement Learning (RL) to control evolutionary algorithms. By enabling real-time adaptation of parameters or on-the-fly selection of operators, RL helps these algorithms dynamically balance exploration and exploitation [30–32]. This hybrid approach has demonstrated significant improvements in solving complex optimization problems, as evidenced by enhanced convergence properties and solution diversity across various benchmark and real-world problems [33].

Although numerous MOO algorithms have been proposed to address MOPs, challenges such as slow convergence speed and susceptibility to local optima persist [34]. These algorithms aim to find a set of Pareto-optimal solutions that are evenly distributed. This even distribution provides a comprehensive range of high-quality choices for decision making. According to the No-Free-Lunch (NFL) theorem, no single algorithm can solve all optimization problems [35]. This fundamental principle underscores the persistent need to develop generalized MOO algorithms capable of effectively addressing diverse types of MOPs.

Social Network Search algorithm (SNS) is a metaheuristic algorithm proposed in 2021 for solving global optimization problems, based on human behavior [36]. The core innovation of the SNS algorithm lies in its guidance of the search process by simulating four distinct behavioral modes of social network users. It features a simple structure, high hierarchical organization, no need for manual parameter tuning, and fast convergence. SNS has demonstrated remarkable effectiveness in addressing diverse engineering problems, representing a novel and promising methodology with substantial potential for further research and development [37,38]. Building upon this foundation, this paper proposes a Reinforcement Learning Enhanced Multi-Objective Social Network Search Algorithm

(QMOSNS) for Engineering Design Problems. The primary contributions of this research work can be expressed as follows:

- The proposed QMOSNS enhances population diversity through Halton sequence initialization and maintains a multi-objective archive for storing Pareto-optimal solutions and selecting parent individuals. Moreover, Q-learning is integrated to dynamically adjust user behavior modes within the SNS framework, improving the balance between exploration and exploitation;
- The performance of QMOSNS was rigorously evaluated on a suite of test problems, including constrained and unconstrained benchmarks as well as engineering design problems. The algorithm was compared against state-of-the-art MOO methods using two performance metrics, with the results validated by statistical tests, robustness analysis, and Pareto front visualizations;
- Analysis shows that QMOSNS successfully handles MOPs, achieving well-balanced Pareto fronts across diverse benchmarks and engineering applications. Comparative results confirm its highly competitive performance, outperforming other algorithms in most cases.

The paper is structured as follows: Section 2 defines definitions related to this research. Section 3 details the original SNS algorithm, followed by Section 4′s presentation of QMOSNS. Section 5 validates the proposed algorithm through extensive experiments on 50 test problems and provides a detailed analysis of the results. Lastly, Section 6 conclude with a discussion of future research directions.

2. Background

2.1. Multi-Objective Optimization

Typically, an optimization problem only requires finding a single solution for a single objective function. However, in many real-world applications, we often face situations where multiple conflicting objectives need to be addressed simultaneously [39]. Taking minimization problems as an example, a multi-objective optimization problem can be formulated as follows:

$$\begin{aligned} &\min \mathrm{F}(x) = \min[f_1(x), f_2(x), \cdots, f_m(x)]^T \in \mathbb{F} \\ &s.t. \begin{cases} g_j(\mathrm{x}) \leq 0, j = 1, \cdots, l \\ h_k(\mathrm{x}) = 0, k = 1, \cdots, n \end{cases} \\ &x = (\mathrm{x}_1, x_2, \cdots, x_d)^T \in \mathbb{S} \end{aligned} \tag{1}$$

where x is a d-dimensional decision vector, $\mathbb{S}$ is the decision space, $F(x)$ consists of m objective functions, $\mathbb{F}$ is the objective space, $f_i(x)$ is the i-th objective function, $g_j(x)$ is the j-th inequality constraint, $h_k(x)$ is the k-th equality constraint, and l and n represent the number of inequality constraints and equality constraints, respectively.

In single-objective optimization, the quality of a solution can be determined simply by comparing its fitness value. However, when comparing solutions for problems with multiple objectives, traditional comparison methods fail to evaluate multiple criteria effectively. To address this issue, most multi-objective algorithms incorporate the concept of dominance in their comparison mechanisms to handle multiple objectives and identify Pareto optimal solutions. Within this context, the relevant concepts in multi-objective optimization are defined as follows.

2.1.1. Pareto Dominance

Suppose $x^1 = (x_1^1, x_2^1, \cdots, x_n^1)$ and $x^2 = (x_1^2, x_2^2, \cdots, x_n^2)$ are two feasible solutions in the objective space that satisfy all constraints. Solution x^1 is said to dominate solution x^2 (denoted as $x^1 \prec x^2$) if and only if the following conditions are met:

$$\forall\, i \in \{1, 2, \cdots, m\} : f_i(x^1) \leq f_i(x^2) \wedge \exists\, j \in \{1, 2, \cdots, m\} : f_j(x^1) < f_j(x^2)$$

2.1.2. Pareto Optimality

A solution x^* is defined as a Pareto optimal solution if it satisfies all constraints within the feasible region Ω and there exists no other solution in Ω that dominates x^*.

$$x^* \in \Omega \wedge \nexists x \in \Omega : x \prec x^*$$

2.1.3. Pareto Optimal Set

The set composed of all Pareto optimal solutions is called the Pareto Optimal Set (PS), defined as follows:

$$PS = \{x^* | x^* \in \Omega \wedge \nexists x \in \Omega : x \prec x^*\}$$

2.1.4. Pareto Optimal Front

The Pareto Optimal Front (PF) represents the projection of all Pareto optimal solutions x^* in the PS onto the objective space $\mathbb{F}$, formally defined as:

$$PF = \{f(x^*) \in \mathbb{F} | x^* \in PS\}$$

2.2. Multi-Objective Performance Metrics

Unlike the evaluation of single-objective optimization algorithms, multi-objective optimization has two primary goals: finding a set of Pareto optimal solutions that satisfy the constraints and ensuring the diversity of solutions within the Pareto optimal set. Consequently, a comprehensive set of performance metrics is required to adequately assess the performance of multi-objective optimization algorithms. The following subsection details the metrics employed in this study.

2.2.1. Inverted Generational Distance

The Inverted Generational Distance (IGD) is a metric used to evaluate the comprehensive performance of multi-objective optimization algorithms [40]. It measures algorithm performance by computing the average distance from points on the true Pareto front to the solution set generated by the algorithm. A smaller IGD value indicates that the generated solution set is closer to the true Pareto front and exhibits better distribution uniformity. The definition of IGD is as follows:

$$\begin{aligned} IGD(\mathrm{P}^*, A) &= \frac{\sum_{y^* \in P^*} d(y^*, A)}{|P^*|} \\ d(y^*, A) &= \min_{y \in A} \{ \sqrt{\sum_{i=1}^{m} (y_i^* - y_i)^2} \} \end{aligned} \tag{2}$$

where P^* represents a set of representative solutions on the true Pareto front, A denotes the approximate PF obtained by the algorithm, and m indicates the number of objectives.

2.2.2. Hypervolume

The Hypervolume (HV) is another key metric for evaluating the comprehensive performance of multi-objective optimization algorithms [41]. Unlike IGD, which requires prior knowledge of the true Pareto front, HV does not necessitate information about the true Pareto front for computation. It quantifies solution set quality by measuring the hypervolume enclosed between the solution set and a predefined reference point. A larger HV value indicates superior overall performance of the solution set, reflecting broader coverage and more uniform distribution in the objective space. The formal definition is as follows:

$$HV(S) = VOL(\underset{x \in S}{\cup}[f_1(x), z_1^r] \times \ldots \times [f_m(x), z_m^r]) \tag{3}$$

where $VOL(.)$ denotes the Lebesgue measure, m represents the number of objectives, and $z^r = (z_1^r, \ldots, z_m^r)^T$ is a user-defined reference point in the objective space.

3. Social Network Search Algorithm

With the advancement of Internet technology, social networks have emerged, significantly enhancing the efficiency of interpersonal communication. The Social Network Search (SNS) algorithm draws inspiration from user interactions within these platforms. In social networks, users frequently influence others' opinions by sharing their perspectives, insights, and ideas—a process that can be conceptualized as an optimization mechanism, as users inherently seek to maximize their influence and popularity. The SNS algorithm generates novel solutions to optimization problems by emulating four distinct behavioral modes observed in social network users: imitation, conversation, disputation, and innovation. The mathematical modeling and interpretation of these behavioral modes are elaborated as follows [36].

3.1. Imitation Mood

In this mood, social network users often try to emulate the popular views of other users to get their views noticed. This mood is mathematically formulated in Equation (4):

$$\begin{aligned} &x_{inew} = x_j + rand(-1, 1) \times R \\ &R = rand(0, 1) \times r \\ &r = x_i - x_j \end{aligned} \tag{4}$$

where x_i and x_j represent the i-th and j-th of the current users, respectively, and i is not equal to j. $rand(-1, 1)$ and $rand(0, 1)$ represent random vectors in the range $[-1, 1]$ and $[0, 1]$, respectively. Moreover, R represents the shock radius of the imitation space and r represents the popular radius of user j.

3.2. Conversation Mood

In this mood, users will interact with each other and discuss different issues to generate new perspectives. This mood can be mathematically formulated as follows:

$$\begin{aligned} &x_{inew} = x_k + R, \\ &\mathrm{R} = rand(0, 1) \times \mathrm{D} \\ &D = sign(f_i - f_j) \times (x_j - x_i) \end{aligned} \tag{5}$$

where x_k and x_j are two randomly selected users. D represents the difference in users' views, where X_i represents the viewpoint of the current i-th user, and i, j, and k are different. Moreover, the expression $(sign(f_i - f_{\mathrm{j}}))$ describes the direction of motion of x_k. It is worth noting that the user's viewpoints are sent to change through the conversation with user j.

3.3. Disputation Mood

In this mood, the user explains his or her view of the issue to others and defends his or her point of view. In this case, the user sees the different views of others and may be influenced by them. The new affected view in disputation is as follows:

$$\begin{aligned} x_{inew} &= x_i + rand(0,1) \times (M - AF \times x_i) \\ M &= \frac{\sum_t^{N_r} x_t}{N_r} \\ AF &= 1 + round(rand) \end{aligned} \tag{6}$$

where M is the average of the views of the users involved in the discussion, and AF is the admitting effect that represents the statement of the user's viewpoints during the discussion and takes the value integer 1 or 2. The symbol (*round*) stands for taking the input value to the closest integer value and (N_r) is the number of people involved in the discussion.

3.4. Innovation Mood

In this mood, users share their personal beliefs and experiences in approaching an issue in a creative way with others, and therefore, new perspectives are presented. The mathematical explanation of innovation is as follows:

$$\begin{aligned} x_{inew}^{\mathrm{d}} &= t x_j^d + (1-t) \times n_{new}^d \\ n_{new}^d &= LB^d + rand(0,1) \times (UB^d - LB^d) \\ t &= rand(0,1) \end{aligned} \tag{7}$$

where d represents the d-th variable in the user's viewpoint; UB^d and LB^d represent the upper and lower bounds of the d-th dimensional variable; n_{new}^d is the new thought in the d-th dimension of the viewpoint; and x_j^d represents the d-th dimension of the j-th user's viewpoint where i is not equal to j. x_{inew}^d consists of the pre-existing thought (n_j^d) and the new thought (n_{new}^d).A change in the d-th dimension of the user's viewpoint also means that the user's viewpoint on the event has changed.

3.5. Network Rules

There are rules defined in social networks that users need to follow when sharing their views. The rules that restrict the user's views are formulated as follows:

$$\begin{aligned} x_{inew}^d &= \min(x_{inew}^d, UB^d) \\ x_{inew}^d &= \max(x_{inew}^d, LB^d) \\ d &= 1, 2, 3, \cdots, view's\ dim \\ i &= 1, 2, 3, \cdots, user's\ number \end{aligned} \tag{8}$$

During the next iteration of the algorithm, one of the four moods will be randomly selected to update the user's view, and the new view generated will need to adhere to the rules of the social network. The main process of SNS is shown in Figure 1.

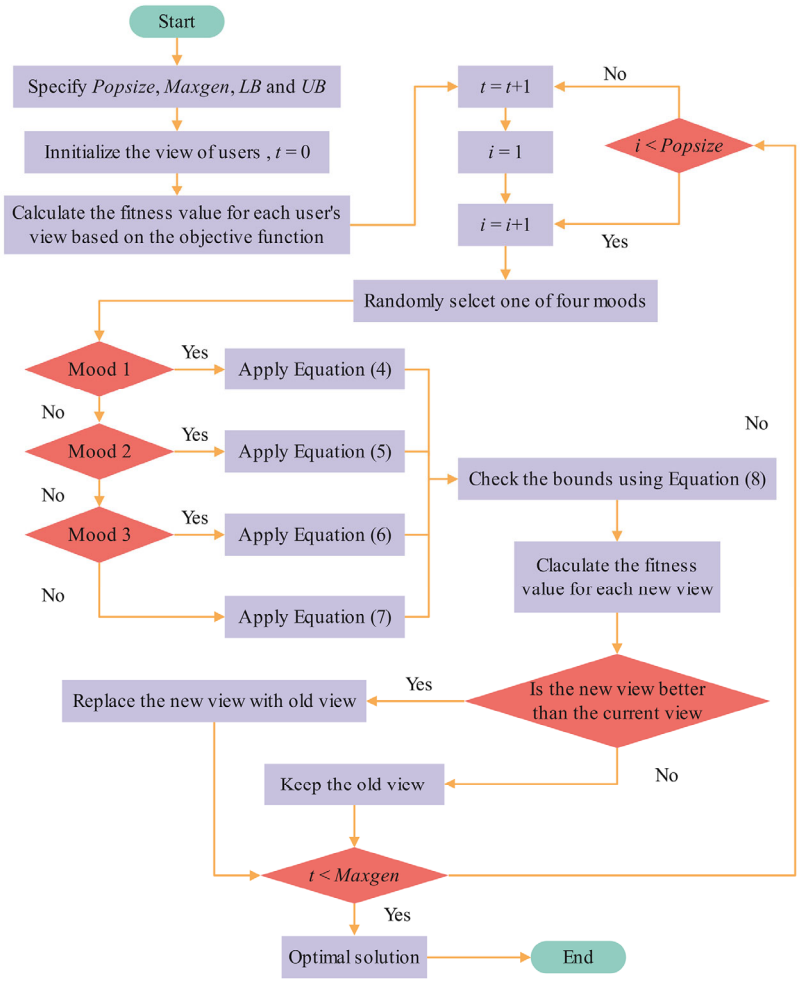


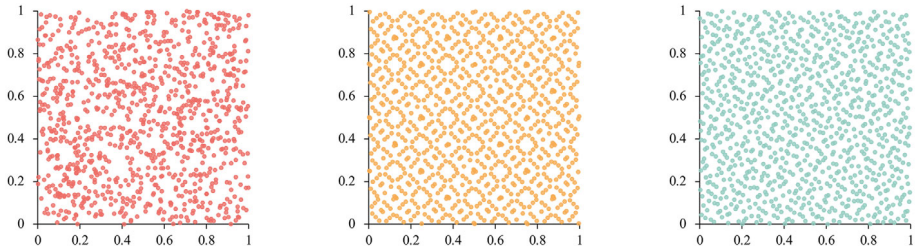
Figure 1. The flowchart of the SNS algorithm.

