



Research article

Study on multi-objective particle swarm optimization algorithm combining multiple strategies

Zhe Luo^{1,*}, Guang-Yu Zhu², Zhi-feng Yang³, Si-jie Wu² and Xin-Yu Wang¹

¹ College of Intelligent Manufacturing, Hunan Institute of Science and Engineering, Yongzhou 425199, China

² School of Mechanical Engineering and Automation, Fuzhou University, Fuzhou 350108, China

³ Engineering Training Center, Putian University, Putian 351100, China

* **Correspondence:** Email: Lzxky@huse.edu.cn.

Abstract: To overcome premature convergence and the diversity problem in multi-objective optimization, a multi-strategy multi-objective particle swarm optimization algorithm was proposed in this study. In the MS-MOPSO algorithm, two complementary sub-swarms, which were based on PSO and GWO optimization techniques, were used. Two sub-swarms were optimized by applying different update rules, but they exchanged information through an external archive based on the Pareto dominance concept. In the proposed algorithm, the solutions in the archive were determined through the Pareto dominance principle, and the permutation-averaged discrete Fréchet distance of each normalized objective vector to an ideal profile obtained using the external archive was utilized for the leader selection process. To improve the local search of the solutions without changing the position and meaning of continuous decision vectors, three different neighborhood concepts, which included single-variable, subset, and full vector perturbations, were used. This proposed method was applied to three objective DTLZ1 to DTLZ7 test functions and was compared with MOPSO, MOGWO, and SPEA2 methods. According to the results, good convergence was found for DTLZ2 and DTLZ4 test functions, while diversity and coverage were achieved for some test functions, even though the proposed approach was not superior to other approaches for all test functions. From the results, it can be seen that using the proposed approach for continuous multi-objective optimization is promising.

Keywords: MOO; PSO algorithm; GWO; individual selection mechanism; variable neighborhood search

Mathematics Subject Classification: 90C29

1. Introduction

1.1. Background

Multi-objective optimization problems (MOPs) are everywhere in engineering, industrial, and computational settings, where you have to optimize several objectives that, most of the time, conflict with each other. The main challenge of the problem is that the objectives conflict with each other, and, therefore, using Pareto optimal solutions (POS), you need not a single optimal solution but a series of well-distributed POS. Traditional optimization algorithms are greatly struggling when it comes to MOPs, particularly if the problem has many objectives or the search space is very nonlinear. Swarm intelligence (SI) algorithms have been inspired by the behavior of animal groups, and now they are considered to be one of the most effective approaches to handle MOPs by providing a good equilibrium between exploration and exploitation of the hypothesis space, which is a very important aspect when dealing with MOPs, as stated in [1]. Of all the different SI algorithms, PSO has emerged as the most promising algorithm that can effectively deal with complicated search spaces, including those in high-dimensional spaces, mostly because of its simplicity and having good global search ability. However, basic PSO and its direct adaptation to MOO, known as basic MOPSO, often fail to provide good optimization capabilities, including premature convergence, lack of diversity, and difficulties in maintaining a well-distributed Pareto front. Moreover, however, different studies have attempted to solve these issues by combining the basic PSO with various strategies, including adapting it to multi-objective optimization scenarios, to enhance convergence rates, diversity maintenance, and avoidance of stagnation during optimization. For example, an MOO algorithm for energy storage optimization using a hybrid PSO with adaptive grid crowding and dynamic nonlinear weights was reported to provide better uniformity of the Pareto front and convergence rates compared to basic PSO, MOPSO, and NSGA-II algorithms [2].

The research of hybrid and adaptive approaches in MOPSO has been identified as an emerging research area of great significance. An emergent school of thought in the research of multi-objective PSO variants advocates for the incorporation of external methods such as dual search methods, dynamic population structures, and archive update methods to overcome the inherent limitations of single-strategy variants of PSOs. As an illustration, a newly introduced dual search cooperative PSO algorithm, referred to as MOCPSO, utilizes two distinct methods to enhance the efficiency of the search method. The new algorithm has exhibited competitive results for complex problems in the context of MOPs [3,4]. Similarly, dynamic population structures of MOPSO variants have been reported to retain diversity while enhancing convergence by dynamically adjusting the population sizes depending on the search process, thereby overcoming the problem of premature convergence of the search agents [5]. Nonetheless, achieving the best possible balance between convergence rates and diversity maintenance is a fundamental problem of multi-objective swarm optimization methods. To take advantage of external methods to improve the search method efficiency of PSOs, a few hybrid variants of PSOs have been created. As an illustration, an attempt has been made to hybridize the Fireworks Algorithm (FA) with the multi-objective PSO to enhance diversity maintenance and prevent homogenization of the population, thereby enhancing robustness and flexibility of the search agents in complex search spaces [6].