4. The Proposed QMOSNS

4.1. Population Initialization Based on Halton Sequence

The population initialization method significantly impacts both the iterative process and the final results of an algorithm. A uniformly distributed initial population enhances the exploration capability of optimization algorithms and improves search efficiency [42]. However, the original SNS algorithm lacks mechanisms for generating uniform initial populations, relying solely on random initialization methods. This often results in poor coverage uniformity across the solution space, leading to insufficient population diversity during early optimization stages and inefficient acquisition of solution space information. Existing literature demonstrates that low-discrepancy sequences like the Halton sequence serve as effective initialization mechanisms for metaheuristic algorithms, as they tend to uniformly cover the search space to facilitate optimal solution discovery [43]. Therefore, this study incorporates the Halton sequence into the initialization phase. By leveraging the deterministic characteristics of the Halton sequence to generate populations instead of random initialization, the proposed method achieves more comprehensive coverage of the solution space compared to traditional random approaches. As shown in Figure 2, a population of 1000 two-dimensional individuals was generated using random sampling, the Sobol sequence, and the Halton sequence method, with each dimension constrained to the interval [0, 1]. It can be seen that the Halton sequence achieved the best performance. The use of Halton sequences for initial population generation effectively prevents clustering phenomena in initial populations, thereby improving population diversity and uniformity.

Such enhancements maintain strong exploration capability during early optimization stages and establish a solid foundation for global optimization.



(**a**) Random (**b**) Sobol (**c**) Halton

Figure 2. Comparison of the distribution of the initial population.

4.2. Multi-Objective Archive

The QMOSNS algorithm is developed based on the SNS algorithm described earlier. In the SNS algorithm, the quality of a solution is evaluated according to its fitness value for a single objective, thereby enabling the selection of the globally optimal solution. However, in multi-objective optimization, solutions possess fitness values across multiple objectives. Their quality cannot be determined simply by directly comparing these objective fitness values. Inspired by SPEA-II [44], this study adopts the same approach to reassign a new fitness value $Fit(X)$ to each multi-objective solution X, thereby evaluating the quality of solutions. The specific definition is as follows:

$$Fit(x) = \sum_{y \in S \wedge y \neq x} |R_x| + \frac{1}{dist(x, x') + 2} \tag{9}$$

where R_x is the set of solutions dominated by the solution x, S is the set of solutions dominating the solution x and x' denotes the nearest neighbor to x. The first component of Equation (9) represents the total number of solutions dominated by x in the solution set, while the second component corresponds to the inverse of the Euclidean distance between x and its nearest neighbor x'. These two components, respectively, assess convergence and diversity performance. A smaller $Fit(x)$ indicates better solution quality of x. Specifically, when $Fit(x) < 1$, it signifies that x is non-dominated.

The utilization of external archives is prevalent in multi-objective optimization algorithms [45]. In this algorithm, an external archive A is likewise adopted to preserve candidate solutions. The external archive A maintains a fixed population size N of solutions, which serve as parent individuals for generating offspring in the subsequent generation. During each evolutionary iteration, both offspring individuals and solutions in archive A are evaluated and assigned fitness $Fit(x)$, which is then used to update the archive. The specific archive update mechanism is implemented as follows:

- If the number of solutions satisfying $Fit(x) < 1$ is precisely equal to N, these solutions are stored in the A;
- If the number of solutions satisfying $Fit(x) < 1$ exceeds N, the top N solutions are selected based on $Fit(x)$ and stored in the A;
- If the number of solutions satisfying $Fit(x) < 1$ is less than N, then the k compliant solutions are stored in the A, and the remaining (N – k) solutions are selected from the top-ranked candidates based on $Fit(x)$ and stored in the A.

4.3. Q-Learning-Based Mood Selection Strategy

The SNS algorithm randomly selects the moods model and the user's optimization mechanism, which is conducive to improving the algorithm's global exploration ability and

avoiding the algorithm's premature convergence in the process of optimization, but it also reduces the algorithm's optimization accuracy and efficiency to a certain extent. Applying reinforcement learning to social network search algorithms helps to help algorithms rationally select sentiment models in the process of searching for excellence, in order to balance the algorithm's global exploration and local exploitation capabilities, and thus improve the efficiency and accuracy of searching for excellence.

The Q-learning algorithm is one of the most frequently used algorithms for RL, which consists of five parts: the intelligent body, the Q-table, the reward value R, the state set st, and the action set ac [46]. The Q-learning algorithm first perceives the state of the environment through the intelligent body, and then makes the optimal decision based on the value of the Q-table. After the decision is applied to the environment, the state of the intelligent body will be changed and the reward value for the decision will be fed back to the intelligent body from the environment to update the Q-table. The dynamic updating mechanism of the Q-table is as follows:

$$Q(st_t, ac_t) \leftarrow Q(st_t, ac_t) + \lambda[R_{t+1} + \gamma \text{max} Q(st_{t+1}, ac) - Q(s_t, ac_t)] \tag{10}$$

where λ represents the learning efficiency, γ represents the discount factor, R_{t+1} denotes the immediate reward obtained by executing the action ac_t in state st_t, and $Q(st_t, ac_t)$ represents the Q-table value of executing the action ac_t in state st_t. λ is crucial to the Q-learning algorithm, and the larger λ is, the greater the influence of the expected future reward on the computation of the future Q-value is; and the smaller λ is, the greater the influence of the current Q-value on the computation of the future Q-value is. Therefore, the learning effect of the intelligence in the search space can be improved by adaptively decreasing λ in the process of SNS iteration, and λ is calculated as follows:

$$\lambda = \frac{\lambda_{initial} + \lambda_{final}}{2} - \frac{\lambda_{initial} + \lambda_{final}}{2} \cos\left(\pi\left(1 - \frac{t}{Maxgen}\right)\right) \tag{11}$$

where $\lambda_{initial}$ and λ_{final} represent the initial and final values of learning efficiency, respectively.

In constrained multi-objective optimization methods, algorithm performance must simultaneously consider convergence and feasibility. The Hypervolume (HV) indicator serves as a comprehensive evaluation metric that assesses both the convergence and diversity of Pareto optimal solution sets by measuring the volume of the multidimensional space dominated by these solutions relative to a reference point [41]. The feasibility of the population is evaluated using both the constraint violation (CV) and the feasibility rate (FR). Notably, the changes in HV and CV (denoted ΔHV and ΔCV) between two consecutive generations can effectively reflect the efficacy of the evolutionary strategy. Based on this, four states can be defined according to the changes in HV and CV:

- State 1 (S_1): Both HV and CV improve simultaneously;
- State 2 (S_2): Only the HV improves;
- State 3 (S_3): Only the CV improves;
- State 4 (S_4): There is no improvement in either HV or CV.

The immediate reward of applying an action to a state, denoted as R, is determined by the improvement of the evolution. The reward value R varies as shown in Equation (12).

$$R = \begin{cases} 1 \quad , & \Delta HV > 0 \ \& \ \text{FR} = 1 \\ 0.5 \ , & \Delta HV > 0 \ \& \ \text{FR} \neq 1 \ \& \ \Delta CV \leq 0 \\ -1 \ , & \text{others} \end{cases} \tag{12}$$

When using the Q-learning algorithm in SNS algorithms, a Q-table needs to be created for each user. Given that the SNS incorporates four mood models and four states have been defined previously, the Q-table is designed as a matrix of 4 rows and 4 columns, where the rows represent states *st* and the columns represent actions *ac*. As shown in Figure 3, before the Q-table update, the agent selected the imitation mood model while in S_3, and the corresponding Q-value for this position was updated to 1.42 according to Equation (11). After the Q-table update, the current state transitions to S_1. Subsequently, a random number *r* will be generated, and if *r* is smaller than the set greedy threshold ε, the action corresponding to the maximum Q-value will be selected; otherwise, an action will be randomly chosen from the four actions. By following this cycle, the appropriate mood model for the current situation can be selected based on the Q value.

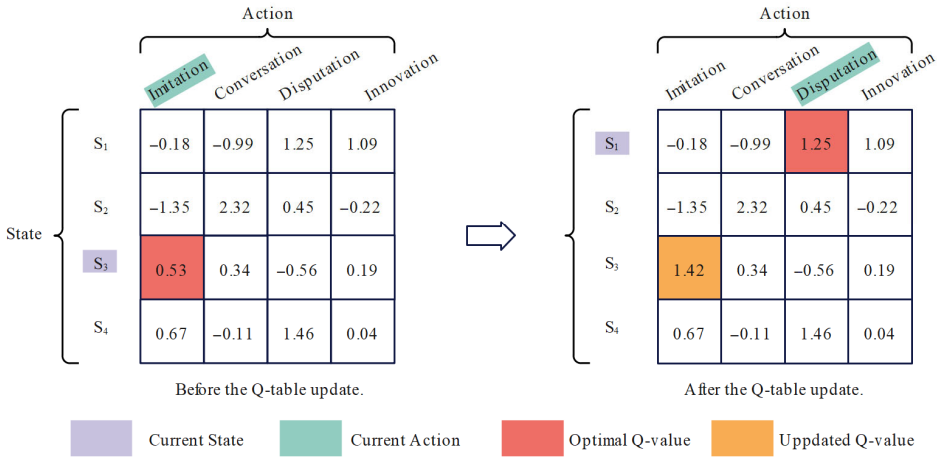


Figure 3. Operation of the Q-learning-based mood selection strategy.

The pseudo code of QMOSNS is presented in Algorithm 1.

Algorithm 1 QMOSNS Algorithm

```
Input: Popsize: Number of population, Maxgen: Maximum iterations, D: Dimension
       size, UB: Upper bound of a variable, LB: Lower bound of a variable.
Output: Return the archive.
Initialize the Q-table.
Initialize the population using Halton sequence.
Calculate the fitness value of each individual in the population.
Update the archive.
for t = 1: Maxgen
    if t = 1
        Randomly select a mood.
    else
        Select a mood using the Q-table.
    End if
    Generate offspring using the selected mood.
    Calculate the fitness value of each individual in the offspring.
    Update the archive and select parent solutions.
    Update the Q-table.
end for
```

4.4. Computational Complexity

The complexity of the proposed QMOSNS is based on the number of decision variables, the objective variables, and the population size. Imagine a multi-objective problem with D decision variables, M objective variables, and N particles as an example. At each generation, the QMOSNS mainly spends the computational cost for updating user's viewpoint, updating archive, and implementing the Q-learning-based emotion selection strategy. The complexity of update user's viewpoint is decided by N and D, so the computational complexity of this process is O(DN). During the archive update process, fitness assignment constitutes the most computationally complex part, with a computational complexity of O(MN^3). The Q-learning-based operator selection process consists of three steps: state determination, Q-table update, and optimal action selection. Their computational complexities are O(N^2), O(1), and O(1), respectively. To sum up, the total computational complexity of the proposed algorithm is O(MN^3), which is highly sensitive to increases in N and is a key factor determining the actual runtime. When N increases by a factor of k, the theoretical runtime grows by k^3 times. This implies that while the algorithm is feasible for small to medium-sized populations, it faces computational bottlenecks in ultra-large-scale scenarios.

5. Numerical Examples and Results

5.1. Experimental Setup

In this section, 50 well-established multi-objective benchmark problems and engineering design problems derived from authoritative studies are employed to rigorously evaluate the optimization performance of QMOSNS. These test problems incorporate objective functions with distinct characteristics and span varying dimensions of design variables, which are systematically classified into three primary categories:

- Unconstrained multi-objective problems: ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ5, DTLZ6, DTLZ7, UF1, UF2, UF3, UF4, UF5, UF6, UF7, UF8, UF9, UF10 [47–49];
- Constrained multi-objective problems: C1_DTLZ1, C1_DTLZ2, C2_DTLZ3, C2_DTLZ4, MW1, MW2, MW3, MW4, MW5, MW6, MW7, MW8, MW9, MW10, MW11, MW12, MW13, MW14 [50,51];
- Multi-objective engineering design problems: Pressure Vessel Design [52], Vibrating Platform Design [53], Welded Beam Design [54], Disc Brake Design [55], Car Side Impact Design [50], Four Bar Plane Truss [56], Multiple Disk Clutch Brake Design [57], Spring Design [52], Multi-product Batch Plant [58], Crash Energy Management for High-speed Train [59].

To validate the effectiveness of the proposed QMOSNS algorithm, comparative experiments were conducted against six state-of-the-art algorithms, including two widely used approaches (NSGA-II [14], MOEA/D [15]) and four advanced methods (ARMOEA [60], MOEADDAE [61], CMOES [62], PPS [63]). All experiments were performed in a computational environment equipped with an Intel(R) Core(TM) i5-12600KF CPU @3.70 GHz and 32.00 GB RAM. The operating system was Windows 11 Pro, and MATLAB R2023b served as the execution platform. The experimental studies were implemented on PlatEMO [64], with all algorithmic parameters maintained at their default settings in the platform.

Each algorithm is independently executed 30 times for each multi-objective optimization problem, and the statistical results are subjected to a Wilcoxon rank-sum test with a significance level of 0.05. The symbols "+", "−", and "=" denote that the algorithm performs better than, worse than, or equal to the compared algorithm, respectively. To ensure reliable and fair comparisons, the population size for all algorithms is set to 100. The maximum number of evaluations for the ZDT, DTLZ, UF, C_DTLZ, and MW test functions

is set to 25,000, 100,000, 100,000, 100,000, and 60,000, respectively. For multi-objective engineering optimization problems, the maximum number of evaluations is determined according to the following criteria (where m represents the number of objectives in the problem and d denotes the problem's dimensionality):

$$\begin{cases} 2 \times 10^4, \ if(m = 2)\&(d \leq 10) \\ 8 \times 10^4, \ elseif(m = 2)\&(d > 10) \\ 2.6250 \times 10^4, \ elseif(m = 3)\&(d \leq 10) \\ 1.05 \times 10^5, \ elseif(m = 3)\&(d > 10) \end{cases} \tag{13}$$

5.2. Multi-Objective Benchmark Problems

5.2.1. Unconstrained Multi-Objective Problems

Table 1 presents the final statistical results of the IGD metrics obtained by QMOSNS, NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES and PPS. The results indicate that in terms of both the average and standard deviation values of the IGD metric, QMOSNS significantly outperforms NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS on ZDT1, ZDT2, ZDT3, and ZDT6, only underperforming NSGA-II and MOEADDAE on ZDT4. This demonstrates that QMOSNS can achieve solutions closer to the true Pareto front on most test functions compared to the other algorithms. Table 2 shows the statistical results of the HV performance metrics for the DTLZ benchmark functions obtained by the QMOSNS algorithm and other multi-objective algorithms. The experimental results reveal that the QMOSNS algorithm outperforms the other algorithms on the ZDT1, ZDT2, and ZDT6 test functions. MOEADDAE achieves the best performance on ZDT3, while NSGA-II performs best on ZDT4, but QMOSNS still secures the second rank on these two test functions. Therefore, QMOSNS demonstrates superior convergence and diversity on the ZDT test problems. Figure 4 displays a graphical representation of the true Pareto front of the ZDT test suite and the Pareto fronts obtained by QMOSNS. All plots demonstrate that the Pareto optimal fronts obtained by QMOSNS exhibit uniform distribution along the true Pareto optimal fronts.

Table 1. Statistical Results of MOO Algorithms in Terms of IGD For ZDT Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
ZDT1	4.8007×10^{-3} (1.47×10^{-4}) −	7.0791×10^{-3} (1.06×10^{-3}) −	3.9528×10^{-3} (2.41×10^{-5}) −	4.8121×10^{-3} (2.25×10^{-4}) −	4.7270×10^{-3} (2.15×10^{-4}) −	2.0890×10^{-2} (7.40×10^{-3}) −	**3.9049×10^{-3} (3.89×10^{-5})**
ZDT2	5.0362×10^{-3} (2.23×10^{-4}) −	8.4050×10^{-3} (9.83×10^{-4}) −	4.0055×10^{-3} (4.51×10^{-4}) =	4.7507×10^{-3} (1.51×10^{-4}) −	4.7763×10^{-3} (2.66×10^{-4}) −	1.4206×10^{-2} (3.44×10^{-3}) −	**3.9016×10^{-3} (2.58×10^{-5})**
ZDT3	5.4060×10^{-3} (1.88×10^{-4}) −	1.5031×10^{-2} (6.45×10^{-3}) −	6.4548×10^{-3} (3.82×10^{-5}) −	6.3604×10^{-3} (5.34×10^{-3}) −	5.1903×10^{-3} (1.37×10^{-4}) −	3.6501×10^{-2} (1.99×10^{-2}) −	**4.8132×10^{-3} (1.01×10^{-4})**
ZDT4	**6.5004×10^{-3} (1.71×10^{-3}) +**	3.3404×10^{-2} (3.83×10^{-2}) −	1.0461×10^{-2} (9.27×10^{-3}) −	6.8525×10^{-3} (1.93×10^{-3}) +	8.1588×10^{-3} (3.02×10^{-3}) −	3.9894×10^{-1} (2.59×10^{-1}) −	7.8272×10^{-3} (1.50×10^{-2})
ZDT6	3.7976×10^{-3} (1.42×10^{-4}) −	1.6414×10^{-2} (3.69×10^{-3}) −	3.3904×10^{-3} (3.56×10^{-4}) −	4.1197×10^{-3} (3.53×10^{-4}) −	4.3082×10^{-3} (1.41×10^{-3}) −	5.0399×10^{-3} (4.64×10^{-4}) −	**3.1914×10^{-3} (1.05×10^{-4})**
+/−/=	1/4/0	0/5/0	0/4/1	1/4/0	0/5/0	0/5/0	

Table 2. Statistical Results of MOO Algorithms in Terms of HV For ZDT Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
ZDT1	7.1890×10^{-1} (2.35×10^{-4}) −	7.1401×10^{-1} (1.55×10^{-3}) −	7.1990×10^{-1} (1.08×10^{-4}) −	7.1898×10^{-1} (2.79×10^{-4}) −	7.1799×10^{-1} (4.48×10^{-4}) −	6.9475×10^{-1} (9.54×10^{-3}) −	**7.2028×10^{-1} (1.24×10^{-4})**
ZDT2	4.4341×10^{-1} (2.61×10^{-4}) −	4.3492×10^{-1} (1.83×10^{-3}) −	4.4421×10^{-1} (1.66×10^{-3}) −	4.4361×10^{-1} (2.79×10^{-4}) −	4.4219×10^{-1} (5.75×10^{-4}) −	4.2465×10^{-1} (6.44×10^{-3}) −	**4.4497×10^{-1} (9.26×10^{-5})**
ZDT3	5.9922×10^{-1} (1.20×10^{-4}) −	5.9586×10^{-1} (1.74×10^{-2}) −	5.9847×10^{-1} (1.66×10^{-4}) −	**6.0238×10^{-1} (1.62×10^{-2}) +**	5.9856×10^{-1} (3.01×10^{-4}) −	5.8569×10^{-1} (1.49×10^{-2}) −	5.9964×10^{-1} (1.52×10^{-4})
ZDT4	**7.1515×10^{-1} (2.76×10^{-3}) +**	6.7703×10^{-1} (5.13×10^{-2}) −	7.1151×10^{-1} (8.12×10^{-3}) −	7.1448×10^{-1} (3.06×10^{-3}) −	7.1270×10^{-1} (4.00×10^{-3}) −	3.0545×10^{-1} (1.75×10^{-1}) −	7.1499×10^{-1} (2.05×10^{-2})
ZDT6	3.8762×10^{-1} (4.61×10^{-4}) −	3.6811×10^{-1} (5.25×10^{-3}) −	3.8764×10^{-1} (7.94×10^{-4}) −	3.8669×10^{-1} (7.49×10^{-4}) −	3.8618×10^{-1} (2.37×10^{-3}) −	3.8675×10^{-1} (5.50×10^{-4}) −	**3.8881×10^{-1} (1.00×10^{-4})**
+/−/=	1/4/0	0/5/0	0/5/0	1/4/0	0/5/0	0/5/0	

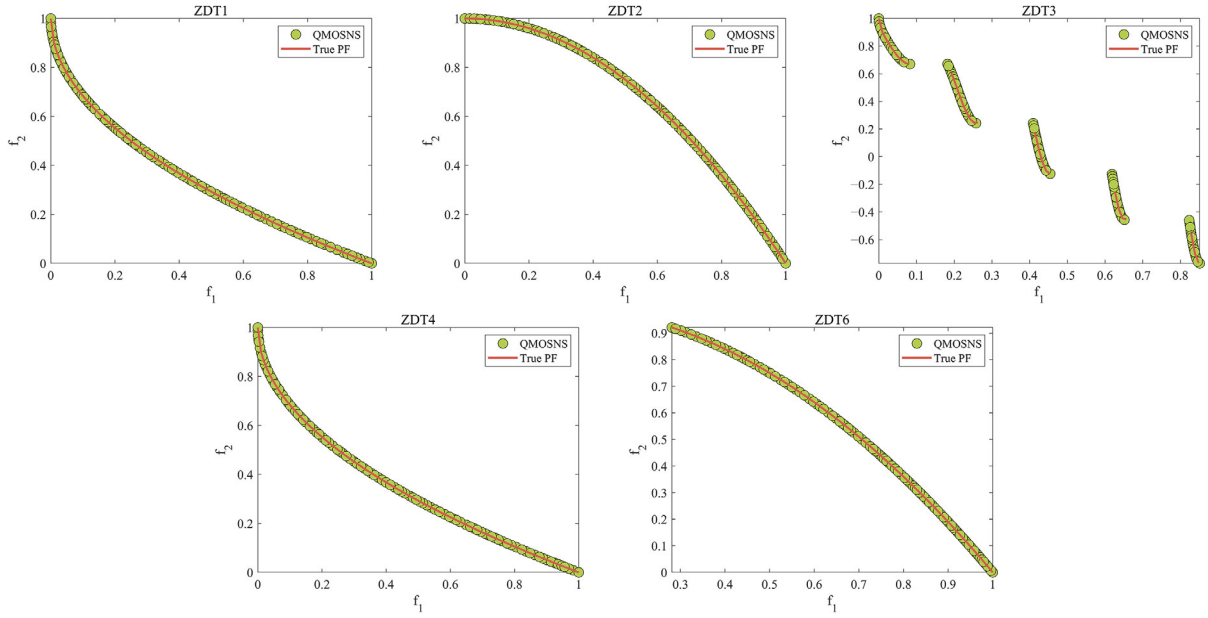


Figure 4. Pareto optimal fronts generated by QMOSNS for test functions ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6.

Tables 3 and 4 present the statistical results of IGD and HV obtained by QMOSNS, NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS. Overall, the IGD metric results indicate that QMOSNS outperforms all other compared algorithms across all test problems except DTLZ4, demonstrating the superior convergence performance of the proposed algorithm on these test functions. The significant differences in IGD values between QMOSNS and other algorithms further support this conclusion. Although QMOSNS shows slightly inferior performance to MOEA/D on DTLZ4 and performs comparably to ARMOEA, this does not compromise its overall effectiveness. Regarding the HV metric, QMOSNS achieves significantly better results than the other algorithms on four test functions: DTLZ2, DTLZ5, DTLZ6, and DTLZ7. MOEA/D delivers the best performance on DTLZ1, DTLZ3, and DTLZ4. However, QMOSNS remains highly competitive with MOEA/D on DTLZ1 and DTLZ3, only slightly underperforming on DTLZ3. In summary, QMOSNS still exhibits strong performance across the DTLZ test suite. Figure 5 illustrates the true Pareto front and the obtained Pareto fronts of all algorithms on the DTLZ test problems. The results in Tables 3 and 4 are visually confirmed by the graphical representations

in Figure 5, where it can be observed that the Pareto front generated by QMOSNS not only converges well to the true Pareto front but also maintains a uniform distribution.

Table 3. Statistical Results of MOO Algorithms in Terms of IGD For DTLZ Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
DTLZ1	2.7010×10^{-2} (1.04×10^{-3}) −	2.0563×10^{-2} (1.32×10^{-5}) −	2.0568×10^{-2} (2.21×10^{-5}) −	2.1404×10^{-2} (5.45×10^{-4}) −	2.0403×10^{-2} (1.99×10^{-4}) =	2.8786×10^{-2} (9.66×10^{-4}) −	**2.0382×10^{-2} (1.70×10^{-4})**
DTLZ2	6.8903×10^{-2} (2.80×10^{-3}) −	5.4464×10^{-2} (1.99×10^{-7}) −	5.4464×10^{-2} (6.50×10^{-6}) −	5.6688×10^{-2} (3.96×10^{-4}) −	5.5572×10^{-2} (6.70×10^{-4}) −	7.5507×10^{-2} (2.58×10^{-3}) −	**5.3918×10^{-2} (4.79×10^{-4})**
DTLZ3	6.9464×10^{-2} (2.82×10^{-3}) −	5.4598×10^{-2} (1.57×10^{-4}) −	5.4733×10^{-2} (3.85×10^{-4}) −	6.6398×10^{-2} (4.96×10^{-3}) −	5.4374×10^{-2} (8.44×10^{-4}) =	1.2166×10^{-1} (2.45×10^{-1}) −	**5.4170×10^{-2} (5.96×10^{-4})**
DTLZ4	9.6545×10^{-2} (1.60×10^{-1}) −	**5.4464×10^{-2} (2.16×10^{-6}) +**	3.0845×10^{-1} (2.93×10^{-1}) =	2.6884×10^{-1} (2.82×10^{-1}) −	5.5979×10^{-2} (9.59×10^{-4}) −	1.2650×10^{-1} (7.54×10^{-2}) −	5.5368×10^{-2} (9.92×10^{-4})
DTLZ5	5.7453×10^{-3} (2.68×10^{-4}) −	3.3923×10^{-2} (3.72×10^{-6}) −	5.4262×10^{-3} (9.78×10^{-5}) −	4.9348×10^{-3} (1.32×10^{-4}) −	4.9725×10^{-3} (2.05×10^{-4}) −	1.0745×10^{-2} (7.82×10^{-4}) −	**4.1164×10^{-3} (3.74×10^{-5})**
DTLZ6	5.9844×10^{-3} (4.38×10^{-4}) −	3.3929×10^{-2} (1.23×10^{-6}) −	5.0046×10^{-3} (6.22×10^{-5}) −	4.6426×10^{-3} (9.85×10^{-5}) −	4.1375×10^{-3} (3.12×10^{-5}) =	6.4126×10^{-3} (3.31×10^{-4}) −	**4.1217×10^{-3} (2.98×10^{-5})**
DTLZ7	8.5734×10^{-2} (5.06×10^{-2}) −	1.5470×10^{-1} (1.80×10^{-3}) −	2.0421×10^{-1} (1.78×10^{-1}) −	7.0888×10^{-2} (5.31×10^{-2}) −	6.2679×10^{-2} (1.62×10^{-3}) −	1.6311×10^{-1} (8.57×10^{-2}) −	**6.0120×10^{-2} (1.83×10^{-3})**
+/−/=	0/7/0	1/6/0	0/6/1	0/7/0	0/4/3	0/7/0	

Table 4. Statistical Results of MOO Algorithms in Terms of HV For DTLZ Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
DTLZ1	8.2451×10^{-1} (3.56×10^{-3}) −	**8.4165×10^{-1} (1.71×10^{-4}) =**	8.4158×10^{-1} (2.71×10^{-4}) =	8.4032×10^{-1} (1.67×10^{-3}) −	8.4151×10^{-1} (7.38×10^{-4}) =	8.0270×10^{-1} (5.47×10^{-3}) −	8.4163×10^{-1} (5.88×10^{-4})
DTLZ2	5.3254×10^{-1} (3.28×10^{-3}) −	5.5962×10^{-1} (2.16×10^{-6}) =	5.5960×10^{-1} (3.07×10^{-5}) =	5.5540×10^{-1} (1.23×10^{-3}) −	5.5119×10^{-1} (1.58×10^{-3}) −	5.1945×10^{-1} (4.19×10^{-3}) −	**5.5963×10^{-1} (1.73×10^{-3})**
DTLZ3	5.3014×10^{-1} (7.32×10^{-3}) −	**5.5696×10^{-1} (1.93×10^{-3}) =**	5.5573×10^{-1} (3.02×10^{-3}) =	5.5408×10^{-1} (2.39×10^{-3}) −	5.5584×10^{-1} (3.22×10^{-3}) =	4.9416×10^{-1} (9.38×10^{-2}) −	5.5673×10^{-1} (3.19×10^{-3})
DTLZ4	5.2057×10^{-1} (8.12×10^{-2}) −	**5.5961×10^{-1} (5.47×10^{-5}) +**	4.4322×10^{-1} (1.41×10^{-1}) =	4.5384×10^{-1} (1.39×10^{-1}) −	5.4909×10^{-1} (2.74×10^{-3}) −	5.0963×10^{-1} (2.09×10^{-2}) −	5.5565×10^{-1} (1.88×10^{-3})
DTLZ5	1.9918×10^{-1} (1.90×10^{-4}) −	**1.8185×10^{-1} (2.28×10^{-6}) −**	1.9909×10^{-1} (1.32×10^{-4}) −	1.9949×10^{-1} (1.50×10^{-4}) −	1.9905×10^{-1} (1.60×10^{-4}) −	1.9603×10^{-1} (2.45×10^{-4}) −	**2.0003×10^{-1} (5.60×10^{-5})**
DTLZ6	1.9947×10^{-1} (1.61×10^{-4}) −	1.8185×10^{-1} (1.04×10^{-6}) −	1.9951×10^{-1} (5.88×10^{-5}) −	1.9994×10^{-1} (7.23×10^{-5}) −	2.0003×10^{-1} (4.69×10^{-5}) =	1.9930×10^{-1} (1.47×10^{-4}) −	**2.0005×10^{-1} (5.32×10^{-5})**
DTLZ7	2.6686×10^{-1} (5.28×10^{-3}) −	2.5729×10^{-1} (5.57×10^{-4}) −	2.6002×10^{-1} (1.97×10^{-2}) −	2.7680×10^{-1} (6.80×10^{-3}) =	2.7520×10^{-1} (1.05×10^{-3}) −	2.4195×10^{-1} (6.62×10^{-3}) −	**2.7806×10^{-1} (9.45×10^{-4})**
+/−/=	0/7/0	1/3/3	0/3/4	0/6/1	0/4/3	0/7/0	

Similarly to Tables 1–4, Tables 5 and 6 present the statistical results of IGD and HV metrics obtained by QMOSNS, NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS on the UF test functions, respectively. In terms of the IGD metric, QMOSNS demonstrates the best convergence performance on UF2, UF5, UF6, and UF7. PPS achieves the best convergence on UF1 and UF3, CMOES performs best on UF4 and UF10, while MOEADDAE shows superior convergence on UF8 and UF9. Regarding the HV metric, the QMOSNS algorithm outperforms the other algorithms on most functions. PPS only exhibits competitive performance on UF1, UF3, and UF7. Additionally, MOEADDAE also demonstrates better performance than other algorithms on UF8. According to the Wilcoxon rank-sum test, it can be observed from Tables 5 and 6 that QMOSNS significantly outperforms the other methods. Figure 6 illustrates the Pareto optimal fronts obtained by QMOSNS and the true Pareto optimal fronts for each function.