The combination of these two algorithms in this research is necessary because of the features of the update mechanisms of both of them. PSO provides movement of particles based on velocity, as

well as fast exchange of information. On the other hand, GWO depends on three selected leaders from the archive for creating various directions of searching. Swarms are evolving independently; however, they have a common archive, which means that any found solution by either of the two search algorithms can lead the evolution of both sub-swarms further.

1.2. Related works

In the fields of engineering and industry, MOPs are a regular occurrence where multiple conflicting optimization objectives need to be optimized simultaneously. Traditional optimization techniques have had trouble dealing with the tradeoff between achieving good convergence properties and keeping the diversity of POS in complex optimization problems. PSO has gained popularity for its simplicity and ability for global optimization. MOPSO methods have also faced difficulties in balancing convergence and diversity sustenance. To overcome these difficulties in MOPSO optimization, researchers have tried various hybrid and adaptive optimization methods using multiple optimization techniques in combination with PSO optimization. Han et al. [7] proposed an adaptive multiple selection technique for optimizing the function of multi-swarm optimization methods in PSO. They proposed a hierarchical evolutionary state detection mechanism for adapting leader selection according to different evolutionary states. This optimization method has demonstrated higher performance than MOPSO optimization methods in terms of producing higher-quality solutions for different benchmark problems. Developed on top of multi-swarm combination tactics, Ton, That, et al. [8] suggested MMPSO, which was the natural migration that inspired the algorithm. MMPSO combines advanced methods such as Multi-Vertex Convergent Search (MVCS) and Selective Crossover for Superior Traits (SCST); these methods increase the exploration, convergence, and adaptability of the algorithm greatly. MMPSO came out on top in many different benchmark problems and large-scale tasks; thus, it can be considered a viable competitor to traditional MOPSO algorithms. Furthermore, Cui et al. [9] proposed MOPSO_TA, which strikes a balance between convergence and diversity by introducing two separate archives: One for convergence (CA) and one for diversity (DA). By using indicator-based schemes for convergence and shift-based density estimation for diversity, along with genetic operators, their method outperformed other representative MOO algorithms in handling complex tasks. Wu et al. [10] developed four advanced versions of MOPSO, each of which implemented optimization techniques like Levy flight, logistic chaotic initialization, and angle-based archive management. All versions performed better than MOPSO in solving multi-objective problems. Another significant development in MOO was presented by Liu et al. [11], where they hybridized MOPSO with another powerful optimizer known as GWO to develop a novel optimizer called EIGMOPSO. This optimizer successfully tackled the trajectory optimization and reliability issues of 6-DOF handling robots by dynamically categorizing the population based on fitness, diversity, and stability. Hybridizing SI optimizers has been proven to be an efficient way of improving the overall performance of these optimizers. This is exemplified in the work integrating PSO with GWO, which achieves a better balance of global and local search processes. A third significant advancement in multi-objective optimization was through Jangir et al. [12], who have introduced a Multi-Objective Marine Predator Algorithm (MOMPA) based on nature. Besides, MOMPA was able to solve multi-objective test problems with high performance, owing to snobbish non-dominated classification and density estimation. Building on the hybridization concept, Houssein et al. [13] presented the Multi-Objective Warlus Optimizer (MOWO). MOWO improves the diversity of the solutions using the