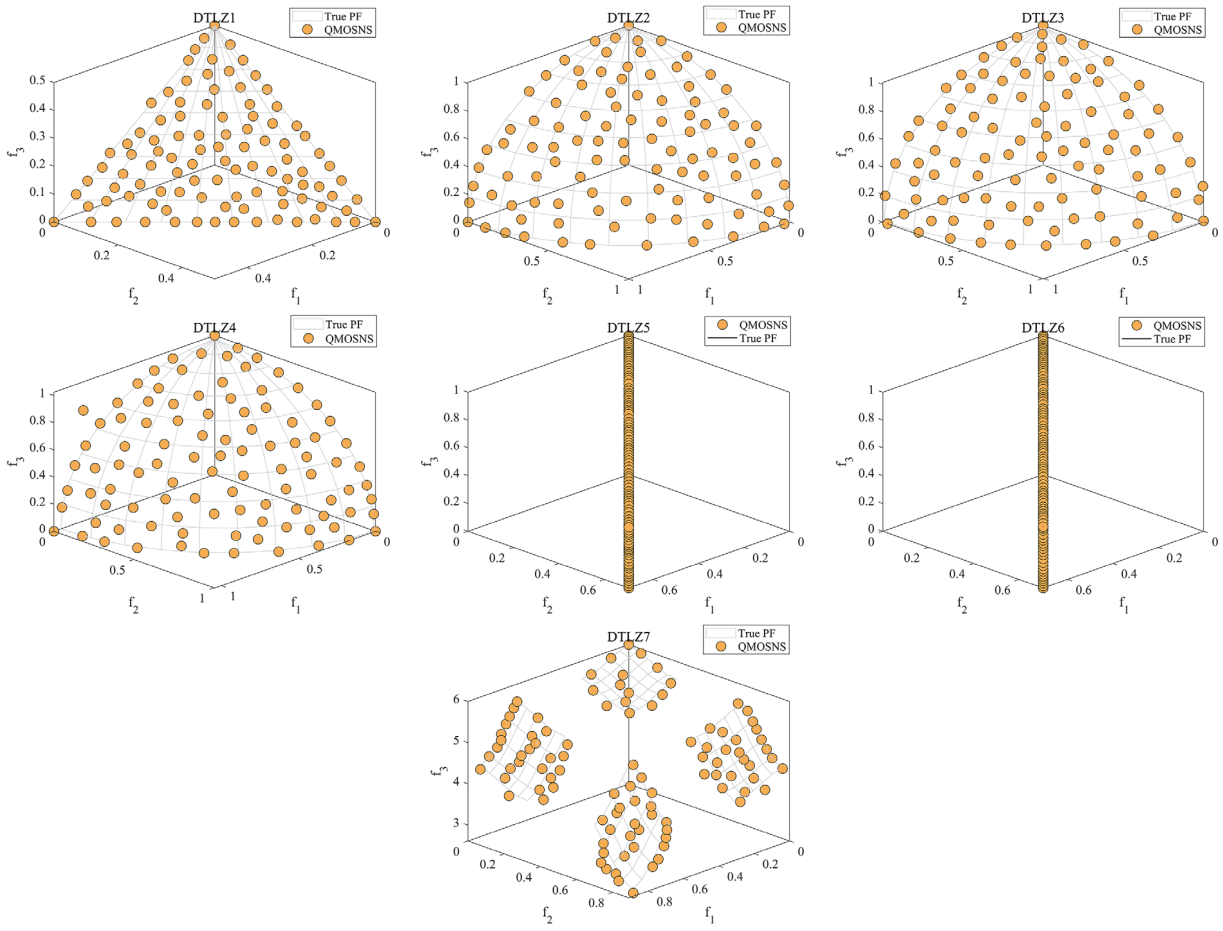


Figure 5. Pareto optimal fronts generated by QMOSNS for test functions DTLZ1-DTLZ7.

Table 5. Statistical Results of MOO Algorithms in Terms of IGD For UF Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
UF1	1.0436×10^{-1} (2.06×10^{-2}) −	1.3958×10^{-1} (5.00×10^{-2}) −	1.1366×10^{-1} (2.55×10^{-2}) −	1.2412×10^{-1} (6.88×10^{-2}) −	9.9917×10^{-2} (2.04×10^{-2}) −	**2.5277×10^{-2} (2.46×10^{-2}) +**	8.1566×10^{-2} (2.49×10^{-3})
UF2	3.7636×10^{-2} (1.35×10^{-2}) −	9.2915×10^{-2} (4.85×10^{-2}) −	4.7011×10^{-2} (1.55×10^{-2}) −	4.9463×10^{-2} (3.75×10^{-2}) −	3.4721×10^{-2} (1.01×10^{-2}) −	3.3001×10^{-2} (2.69×10^{-2}) −	**2.9344×10^{-2} (6.98×10^{-3})**
UF3	2.2090×10^{-1} (6.33×10^{-2}) =	2.5427×10^{-1} (3.45×10^{-2}) =	2.9052×10^{-1} (3.39×10^{-2}) −	2.6687×10^{-1} (4.38×10^{-2}) −	1.8408×10^{-1} (4.54×10^{-2}) +	**1.2160×10^{-1} (5.52×10^{-2}) +**	2.3783×10^{-1} (3.95×10^{-2})
UF4	4.6240×10^{-2} (1.18×10^{-3}) =	5.5563×10^{-2} (2.96×10^{-3}) −	4.6259×10^{-2} (1.60×10^{-3}) =	5.2656×10^{-2} (4.99×10^{-3}) −	**4.4842×10^{-2} (7.00×10^{-4}) =**	7.9341×10^{-2} (7.25×10^{-3}) −	4.8162×10^{-2} (5.50×10^{-3})
UF5	3.0432×10^{-1} (7.93×10^{-2}) −	4.3658×10^{-1} (1.18×10^{-1}) −	3.3533×10^{-1} (1.15×10^{-1}) −	3.9002×10^{-1} (1.38×10^{-1}) −	3.1027×10^{-1} (1.07×10^{-1}) −	4.9008×10^{-1} (1.08×10^{-1}) −	**2.7604×10^{-1} (1.28×10^{-1})**
UF6	1.9411×10^{-1} (9.74×10^{-2}) =	2.7970×10^{-1} (1.58×10^{-1}) −	2.1489×10^{-1} (1.29×10^{-1}) =	2.8230×10^{-1} (1.24×10^{-1}) −	1.5863×10^{-1} (9.66×10^{-2}) −	3.3513×10^{-1} (1.61×10^{-1}) −	**1.5848×10^{-1} (7.48×10^{-2})**
UF7	1.5186×10^{-1} (1.52×10^{-1}) =	2.1207×10^{-1} (1.64×10^{-1}) −	2.0253×10^{-1} (1.49×10^{-1}) −	2.9699×10^{-1} (1.92×10^{-1}) −	9.4093×10^{-2} (1.07×10^{-1}) =	3.3251×10^{-2} (1.00×10^{-1}) +	**1.4691×10^{-1} (1.28×10^{-1})**
UF8	2.7834×10^{-1} (6.53×10^{-2}) −	2.9534×10^{-1} (4.76×10^{-2}) −	2.5379×10^{-1} (2.50×10^{-2}) −	**1.8529×10^{-1} (1.16×10^{-1}) +**	2.6765×10^{-1} (6.46×10^{-2}) −	2.4754×10^{-1} (6.58×10^{-2}) −	2.1906×10^{-1} (5.01×10^{-2})
UF9	3.7086×10^{-1} (1.13×10^{-1}) −	3.3210×10^{-1} (5.02×10^{-2}) −	2.6802×10^{-1} (8.26×10^{-2}) =	**2.5230×10^{-1} (6.55×10^{-2}) =**	**2.9483×10^{-1} (9.63×10^{-2}) =**	3.0348×10^{-1} (8.14×10^{-2}) =	2.8510×10^{-1} (3.74×10^{-2})
UF10	4.4203×10^{-1} (9.14×10^{-2}) =	6.3308×10^{-1} (2.43×10^{-1}) −	4.2346×10^{-1} (1.46×10^{-1}) =	4.9013×10^{-1} (1.02×10^{-1}) −	**3.6983×10^{-1} (5.24×10^{-2}) =**	7.0851×10^{-1} (1.07×10^{-1}) −	3.9883×10^{-1} (1.23×10^{-1})
+/−/=	0/5/5	0/9/1	0/6/4	1/8/1	1/5/4	3/6/1	

Table 6. Statistical Results of MOO Algorithms in Terms of HV For UF Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
UF1	5.9375×10^{-1} (2.46×10^{-2}) −	5.6023×10^{-1} (4.72×10^{-2}) −	5.8039×10^{-1} (2.32×10^{-2}) −	5.7774×10^{-1} (4.54×10^{-2}) −	6.0197×10^{-1} (2.17×10^{-2}) −	**6.9003×10^{-1} (3.21×10^{-2}) +**	6.2120×10^{-1} (3.92×10^{-3})
UF2	6.8125×10^{-1} (8.17×10^{-3}) −	6.5392×10^{-1} (2.23×10^{-2}) −	6.7160×10^{-1} (9.89×10^{-3}) −	6.7670×10^{-1} (1.95×10^{-2}) −	6.8453×10^{-1} (6.11×10^{-3}) −	6.8895×10^{-1} (2.04×10^{-2}) =	**6.9277×10^{-1} (4.14×10^{-3})**
UF3	4.6623×10^{-1} (5.40×10^{-2}) =	4.4097×10^{-1} (3.10×10^{-2}) −	4.0582×10^{-1} (2.90×10^{-2}) −	4.0569×10^{-1} (3.53×10^{-2}) −	5.0818×10^{-1} (5.12×10^{-2}) +	**5.3844×10^{-1} (7.89×10^{-2}) +**	4.7645×10^{-1} (2.50×10^{-2})
UF4	3.8307×10^{-1} (1.43×10^{-3}) −	3.6728×10^{-1} (5.20×10^{-3}) −	3.8356×10^{-1} (1.95×10^{-3}) −	3.7410×10^{-1} (7.25×10^{-3}) −	3.8632×10^{-1} (8.47×10^{-4}) −	3.3559×10^{-1} (8.86×10^{-3}) −	**3.9195×10^{-1} (4.71×10^{-3})**
UF5	2.2367×10^{-1} (7.89×10^{-2}) −	1.6565×10^{-1} (6.24×10^{-2}) −	2.2215×10^{-1} (6.74×10^{-2}) −	1.9810×10^{-1} (7.67×10^{-2}) −	2.2981×10^{-1} (7.94×10^{-2}) −	1.2029×10^{-1} (8.22×10^{-2}) −	**2.7202×10^{-1} (8.26×10^{-2})**
UF6	3.1154×10^{-1} (6.15×10^{-2}) −	2.8049×10^{-1} (6.81×10^{-2}) −	3.1103×10^{-1} (5.79×10^{-2}) −	2.7102×10^{-1} (7.11×10^{-2}) −	3.3925×10^{-1} (3.41×10^{-2}) −	2.1913×10^{-1} (9.83×10^{-2}) −	**3.5475×10^{-1} (3.07×10^{-2})**
UF7	4.4883×10^{-1} (1.06×10^{-1}) =	3.9918×10^{-1} (1.10×10^{-1}) −	4.0966×10^{-1} (1.05×10^{-1}) −	3.4553×10^{-1} (1.32×10^{-1}) −	4.8942×10^{-1} (7.79×10^{-2}) =	**5.4923×10^{-1} (7.42×10^{-2}) +**	4.5492×10^{-1} (1.02×10^{-1})
UF8	2.6536×10^{-1} (4.80×10^{-2}) −	3.1474×10^{-1} (2.36×10^{-2}) −	3.3145×10^{-1} (1.48×10^{-2}) −	**3.9744×10^{-1} (7.11×10^{-2}) +**	2.8342×10^{-1} (4.08×10^{-2}) −	2.9743×10^{-1} (4.79×10^{-2}) −	3.5811×10^{-1} (3.54×10^{-2})
UF9	3.7830×10^{-1} (9.85×10^{-2}) −	4.4321×10^{-1} (4.34×10^{-2}) −	4.9272×10^{-1} (7.11×10^{-2}) −	5.4239×10^{-1} (6.47×10^{-2}) −	4.5524×10^{-1} (9.24×10^{-2}) −	4.8820×10^{-1} (9.64×10^{-2}) −	**5.9433×10^{-1} (3.51×10^{-2})**
UF10	1.0160×10^{-1} (4.50×10^{-2}) −	1.5012×10^{-1} (7.41×10^{-2}) −	1.6860×10^{-1} (7.13×10^{-2}) =	1.2644×10^{-1} (6.87×10^{-2}) −	1.4521×10^{-1} (2.65×10^{-2}) =	3.0092×10^{-2} (2.45×10^{-2}) −	**2.0826×10^{-1} (1.02×10^{-1})**
+/−/=	0/8/2	0/10/0	0/9/1	1/9/0	1/7/2	3/6/1	

The aggregated performance metrics across Tables 1–6 demonstrate that QMOSNS emerges as the superior algorithm among the seven evaluated methods for handling unconstrained multi-objective optimization problems.

5.2.2. Constrained Multi-Objective Problems

In the previous section on unconstrained multi-objective test problems, the proposed QMOSNS demonstrated superior performance. To further evaluate its capability in handling constrained multi-objective optimization problems, we conducted performance assessments on two constrained test suites.

Tables 7 and 8 present the statistical results of IGD and HV metrics obtained by QMOSNS and comparative algorithms on the C_DTLZ test set. The results indicate that QMOSNS achieves the best performance on C1_DTLZ1, C2_DTLZ2, and C3_DTLZ4, with its IGD values demonstrating excellent convergence properties. On the C1_DTLZ3 function, CMOES yields the optimal results, followed by MOEADDAE, while the other algorithms fail to converge close to the true Pareto optimal front. These observations are further supported by the HV metric results in Table 8. Overall, QMOSNS exhibits the strongest performance across this test set. Figure 7 illustrates the Pareto optimal front obtained by QMOSNS and the corresponding true Pareto front for this test set. Figure 7 reveals that QMOSNS achieves excellent convergence to the true Pareto optimal front with proper distribution and spread for the Pareto optimal solutions, except in the case of C1_DTLZ3.

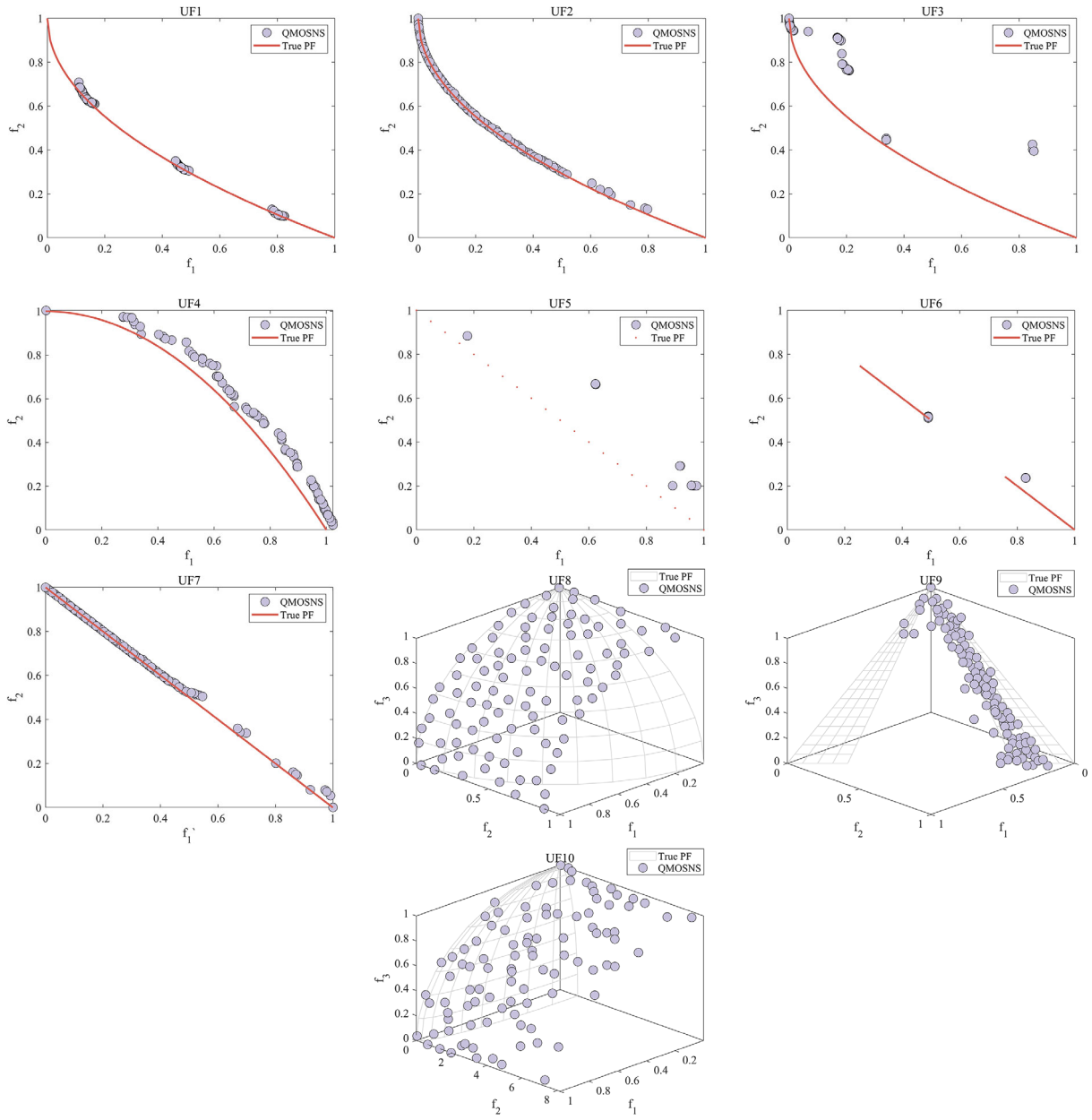


Figure 6. Pareto optimal fronts generated by QMOSNS for test functions UF1-UF10.

Table 7. Statistical Results of MOO Algorithms in Terms of IGD For C_DTLZ Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	**Mean (Std)**	**Mean (Std)**	**Mean (Std)**	**Mean (Std)**	**Mean (Std)**	**Mean (Std)**
C1_DTLZ1	2.6661×10^{-2} (1.11×10^{-3}) −	2.0490×10^{-2} (5.91×10^{-5}) −	2.0443×10^{-2} (8.87×10^{-5}) −	2.1013×10^{-2} (1.25×10^{-4}) −	2.0163×10^{-2} (1.84×10^{-4}) =	2.5783×10^{-2} (6.41×10^{-4}) −	**2.0190×10^{-2} (1.43×10^{-4})**
C1_DTLZ3	4.5775×10^{0} (4.01×10^{0}) =	4.4759×10^{0} (3.90×10^{0}) +	6.4188×10^{0} (3.23×10^{0}) +	6.2912×10^{-2} (4.69×10^{-3}) +	**5.4429×10^{-2} (7.75×10^{-4}) +**	2.4606×10^{0} (3.70×10^{0}) +	8.0120×10^{0} (3.77×10^{-3})
C2_DTLZ2	5.6989×10^{-2} (3.13×10^{-3}) −	4.9314×10^{-2} (3.04×10^{-5}) −	4.3653×10^{-2} (2.09×10^{-4}) −	4.4162×10^{-2} (4.91×10^{-4}) −	4.2947×10^{-2} (7.13×10^{-4}) −	5.5358×10^{-2} (1.71×10^{-3}) −	**4.2112×10^{-2} (5.25×10^{-4})**
C3_DTLZ4	1.2813×10^{-1} (5.22×10^{-3}) −	1.1649×10^{-1} (1.37×10^{-1}) −	2.4717×10^{-1} (3.84×10^{-1}) −	8.9333×10^{-1} (4.07×10^{-1}) −	9.7132×10^{-2} (1.54×10^{-3}) −	1.7135×10^{-1} (9.25×10^{-2}) −	**9.4932×10^{-2} (1.71×10^{-3})**
+/−/=	0/3/1	1/3/0	1/3/0	1/3/0	1/2/1	1/3/0	

Table 8. Statistical Results of MOO Algorithms in Terms of HV For C_DTLZ Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
C1_DTLZ1	8.2413×10^{-1} (4.20×10^{-3}) −	8.4050×10^{-1} (1.22×10^{-3}) =	8.3981×10^{-1} (2.04×10^{-3}) −	8.4032×10^{-1} (1.00×10^{-3}) −	8.3986×10^{-1} (2.55×10^{-3}) =	8.1900×10^{-1} (4.06×10^{-3}) −	**8.4101×10^{-1} (6.92×10^{-4})**
C1_DTLZ3	2.2957×10^{-1} (2.67×10^{-1}) +	2.1690×10^{-1} (2.72×10^{-1}) +	1.0804×10^{-1} (2.21×10^{-1}) +	5.5702×10^{-1} (1.70×10^{-3}) +	**5.5816×10^{-1} (1.59×10^{-3}) +**	3.5964×10^{-1} (2.40×10^{-1}) +	0.0000×10^{0} (0.00×10^{0})
C2_DTLZ2	4.8983×10^{-1} (4.02×10^{-3}) −	5.1524×10^{-1} (9.47×10^{-5}) −	5.1410×10^{-1} (1.74×10^{-3}) −	5.1497×10^{-1} (1.79×10^{-3}) −	5.1247×10^{-1} (2.26×10^{-3}) −	4.9937×10^{-1} (3.75×10^{-3}) −	**5.1746×10^{-1} (1.17×10^{-3})**
C3_DTLZ4	7.6465×10^{-1} (4.48×10^{-3}) −	7.8688×10^{-1} (4.87×10^{-2}) −	7.3721×10^{-1} (1.51×10^{-1}) −	4.0066×10^{-1} (1.83×10^{-1}) −	7.8811×10^{-1} (1.43×10^{-3}) −	7.5983×10^{-1} (2.65×10^{-2}) −	**7.9090×10^{-1} (1.71×10^{-3})**
+/−/=	1/3/0	1/2/1	1/3/0	1/2/1	1/2/1	1/3/0	

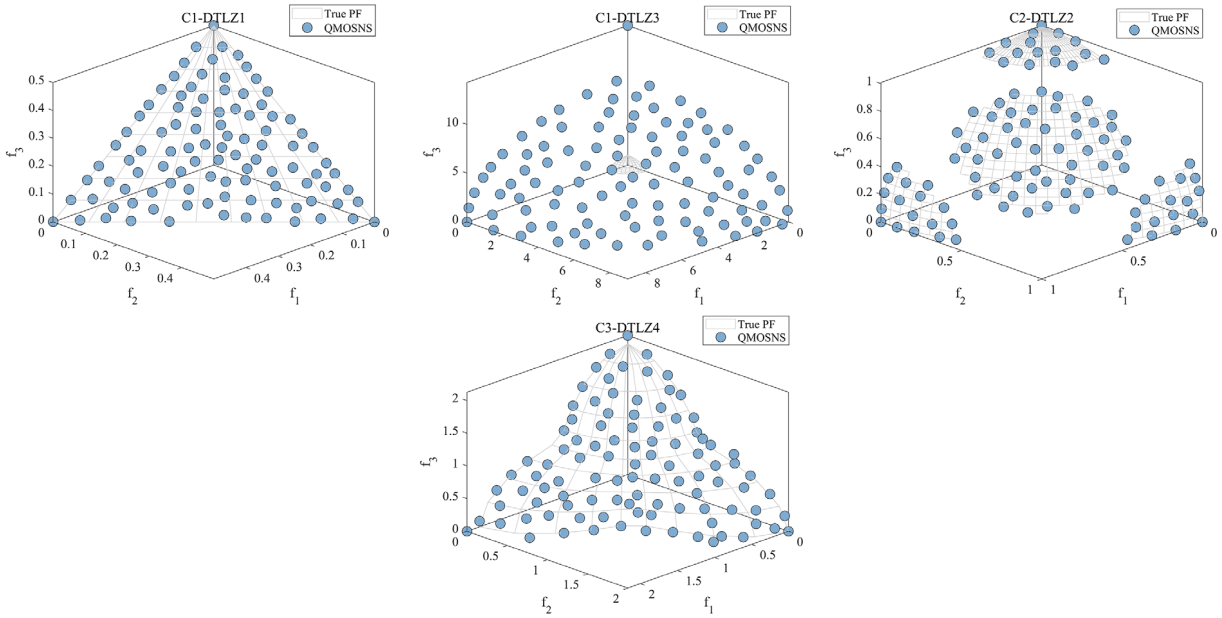


Figure 7. Pareto optimal fronts generated by QMOSNS for test functions C1-DTLZ1, C1-DTLZ3, C2-DTLZ2, and C3-DTLZ4.

The MW test suite presents significant optimization challenges due to its notably small feasible regions and multiple nonlinear constraints. Tables 9 and 10 display the statistical results of IGD and HV metrics obtained by QMOSNS, NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS on the MW test set, respectively. Analysis of IGD metrics reveals that QMOSNS achieves optimal performance on most functions, including MW1, MW2, MW4, MW6, MW8, MW10, and MW13. This superior performance is consistently reflected in the corresponding HV metric results as well. While QMOSNS shows relatively weaker performance compared to some algorithms on the remaining functions, the Wilcoxon rank-sum test results for both IGD and HV metrics demonstrate that the proposed algorithm maintains substantial advantages over NSGA-II, MOEA/D, ARMOEA, MOEADDAE, and PPS, with only marginal superiority when compared to CMOES. Figure 8 provides a graphical representation comparing the true Pareto front of the MW test suite with the Pareto fronts obtained by QMOSNS. These figures show that some of the test problems have special Pareto optimal fronts. Figure 8 reveals that QMOSNS consistently achieves solutions with both strong convergence to the true Pareto fronts and uniform distribution across the entire optimal regions.

Table 9. Statistical Results of MOO Algorithms in Terms of IGD For MW Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
MW1	5.0439×10^{-3} (2.92×10^{-3}) −	8.6758×10^{-3} (3.45×10^{-3}) −	1.1113×10^{-2} (1.55×10^{-2}) −	8.2137×10^{-3} (1.85×10^{-2}) −	1.6647×10^{-3} (4.00×10^{-5}) −	7.6059×10^{-3} (4.53×10^{-3}) −	**1.6484×10^{-3} (5.13×10^{-5})**
MW2	2.5613×10^{-2} (1.03×10^{-2}) −	2.3825×10^{-2} (8.72×10^{-3}) −	2.8420×10^{-2} (1.40×10^{-2}) −	3.9482×10^{-2} (1.23×10^{-2}) −	1.3023×10^{-2} (8.64×10^{-3}) −	1.5179×10^{-1} (1.27×10^{-1}) −	**3.7268×10^{-3} (1.89×10^{-5})**
MW3	6.0586×10^{-3} (2.61×10^{-4}) =	1.7567×10^{-2} (3.35×10^{-2}) −	1.3161×10^{-2} (2.39×10^{-2}) =	5.7479×10^{-3} (5.07×10^{-4}) =	**5.1306×10^{-3} (3.03×10^{-4}) +**	6.0695×10^{-3} (5.26×10^{-4}) =	5.6287×10^{-3} (5.54×10^{-4})
MW4	5.6744×10^{-2} (2.20×10^{-3}) −	4.4540×10^{-2} (2.55×10^{-3}) −	4.1562×10^{-2} (1.10×10^{-4}) −	4.3055×10^{-2} (1.22×10^{-3}) −	4.2187×10^{-2} (6.53×10^{-4}) −	8.9115×10^{-2} (7.52×10^{-2}) −	**4.1135×10^{-2} (3.62×10^{-4})**
MW5	4.7616×10^{-1} (3.45×10^{-1}) −	3.6708×10^{-3} (2.25×10^{-3}) =	3.3312×10^{-1} (3.57×10^{-1}) −	1.4872×10^{-2} (1.30×10^{-2}) −	**8.3645×10^{-4} (1.36×10^{-4}) +**	2.4809×10^{-1} (2.98×10^{-1}) −	3.6696×10^{-3} (1.14×10^{-3})
MW6	1.0426×10^{-1} (1.69×10^{-1}) −	1.7326×10^{-2} (9.05×10^{-3}) −	6.6015×10^{-2} (1.36×10^{-1}) −	7.0572×10^{-2} (4.29×10^{-2}) −	1.2291×10^{-2} (8.44×10^{-3}) −	5.2652×10^{-1} (3.31×10^{-1}) −	**3.2689×10^{-3} (1.57×10^{-3})**
MW7	1.0394×10^{-1} (1.84×10^{-1}) =	**4.8889×10^{-3} (2.11×10^{-4}) +**	1.4826×10^{-2} (1.98×10^{-2}) =	5.1604×10^{-3} (7.30×10^{-4}) +	4.9246×10^{-3} (3.83×10^{-4}) +	5.5890×10^{-3} (5.08×10^{-4}) +	6.8865×10^{-3} (1.09×10^{-3})
MW8	6.5049×10^{-2} (8.69×10^{-3}) −	5.0555×10^{-2} (1.18×10^{-3}) −	4.9557×10^{-2} (5.82×10^{-3}) −	7.7832×10^{-2} (4.26×10^{-2}) −	4.3733×10^{-2} (1.18×10^{-3}) −	1.5878×10^{-1} (1.02×10^{-1}) −	**4.2075×10^{-2} (6.58×10^{-4})**
MW9	1.5317×10^{-1} (2.86×10^{-1}) =	2.8690×10^{-2} (2.89×10^{-2}) =	1.5522×10^{-2} (8.52×10^{-3}) =	7.7868×10^{-2} (2.22×10^{-1}) −	**4.7448×10^{-3} (2.06×10^{-4}) +**	2.1343×10^{-1} (3.05×10^{-1}) −	1.4595×10^{-2} (4.58×10^{-3})
MW10	1.3243×10^{-1} (6.33×10^{-2}) −	5.6674×10^{-2} (2.03×10^{-2}) −	1.6107×10^{-1} (1.72×10^{-1}) −	2.4756×10^{-1} (2.18×10^{-1}) −	1.6548×10^{-2} (2.04×10^{-2}) −	3.9786×10^{-1} (2.64×10^{-1}) −	**3.5036×10^{-3} (6.87×10^{-5})**
MW11	5.5903×10^{-1} (2.90×10^{-1}) −	3.5536×10^{-1} (3.61×10^{-1}) −	4.2591×10^{-1} (3.50×10^{-1}) −	7.7566×10^{-3} (5.96×10^{-4}) =	**6.1930×10^{-3} (1.95×10^{-4}) +**	7.1952×10^{-3} (2.41×10^{-4}) =	7.9184×10^{-3} (1.85×10^{-3})
MW12	1.5406×10^{-1} (3.01×10^{-1}) −	5.0312×10^{-3} (1.55×10^{-4}) =	7.6721×10^{-2} (1.74×10^{-1}) −	8.2274×10^{-2} (2.43×10^{-1}) −	7.2094×10^{-2} (2.12×10^{-1}) =	1.5019×10^{-1} (2.51×10^{-1}) −	**5.0175×10^{-3} (1.77×10^{-4})**
MW13	1.1926×10^{-1} (8.21×10^{-2}) −	7.2207×10^{-2} (3.29×10^{-2}) −	4.1184×10^{-1} (4.19×10^{-1}) −	1.0196×10^{-1} (3.71×10^{-2}) −	3.6053×10^{-2} (2.90×10^{-2}) −	4.5019×10^{-1} (3.62×10^{-1}) −	**1.1450×10^{-2} (1.02×10^{-3})**
MW14	1.2762×10^{-1} (8.93×10^{-3}) =	2.1166×10^{-1} (2.53×10^{-3}) =	1.0980×10^{-1} (2.44×10^{-3}) =	1.0405×10^{-1} (3.87×10^{-3}) =	**1.0135×10^{-1} (1.49×10^{-3}) =**	1.7169×10^{-1} (4.30×10^{-2}) =	1.6694×10^{-1} (1.08×10^{-1})
+/−/=	0/10/4	1/9/4	0/10/4	1/10/3	5/7/2	1/10/3	

Table 10. Statistical Results of MOO Algorithms in Terms of HV For MW Test Suites.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
MW1	4.8335×10^{-1} (6.12×10^{-3}) −	4.7744×10^{-1} (7.10×10^{-3}) −	4.7571×10^{-1} (1.67×10^{-2}) −	4.8100×10^{-1} (2.43×10^{-2}) =	4.8970×10^{-1} (1.54×10^{-4}) =	4.7675×10^{-1} (9.19×10^{-3}) −	**4.8975×10^{-1} (2.63×10^{-4})**
MW2	5.4575×10^{-1} (1.55×10^{-2}) −	5.4763×10^{-1} (1.29×10^{-2}) −	5.4218×10^{-1} (2.19×10^{-2}) −	5.2548×10^{-1} (1.71×10^{-2}) −	5.6644×10^{-1} (1.47×10^{-2}) −	3.9623×10^{-1} (1.33×10^{-1}) −	**5.8242×10^{-1} (2.24×10^{-5})**
MW3	5.4287×10^{-1} (4.00×10^{-4}) −	5.3498×10^{-1} (2.29×10^{-2}) −	5.3708×10^{-1} (2.16×10^{-2}) =	5.4328×10^{-1} (4.94×10^{-4}) =	**5.4377×10^{-1} (7.16×10^{-4}) =**	5.4277×10^{-1} (8.79×10^{-4}) =	5.4361×10^{-1} (1.02×10^{-3})
MW4	8.2383×10^{-1} (2.81×10^{-3}) −	8.3859×10^{-1} (2.21×10^{-3}) −	8.4127×10^{-1} (1.37×10^{-4}) =	8.3964×10^{-1} (1.49×10^{-3}) −	8.3994×10^{-1} (6.07×10^{-4}) −	7.7147×10^{-1} (8.04×10^{-2}) −	**8.4142×10^{-1} (2.69×10^{-4})**
MW5	1.6046×10^{-1} (9.06×10^{-2}) −	3.2280×10^{-1} (1.08×10^{-3}) =	2.0545×10^{-1} (1.07×10^{-1}) −	3.1550×10^{-1} (9.29×10^{-3}) =	**3.2410×10^{-1} (1.22×10^{-4}) +**	2.2144×10^{-1} (1.04×10^{-1}) −	3.2206×10^{-1} (8.09×10^{-4})
MW6	2.7885×10^{-1} (4.08×10^{-2}) −	3.0718×10^{-1} (1.16×10^{-2}) −	2.8563×10^{-1} (4.31×10^{-2}) −	2.5336×10^{-1} (3.65×10^{-2}) −	3.1386×10^{-1} (1.24×10^{-2}) −	1.0847×10^{-1} (9.03×10^{-2}) −	**3.2764×10^{-1} (2.60×10^{-3})**
MW7	3.7545×10^{-1} (6.97×10^{-2}) =	4.1097×10^{-1} (2.78×10^{-4}) +	4.0907×10^{-1} (3.56×10^{-3}) =	4.1207×10^{-1} (3.71×10^{-4}) +	4.1144×10^{-1} (7.45×10^{-4}) +	**4.1146×10^{-1} (3.37×10^{-4}) +**	4.0883×10^{-1} (1.82×10^{-3})
MW8	4.7937×10^{-1} (2.39×10^{-2}) −	5.3247×10^{-1} (6.10×10^{-3}) −	5.2155×10^{-1} (1.77×10^{-2}) −	4.6104×10^{-1} (7.82×10^{-2}) −	5.4346×10^{-1} (7.60×10^{-3}) −	3.3354×10^{-1} (1.32×10^{-1}) −	**5.5278×10^{-1} (6.68×10^{-4})**
MW9	3.0354×10^{-1} (1.60×10^{-1}) =	3.6331×10^{-1} (3.07×10^{-2}) =	3.7990×10^{-1} (8.09×10^{-3}) =	3.5461×10^{-1} (1.25×10^{-1}) −	**3.9749×10^{-1} (1.63×10^{-3}) +**	2.5766×10^{-1} (1.74×10^{-1}) −	3.8034×10^{-1} (4.66×10^{-3})

Table 10. *Cont.*

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
MW10	3.5692×10^{-1} (4.08×10^{-2}) −	4.0208×10^{-1} (1.45×10^{-2}) −	3.4582×10^{-1} (8.59×10^{-2}) −	3.0205×10^{-1} (1.09×10^{-1}) −	4.3812×10^{-1} (1.82×10^{-2}) −	2.2690×10^{-1} (1.23×10^{-1}) −	**4.5434×10^{-1} (2.14×10^{-4})**
MW11	3.0737×10^{-1} (7.33×10^{-2}) −	3.5860×10^{-1} (9.10×10^{-2}) −	3.4045×10^{-1} (8.79×10^{-2}) −	4.4413×10^{-1} (1.08×10^{-3}) −	4.4728×10^{-1} (1.64×10^{-4}) +	**4.4730×10^{-1} (1.21×10^{-4}) +**	4.4638×10^{-1} (7.78×10^{-4})
MW12	4.7973×10^{-1} (2.48×10^{-1}) −	**6.0436×10^{-1} (3.16×10^{-4}) +**	5.3769×10^{-1} (1.64×10^{-1}) =	5.4315×10^{-1} (1.91×10^{-1}) =	5.4577×10^{-1} (1.86×10^{-1}) −	4.6451×10^{-1} (2.24×10^{-1}) −	6.0402×10^{-1} (3.06×10^{-4})
MW13	4.1837×10^{-1} (4.53×10^{-2}) −	4.4437×10^{-1} (1.93×10^{-2}) −	3.6279×10^{-1} (7.88×10^{-2}) −	4.2589×10^{-1} (2.17×10^{-2}) −	4.6150×10^{-1} (1.28×10^{-2}) −	2.8453×10^{-1} (1.13×10^{-1}) −	**4.7643×10^{-1} (5.69×10^{-4})**
MW14	4.5107×10^{-1} (1.93×10^{-3}) =	4.3980×10^{-1} (3.66×10^{-3}) =	4.7076×10^{-1} (2.92×10^{-3}) +	**4.7392×10^{-1} (1.70×10^{-3}) +**	4.6970×10^{-1} (1.70×10^{-3}) =	4.3802×10^{-1} (9.40×10^{-3}) =	4.4873×10^{-1} (3.67×10^{-2})
+/−/=	0/11/3	2/9/3	1/8/5	2/8/4	4/7/3	2/10/2	

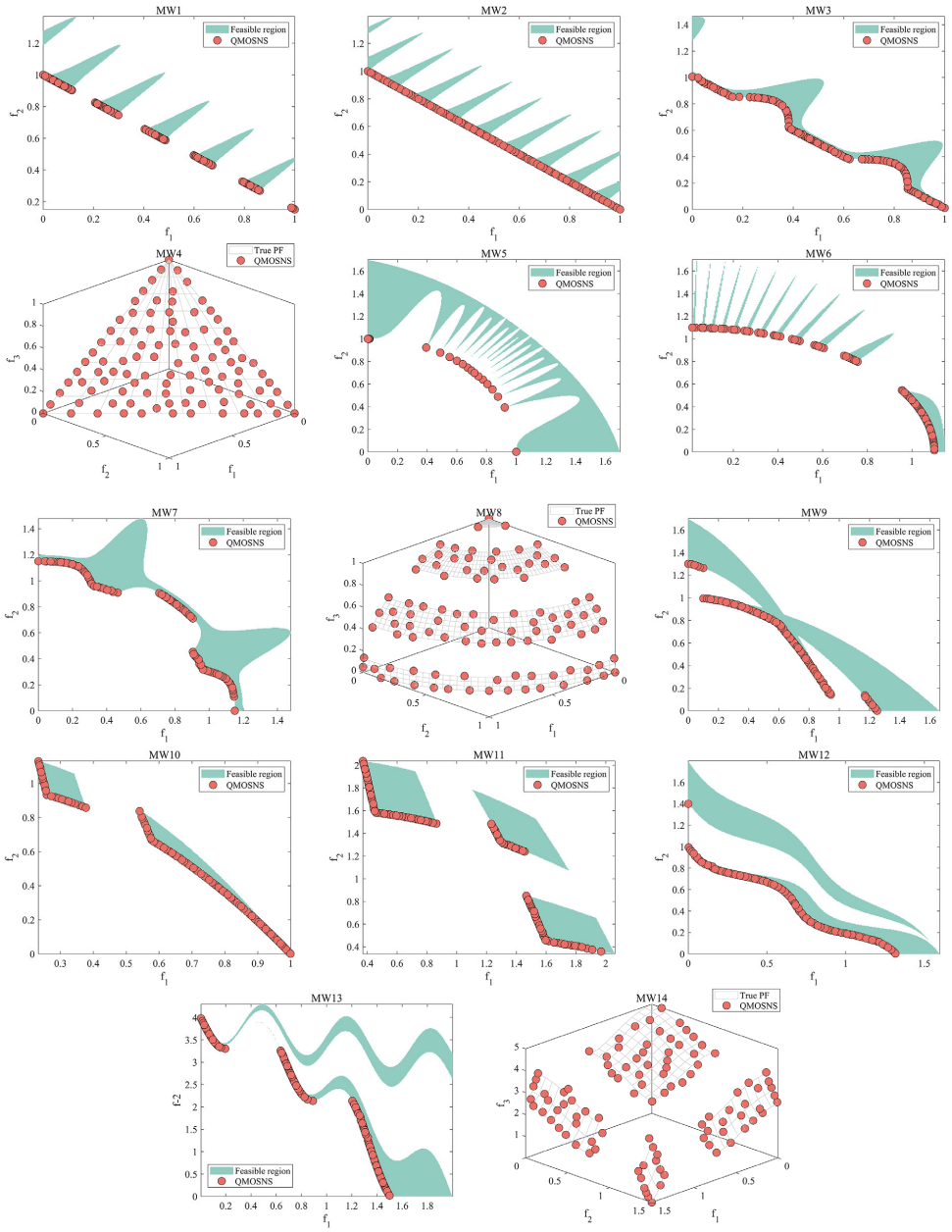


Figure 8. Pareto optimal fronts generated by QMOSNS for test functions MW1-MW14.

The test results on constrained multi-objective optimization problems demonstrate that QMOSNS effectively handles constraints and obtains Pareto optimal fronts with both high convergence and good distribution across different Pareto optimal regions.

5.3. Multi-Objective Engineering Design Problems

While multi-objective benchmark functions are effective for algorithm testing, real-world engineering optimization problems pose greater challenges to algorithm performance. To evaluate the applicability of the proposed algorithm, this study further tests the algorithm on 10 multi-objective engineering design problems. The basic information of these 10 test problems is presented in Table 11, where m denotes the number of objectives, d represents the problem dimension, ng indicates the number of inequality constraints, and nh stands for the number of equality constraints. Since real multi-objective engineering design problems lack known true Pareto fronts, the HV metric is exclusively used to evaluate algorithm performance on these problems. Table 12 provides the statistical results of HV metrics obtained by each algorithm.

Table 11. Basic Information on Multi-objective Engineering Design Problems.

Problem	Name	*m*	*d*	*ng*	*nh*
RWMOP1	Pressure Vessel Design	2	2	2	0
RWMOP2	Vibrating Platform Design	2	5	5	0
RWMOP3	Welded Beam Design	2	4	4	0
RWMOP4	Disc Brake Design	2	4	4	0
RWMOP5	Car Side Impact Design	3	7	9	0
RWMOP6	Four Bar Plane Truss	2	4	1	0
RWMOP7	Multiple Disk Clutch Brake Design	2	5	8	0
RWMOP8	Spring Design	2	3	8	0
RWMOP9	Multi-product Batch Plant	3	10	10	0
RWMOP10	Crash Energy Management for High-speed Train	2	6	4	0

Table 12 demonstrates that QMOSNS achieves the best HV metric values on 6 engineering design problems: Vibrating Platform Design, Disc Brake Design, Multiple Disk Clutch Brake Design, Spring Design, Multi-product Batch Plant, and Crash Energy Management for High-speed Train. It ranked second only to ARMOEA on problems Pressure Vessel Design, Car Side Impact Design, and Four Bar Plane Truss, while ranking third on problem Welded Beam Design, performing inferior to NSGA-II and CMOES. Furthermore, the results of the Wilcoxon rank-sum test provide additional evidence that, across these 10 real-world engineering design problems, QMOSNS significantly outperforms NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS algorithms while maintaining favorable robustness. These experimental results confirm that QMOSNS exhibits stronger competitiveness in solving practical engineering design problems compared to other algorithms. Figure 9 provides a graphical representation of the Pareto fronts obtained by QMOSNS.

Table 12. Statistical Results of MOO Algorithms in Terms of HV For Engineering Design Problems.

Function	NSGA-II	MOEA/D	ARMOEA	MOEADDAE	CMOES	PPS	QMOSNS
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
RWMOP1	6.0530×10^{-1} (5.84×10^{-4}) −	1.0776×10^{-1} (5.79×10^{-4}) −	**6.0749×10^{-1} (6.23×10^{-4}) +**	5.5592×10^{-1} (2.36×10^{-2}) −	6.0590×10^{-1} (5.74×10^{-4}) −	4.7622×10^{-1} (5.18×10^{-2}) −	6.0670×10^{-1} (5.63×10^{-4})
RWMOP2	2.4666×10^{-1} (1.27×10^{-1}) −	2.9075×10^{-1} (1.08×10^{-1}) −	2.1771×10^{-1} (1.31×10^{-1}) −	5.6270×10^{-3} (2.57×10^{-2}) −	3.1711×10^{-1} (8.92×10^{-2}) −	3.9133×10^{-1} (8.94×10^{-4}) −	**3.9286×10^{-1} (6.52×10^{-5})**
RWMOP3	**8.5762×10^{-1} (6.20×10^{-3}) +**	1.6127×10^{-2} (3.04×10^{-2}) −	8.5250×10^{-1} (9.13×10^{-3}) =	8.5470×10^{-1} (2.28×10^{-3}) −	8.5671×10^{-1} (4.08×10^{-3}) =	8.4743×10^{-1} (3.32×10^{-3}) −	8.5620×10^{-1} (2.84×10^{-3})
RWMOP4	4.3373×10^{-1} (9.72×10^{-4}) −	4.1892×10^{-1} (8.92×10^{-3}) −	4.3348×10^{-1} (1.53×10^{-3}) −	4.2924×10^{-1} (3.33×10^{-3}) −	4.3388×10^{-1} (8.57×10^{-4}) −	4.3296×10^{-1} (4.76×10^{-4}) −	**4.3477×10^{-1} (1.82×10^{-4})**
RWMOP5	2.5911×10^{-2} (5.60×10^{-5}) −	9.7880×10^{-3} (6.84×10^{-4}) −	**2.6094×10^{-2} (5.91×10^{-5}) +**	2.5747×10^{-2} (1.14×10^{-4}) −	2.5990×10^{-2} (5.96×10^{-5}) −	2.4220×10^{-2} (9.09×10^{-4}) −	2.6040×10^{-2} (5.00×10^{-5})
RWMOP6	4.0908×10^{-1} (1.47×10^{-4}) −	5.3063×10^{-2} (3.74×10^{-5}) −	**4.1008×10^{-1} (7.14×10^{-5}) +**	3.8484×10^{-1} (4.90×10^{-3}) −	4.0936×10^{-1} (1.14×10^{-4}) −	3.8505×10^{-1} (6.51×10^{-3}) −	4.0945×10^{-1} (1.25×10^{-4})
RWMOP7	6.1757×10^{-1} (1.02×10^{-3}) =	1.1983×10^{-1} (2.61×10^{-2}) −	6.1727×10^{-1} (1.41×10^{-3}) =	6.0043×10^{-1} (1.77×10^{-2}) −	6.1585×10^{-1} (1.31×10^{-3}) −	5.7821×10^{-1} (1.68×10^{-2}) −	**6.1801×10^{-1} (2.73×10^{-4})**
RWMOP8	5.4172×10^{-1} (9.93×10^{-4}) −	6.6012×10^{-2} (7.77×10^{-6}) −	5.4127×10^{-1} (1.85×10^{-3}) −	4.1054×10^{-1} (6.20×10^{-2}) −	5.3602×10^{-1} (4.67×10^{-3}) −	5.3369×10^{-1} (2.23×10^{-3}) −	**5.4288×10^{-1} (8.25×10^{-4})**
RWMOP9	3.3496×10^{-1} (1.03×10^{-2}) =	1.9624×10^{-1} (4.20×10^{-2}) −	3.1811×10^{-1} (1.49×10^{-2}) −	2.2856×10^{-1} (5.31×10^{-2}) −	3.3105×10^{-1} (1.17×10^{-2}) −	3.1082×10^{-1} (2.08×10^{-2}) −	**3.3861×10^{-1} (1.75×10^{-2})**
RWMOP10	3.1711×10^{-2} (1.49×10^{-4}) −	2.9320×10^{-2} (2.81×10^{-6}) −	3.1658×10^{-2} (2.34×10^{-4}) −	3.1485×10^{-2} (1.12×10^{-4}) −	3.1741×10^{-2} (4.08×10^{-5}) −	3.1627×10^{-2} (2.60×10^{-5}) −	**3.1760×10^{-2} (8.15×10^{-7})**
+/−/=	1/7/2	0/10/0	3/5/2	0/10/0	0/9/1	0/10/0	

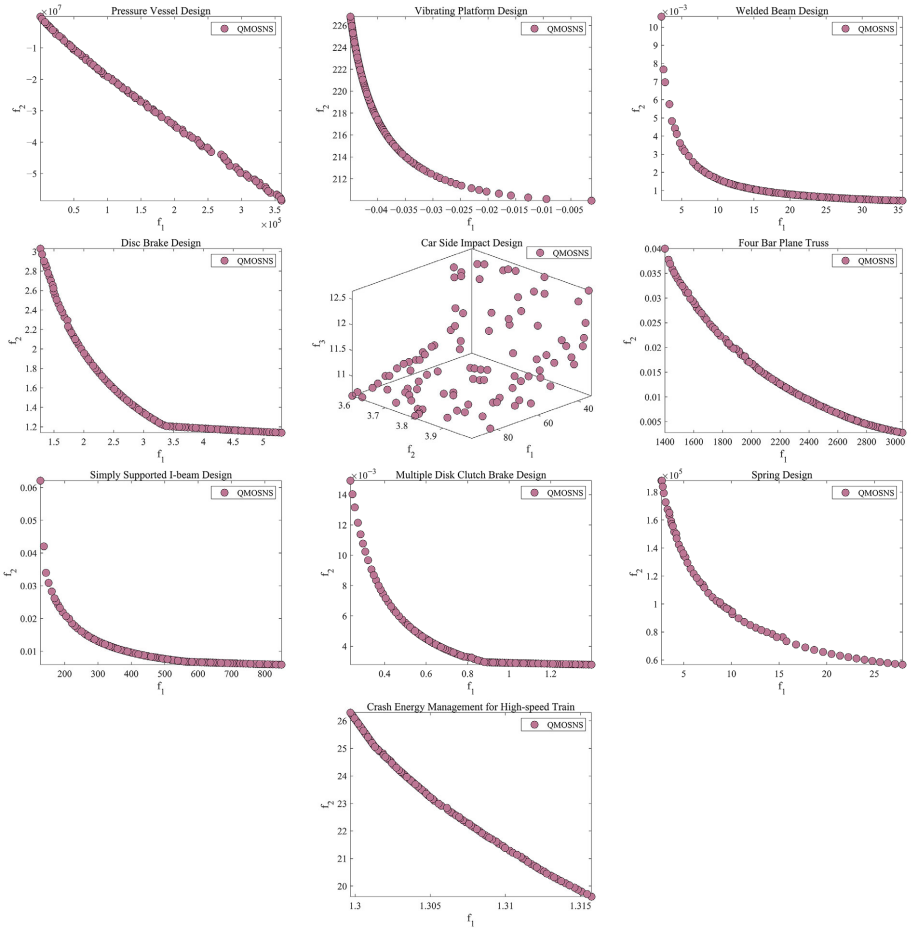


Figure 9. Pareto optimal fronts generated by QMOSNS for engineering design problems.

5.4. Ablation Study on Initialization Strategy

To evaluate the effectiveness of the population initialization based on Halton sequence, an ablation study was conducted on ten test functions selected from the aforementioned 50. In the ablation study, the algorithm without Halton sequence-based population initialization is referred to as QMOSNS-1, while the version employing this initialization method is designated as QMOSNS-2. The performance is measured using the IGD and HV metrics, with all other settings consistent with the previous experiments. According to the results shown in Table 13, QMOSNS-1 and QMOSNS-2 yield generally similar results, though QMOSNS-2 performs slightly better. This indicates that the Halton sequence-based initialization method contributes to some extent to improving the optimization outcomes, thereby demonstrating the effectiveness of the approach.

Table 13. Statistical Results in the Ablation Study.

Problem	QMOSNS-1	QMOSNS-2	QMOSNS-1	QMOSNS-2
	Mean (Std)	**Mean (Std)**	**Mean (Std)**	**Mean (Std)**
	IGD		HV	
ZDT1	3.9705×10^{-3} (6.68×10^{-5}) –	**3.9049×10^{-3} (3.89×10^{-5})**	7.1997×10^{-1} (2.24×10^{-4}) –	**7.2028×10^{-1} (1.24×10^{-4})**
ZDT6	6.7421×10^{-3} (1.74×10^{-2}) –	**3.1914×10^{-3} (1.05×10^{-4})**	3.8432×10^{-1} (2.18×10^{-2}) –	**3.8881×10^{-1} (1.00×10^{-4})**
DTLZ1	2.0464×10^{-2} (1.92×10^{-4}) =	**2.0382×10^{-2} (1.70×10^{-4})**	8.4135×10^{-1} (7.37×10^{-4}) =	**8.4163×10^{-1} (5.88×10^{-4})**
DTLZ2	**5.3883×10^{-2} (6.02×10^{-4}) =**	5.3918×10^{-2} (4.79×10^{-4})	5.5960×10^{-1} (1.25×10^{-3}) =	**5.5963×10^{-1} (1.73×10^{-3})**
UF6	2.1746×10^{-1} (1.37×10^{-1}) =	**1.5848×10^{-1} (7.48×10^{-2})**	3.2183×10^{-1} (5.77×10^{-2}) –	**3.5475×10^{-1} (3.07×10^{-2})**
UF8	2.4906×10^{-1} (4.16×10^{-3}) =	**2.1906×10^{-1} (5.01×10^{-2})**	3.3624×10^{-1} (3.86×10^{-3}) –	**3.5811×10^{-1} (3.54×10^{-2})**
C1_DTLZ1	2.0196×10^{-2} (2.24×10^{-4}) =	**2.0190×10^{-2} (1.43×10^{-4})**	8.4041×10^{-1} (1.10×10^{-3}) –	**8.4101×10^{-1} (6.92×10^{-4})**
C2_DTLZ2	**4.2039×10^{-2} (3.31×10^{-4}) =**	4.2112×10^{-2} (5.25×10^{-4})	5.1651×10^{-1} (1.75×10^{-3})	**5.1746×10^{-1} (1.17×10^{-3})**
MW3	5.8982×10^{-3} (4.08×10^{-4}) =	**5.6287×10^{-3} (5.54×10^{-4})**	5.4295×10^{-1} (8.32×10^{-4}) =	**5.4361×10^{-1} (1.02×10^{-3})**
MW10	5.5396×10^{-3} (3.06×10^{-3}) –	**3.5036×10^{-3} (6.87×10^{-5})**	4.5037×10^{-1} (5.86×10^{-3}) =	**4.5434×10^{-1} (2.14×10^{-4})**
+/−/=				

6. Conclusions

This study proposes a novel multi-objective optimization algorithm called QMOSNS, which is developed based on the Social Network Search algorithm framework to solve complex multi-objective optimization problems. QMOSNS employs the Halton sequence to generate the initial population, thereby enhancing population diversity. By replacing the random selection mechanism in the traditional SNS emotion model with Q-learning, the algorithm can effectively identify optimal mutation strategies during iterations. The approach incorporates an external archive population combined with non-dominated sorting and crowding distance to handle multi-objective optimization problems. The performance of QMOSNS was evaluated against NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS using three test suites: 22 unconstrained multi-objective benchmark problems,

18 constrained multi-objective benchmark problems, and 10 multi-objective engineering design problems, with both IGD and HV metrics employed for performance assessment. The experimental results across all test categories demonstrate that QMOSNS outperforms NSGA-II, MOEA/D, ARMOEA, MOEADDAE, CMOES, and PPS in solving multi-objective optimization problems. In the real world, most multi-objective engineering optimization problems often involve complex constraints and intricate Pareto front structures, posing significant challenges to the MOO algorithms. The QMOSNS method proposed in this study adaptively selects evolutionary operators through a Q-learning mechanism, effectively balancing global exploration and local exploitation processes, thereby enhancing its problem-solving capability in complex optimization scenarios. It has demonstrated strong performance across multiple test problems, proving its considerable applicability and effectiveness in addressing real-world multi-objective optimization challenges. For future research, QMOSNS will be applied to address dynamic optimization, uncertainty optimization, and practical engineering problems, particularly optimization challenges in mechanical design. Additionally, comparative studies will be conducted between QMOSNS and other state-of-the-art multi-objective optimization algorithms to further investigate its performance in solving real-world engineering problems.

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

Funding: This research was funded by the National High Technology Research and Development Program of China (Grant No. 2014AA042001), the National Natural Science Foundation of China (Grant No. 52365012 and 52165008), the Key Project of the Natural Science Foundation of Jiangxi (Grant No. 20224ACB204015), and the Jiangxi Provincial Natural Science Foundation (Grant No. 20252BAC200332).

Data Availability Statement: The data presented in this study are openly available from the PlatEMO 4.11 platform, which provides publicly available benchmark test sets (ZDT, DTLZ, UF, C_DTLZ, MW, and RWMOP). The data can be accessed at: https://github.com/BIMK/PlatEMO(accessed on 14 March 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. He, M.; Wang, Z.; Chen, H.; Cao, Y.; Ma, L. Multi-Objective Evolutionary Algorithm Based on Decomposition with Orthogonal Experimental Design. *Expert Syst.* **2025**, *42*, e13802. [CrossRef]
2. Yin, L.; Sun, Z. Distributed Multi-Objective Grey Wolf Optimizer for Distributed Multi-Objective Economic Dispatch of Multi-Area Interconnected Power Systems. *Appl. Soft Comput.* **2022**, *117*, 108345. [CrossRef]
3. Tang, J.; Dou, T.; Wu, F.; Hu, L.; Yu, T. Scheduling Optimization of a Vehicle Power Battery Workshop Based on an Improved Multi-Objective Particle Swarm Optimization Method. *Mathematics* **2025**, *13*, 2790. [CrossRef]
4. Méndez, M.; Rossit, D.A.; González, B.; Frutos, M. Proposal and Comparative Study of Evolutionary Algorithms for Optimum Design of a Gear System. *IEEE Access* **2020**, *8*, 3482–3497. [CrossRef]
5. Zhou, X.; Cai, X.; Zhang, H.; Zhang, Z.; Jin, T.; Chen, H.; Deng, W. Multi-Strategy Competitive-Cooperative Co-Evolutionary Algorithm and Its Application. *Inf. Sci.* **2023**, *635*, 328–344. [CrossRef]
6. Marler, R.T.; Arora, J.S. The Weighted Sum Method for Multi-Objective Optimization: New Insights. *Struct. Multidiscip. Optim.* **2010**, *41*, 853–862. [CrossRef]
7. Konak, A.; Coit, D.W.; Smith, A.E. Multi-Objective Optimization Using Genetic Algorithms: A Tutorial. *Reliab. Eng. Syst. Saf.* **2006**, *91*, 992–1007. [CrossRef]

8. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A Bio-Inspired Optimizer for Engineering Design Problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
9. Mirjalili, S.; Jangir, P.; Saremi, S. Multi-Objective Ant Lion Optimizer: A Multi-Objective Optimization Algorithm for Solving Engineering Problems. *Appl. Intell.* **2017**, *46*, 79–95. [CrossRef]
10. Premkumar, M.; Jangir, P.; Sowmya, R.; Alhelou, H.H.; Heidari, A.A.; Chen, H. MOSMA: Multi-Objective Slime Mould Algorithm Based on Elitist Non-Dominated Sorting. *IEEE Access* **2021**, *9*, 3229–3248. [CrossRef]
11. Kalita, K.; Jangir, P.; Pandya, S.B.; Cep, R.; Abualigah, L.; Migdady, H.; Daoud, M.S. Many-Objective Artificial Hummingbird Algorithm: An Effective Many-Objective Algorithm for Engineering Design Problems. *J. Comput. Des. Eng.* **2024**, *11*, 16–39. [CrossRef]
12. Schaffer, J.D. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In Proceedings of the First International Conference on Genetic Algorithms and Their Applications, Pittsburgh, PA, USA, 24–26 July 1985; Lawrence Erlbaum: Hillsdale, NJ, USA, 1985; pp. 93–100.
13. Srinivas, N.; Deb, K. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evol. Comput.* **1994**, *2*, 221–248. [CrossRef]
14. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
15. Zhang, Q.; Li, H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
16. Coello, C.A.C.; Pulido, G.T.; Lechuga, M.S. Handling Multiple Objectives with Particle Swarm Optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 256–279. [CrossRef]
17. Fu, S.; Li, K.; Huang, H.; Ma, C.; Fan, Q.; Zhu, Y. Red-Billed Blue Magpie Optimizer: A Novel Metaheuristic Algorithm for 2D/3D UAV Path Planning and Engineering Design Problems. *Artif. Intell. Rev.* **2024**, *57*, 134. [CrossRef]
18. Mirjalili, S.; Saremi, S.; Mirjalili, S.M.; Coelho, L.d.S. Multi-Objective Grey Wolf Optimizer: A Novel Algorithm for Multi-Criterion Optimization. *Expert Syst. Appl.* **2016**, *47*, 106–119. [CrossRef]
19. Sadollah, A.; Eskandar, H.; Kim, J. Water Cycle Algorithm for Solving Constrained Multi-Objective Optimization Problems. *Appl. Soft Comput.* **2015**, *27*, 279–298. [CrossRef]
20. Ganesh, N.; Shankar, R.; Kalita, K.; Jangir, P.; Oliva, D.; Perez-Cisneros, M. A Novel Decomposition-Based Multi-Objective Symbiotic Organism Search Optimization Algorithm. *Mathematics* **2023**, *11*, 1898. [CrossRef]
21. Khodadadi, N.; Soleimanian Gharehchopogh, F.; Mirjalili, S. MOAVOA: A New Multi-Objective Artificial Vultures Optimization Algorithm. *Neural Comput. Appl.* **2022**, *34*, 20791–20829. [CrossRef]
22. Rahman, C.M.; Mohammed, H.M.; Abdul, Z.K. Multi-Objective Group Learning Algorithm with a Multi-Objective Real-World Engineering Problem. *Appl. Soft Comput.* **2024**, *166*, 112145. [CrossRef]
23. Khodadadi, N.; Abualigah, L.; Mirjalili, S. Multi-Objective Stochastic Paint Optimizer (MOSPO). *Neural Comput. Appl.* **2022**, *34*, 18035–18058. [CrossRef]
24. Lin, W.; Yu, D.Y.; Zhang, C.; Liu, X.; Zhang, S.; Tian, Y.; Liu, S.; Xie, Z. A Multi-Objective Teaching−learning-Based Optimization Algorithm to Scheduling in Turning Processes for Minimizing Makespan and Carbon Footprint. *J. Clean. Prod.* **2015**, *101*, 337–347. [CrossRef]
25. Tawhid, M.A.; Savsani, V. Multi-Objective Sine-Cosine Algorithm (MO-SCA) for Multi-Objective Engineering Design Problems. *Neural Comput. Appl.* **2019**, *31*, 915–929. [CrossRef]
26. Yang, Z.; Hu, X.; Li, Y.; Liang, M.; Wang, K.; Wang, L.; Tang, H.; Guo, S. A Q-Learning-Based Improved Multi-Objective Genetic Algorithm for Solving Distributed Heterogeneous Assembly Flexible Job Shop Scheduling Problems with Transfers. *J. Manuf. Syst.* **2025**, *79*, 398–418. [CrossRef]
27. Zhang, Z.; Tang, Q.; Zhang, L.; Li, Z.; Cheng, L. A Q-Learning-Based Multi-Population Algorithm for Multi-Objective Distributed Heterogeneous Assembly No-Idle Flowshop Scheduling with Batch Delivery. *Expert Syst. Appl.* **2025**, *263*, 125690. [CrossRef]
28. Zheng, J.; Chen, S. A Q-Learning Multi-Objective Grey Wolf Optimizer for the Distributed Hybrid Flowshop Scheduling Problem. *Eng. Optim.* **2024**, *57*, 2609–2628. [CrossRef]
29. Zhang, Z.; Shao, Z.; Shao, W.; Chen, J.; Pi, D. MRLM: A Meta-Reinforcement Learning-Based Metaheuristic for Hybrid Flow-Shop Scheduling Problem with Learning and Forgetting Effects. *Swarm Evol. Comput.* **2024**, *85*, 101479. [CrossRef]
30. Huang, Y.; Guo, Y.; Chen, G.; Wei, H.; Zhao, X.; Yang, S.; Ge, S. Q-Learning Assisted Multi-Objective Evolutionary Optimization for Low-Carbon Scheduling of Open-Pit Mine Trucks. *Swarm Evol. Comput.* **2025**, *92*, 101778. [CrossRef]
31. Ebrie, A.S.; Kim, Y.J. Reinforcement Learning-Based Multi-Objective Optimization for Generation Scheduling in Power Systems. *Systems* **2024**, *12*, 106. [CrossRef]
32. Zheng, R.; Zhang, Y.; Sun, X.; Yang, L.; Song, X. A Reinforcement Learning-Assisted Multi-Objective Evolutionary Algorithm for Generating Green Change Plans of Complex Products. *Appl. Soft Comput.* **2025**, *170*, 112660. [CrossRef]

33. Qi, R.; Li, J.; Wang, J.; Jin, H.; Han, Y. QMOEA: A Q-Learning-Based Multiobjective Evolutionary Algorithm for Solving Time-Dependent Green Vehicle Routing Problems with Time Windows. *Inf. Sci.* **2022**, *608*, 178–201. [CrossRef]
34. Huy, T.H.B.; Nallagownden, P.; Truong, K.H.; Kannan, R.; Vo, D.N.; Ho, N. Multi-Objective Search Group Algorithm for Engineering Design Problems. *Appl. Soft Comput.* **2022**, *126*, 109287. [CrossRef]
35. Adam, S.P.; Alexandropoulos, S.-A.N.; Pardalos, P.M.; Vrahatis, M.N. No Free Lunch Theorem: A Review. In *Approximation and Optimization: Algorithms, Complexity and Applications*; Demetriou, I.C., Pardalos, P.M., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 57–82, ISBN 978-3-030-12767-1.
36. Talatahari, S.; Bayzidi, H.; Saraee, M. Social Network Search for Global Optimization. *IEEE Access* **2021**, *9*, 92815–92863. [CrossRef]
37. Zamli, K.Z.; Alhadawi, H.S.; Din, F. Utilizing the Roulette Wheel Based Social Network Search Algorithm for Substitution Box Construction and Optimization. *Neural Comput. Appl.* **2023**, *35*, 4051–4071. [CrossRef]
38. M. Shaheen, A.; El-Sehiemy, R.A.; Hasanien, H.M.; Ginidi, A. An Enhanced Optimizer of Social Network Search for Multi-Dimension Optimal Power Flow in Electrical Power Grids. *Int. J. Electr. Power Energy Syst.* **2024**, *155*, 109572. [CrossRef]
39. Han, H.; Zhang, L.; Yinga, A.; Qiao, J. Adaptive Multiple Selection Strategy for Multi-Objective Particle Swarm Optimization. *Inf. Sci.* **2023**, *624*, 235–251. [CrossRef]
40. Sierra, M.R.; Coello Coello, C.A. Improving PSO-Based Multi-Objective Optimization Using Crowding, Mutation and ε-Dominance. In *Evolutionary Multi-Criterion Optimization*; Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 505–519.
41. Zitzler, E.; Thiele, L. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Trans. Evol. Comput.* **1999**, *3*, 257–271. [CrossRef]
42. Li, Q.; Liu, S.-Y.; Yang, X.-S. Influence of Initialization on the Performance of Metaheuristic Optimizers. *Appl. Soft Comput.* **2020**, *91*, 106193. [CrossRef]
43. Tharwat, A.; Schenck, W. Population Initialization Techniques for Evolutionary Algorithms for Single-Objective Constrained Optimization Problems: Deterministic vs. Stochastic Techniques. *Swarm Evol. Comput.* **2021**, *67*, 100952. [CrossRef]
44. Zitzler, E.; Laumanns, M.; Thiele, L. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*; ETH Zurich, Computer Engineering and Networks Laboratory: Zurich, Switzerland, 2001.
45. Xia, M.; Dong, M. A Novel Two-Archive Evolutionary Algorithm for Constrained Multi-Objective Optimization with Small Feasible Regions. *Knowl.-Based Syst.* **2022**, *237*, 107693. [CrossRef]
46. Gao, M.; Gao, K.; Ma, Z.; Tang, W. Ensemble Meta-Heuristics and Q-Learning for Solving Unmanned Surface Vessels Scheduling Problems. *Swarm Evol. Comput.* **2023**, *82*, 101358. [CrossRef]
47. Zitzler, E.; Deb, K.; Thiele, L. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evol. Comput.* **2000**, *8*, 173–195. [CrossRef] [PubMed]
48. Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E. Scalable Test Problems for Evolutionary Multiobjective Optimization. In *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*; Abraham, A., Jain, L., Goldberg, R., Eds.; Springer: London, UK, 2005; pp. 105–145, ISBN 978-1-84628-137-2.
49. Zhang, Q.; Zhou, A.; Zhao, S.; Suganthan, P.N.; Liu, W.; Tiwari, S. Multiobjective Optimization Test Instances for the CEC 2009 Special Session and Competition. In *Special Session on Performance Assessment of Multi-Objective Optimization Algorithms, Technical Report*; University of Essex: Colchester, UK; Nanyang Technological University: Singapore, 2008.
50. Jain, H.; Deb, K. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach. *IEEE Trans. Evol. Comput.* **2014**, *18*, 602–622. [CrossRef]
51. Ma, Z.; Wang, Y. Evolutionary Constrained Multiobjective Optimization: Test Suite Construction and Performance Comparisons. *IEEE Trans. Evol. Comput.* **2019**, *23*, 972–986. [CrossRef]
52. Narayanan, S.; Azarm, S. On Improving Multiobjective Genetic Algorithms for Design Optimization. *Struct. Optim.* **1999**, *18*, 146–155. [CrossRef]
53. Chiandussi, G.; Codegone, M.; Ferrero, S.; Varesio, F.E. Comparison of Multi-Objective Optimization Methodologies for Engineering Applications. *Comput. Math. Appl.* **2012**, *63*, 912–942. [CrossRef]
54. Deb, K. Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design. *Evol. Algorithms Eng. Comput. Sci.* **1999**, *2*, 135–161.
55. Kumar, A.; Wu, G.; Ali, M.; Luo, Q.; Mallipeddi, R.; Suganthan, P.; Das, S. A Benchmark-Suite of Real-World Constrained Multi-Objective Optimization Problems and Some Baseline Results. *Swarm Evol. Comput.* **2021**, *67*, 100961. [CrossRef]
56. Cheng, F.Y.; Li, X.S. Generalized Center Method for Multiobjective Engineering Optimization. *Eng. Optim.* **1999**, *31*, 641–661. [CrossRef]
57. Steven, G. Evolutionary Algorithms for Single and Multicriteria Design Optimization. *Struct. Multidisc. Optim.* **2002**, *24*, 88–89. [CrossRef]

58. Dhiman, G.; Kumar, V. Multi-Objective Spotted Hyena Optimizer: A Multi-Objective Optimization Algorithm for Engineering Problems. *Knowl.-Based Syst.* **2018**, *150*, 175–197. [CrossRef]
59. Zhang, H.; Peng, Y.; Hou, L.; Tian, G.; Li, Z. A Hybrid Multi-Objective Optimization Approach for Energy-Absorbing Structures in Train Collisions. *Inf. Sci.* **2019**, *481*, 491–506. [CrossRef]
60. Tian, Y.; Cheng, R.; Zhang, X.; Cheng, F.; Jin, Y. An Indicator-Based Multiobjective Evolutionary Algorithm with Reference Point Adaptation for Better Versatility. *IEEE Trans. Evol. Comput.* **2018**, *22*, 609–622. [CrossRef]
61. Zhu, Q.; Zhang, Q.; Lin, Q. A Constrained Multiobjective Evolutionary Algorithm with Detect-and-Escape Strategy. *IEEE Trans. Evol. Comput.* **2020**, *24*, 938–947. [CrossRef]
62. Zhang, K.; Xu, Z.; Yen, G.G.; Zhang, L. Two-Stage Multiobjective Evolution Strategy for Constrained Multiobjective Optimization. *IEEE Trans. Evol. Comput.* **2024**, *28*, 17–31. [CrossRef]
63. Fan, Z.; Li, W.; Cai, X.; Li, H.; Wei, C.; Zhang, Q.; Deb, K.; Goodman, E. Push and Pull Search for Solving Constrained Multi-Objective Optimization Problems. *Swarm Evol. Comput.* **2019**, *44*, 665–679. [CrossRef]
64. Tian, Y.; Cheng, R.; Zhang, X.; Jin, Y. PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization [Educational Forum]. *IEEE Comput. Intell. Mag.* **2017**, *12*, 73–87. [CrossRef]

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