“mutate leaders” approach, while the population is updated using the Pareto concept. The experiments confirm that MOWO outperforms other algorithms in the comparison, such as, MOPSO, thus highlighting that the idea of hybridization is very effective for using efficient optimization algorithms to tackle complex problems. Khodadadi et al. [14] presented the Multi-objective Generalized Normal Distribution Optimization (MOGND), where the GND is extended to include the concept of the “archive” of non-dominated solutions, along with the leader selection mechanism. The results show that the MOGND is more efficient compared to some of the MOPSO algorithms, while the hybridization concept is used to overcome the problems of the traditional MOPSO algorithms, as demonstrated using 35 different problems. Quan et al. [15] presented the NSGA-II, where the existing NSGA-II is extended using the Lévy flight and simulated annealing concepts to overcome the problems of the traditional multi-objective evolutionary mechanisms. The outcomes display that the extended NSGA-II is more efficient compared to the NSGA-II, while the hybridization concept is used to overcome the problems of the traditional algorithms, such as the NSGA-II. In addition to the above-mentioned strategies, Song et al. [16] presented the Octopus Optimization Algorithm (OOA) and its multi-objective version, i.e., Multi-Objective OOA (MOOA), based on the behavior of octopuses. OOA and its multi-objective version make use of a population-grouping strategy to maintain diversity and ensure convergence, reflecting the increasing trend of utilizing natural inspirations to develop MOO algorithms. All of these algorithms are in line with the increasing trend of utilizing multiple strategies in conjunction with the PSO algorithm to achieve better optimization results. Rajan [17] developed an integrated technique called HPSGWO, which is a combination of the particle swarm optimization technique and grey wolf optimizer, to estimate the parameters of a photovoltaic panel model. It was found that HPSGWO performed better than many other established meta-heuristic techniques because of the low error rate and greater accuracy and efficiency. Rajeyagari and Nowduri [18] used an RBF Neural Network together with different types of optimization approaches such as meta-heuristic algorithms, to optimize the SOFCs and model the performance of SOFCs. As per the results obtained from the study, it can be seen that the angle of the attack approach was better than PSOGWO, PSO, GWO, MFO, and MVO. In particular, Mrkela and Stanimirović [19] presented their work on the multi-objective maximal covering location problem, with special emphasis on optimizing facility location considering customer preferences. They suggested a multi-objective variable neighborhood search (MO-VNS) algorithm for the problem mentioned above. The algorithm contributes to a major function in the optimization of service networks, such as telecom and waste management networks. In particular, the introduction of the MO-VNS algorithm, along with its variants like Multi-Objective Bounded VNS (MO-BVNS) and Multi-Objective General VNS (MO-GVNS), reflects the increasing trend of utilizing hybrid optimization strategies to address real-world MOPs, similar to the above-mentioned MOPSO algorithms. In a similar direction, Sánchez-Oro et al. [20] presented their work on the multi-objective parallel VNS (Mo-PVNS) algorithm to address the bi-objective obnoxious p-median problem, which holds significant importance in the location of obnoxious facilities like waste disposal plants and chemical plants. This reflects the increasing trend of utilizing MOO algorithms to address real-world optimization problems in the location of facilities. Zhang et al. [21] introduced an approach that was based on the concept of cooperative multi-objective particle swarm optimization and used two search approaches. The proposed algorithm generated impressive results when applied to large-scale problems. These researchers proved the effectiveness of using a cooperative search approach, while we try to extend this idea in different ways. Zhang et al. [22] proposed a parameter-free convergence test by using the idea of dynamic local ideal points and the density function. The

results obtained by Zhang et al. proved that the use of ideal points for guiding was successful; however, we make a further step in this direction through the use of Fréchet distance-based leader evaluation and the CPSO-GWO search technique. Tang et al. [23] presented an optimization algorithm that utilized cooperative sub-swarms for adaptive inertia and learning particle swarm optimization and resulted in better convergence behavior for benchmark functions and path planning. In this work, our objective is to further explore the approach proposed by Tang et al., in which particle swarm optimization and grey wolf optimization are used for multi-objective optimization through external archives. Xu et al. [24] designed a fusion-ranking technique that made use of convergence and diversity measures for elite evaluation and leader selection. Xu et al.'s findings indicated the shortcomings of the sole dependence on Pareto dominance; however, the proposed model in this paper addresses this issue using the Fréchet distance leader ranking technique in collaborative PSO-GWO.

This growing volume of research literature points to the growing trend of improving the basic MOPSO paradigm using hybridized methodologies, multi-swarm strategies, and external optimization schemes. These strategies are geared toward addressing the inherent problems of MOPSO, such as premature convergence and the loss of diversity, while improving the probe of the hypothesis space. Hybrid MOPSO with different optimization algorithms like GWO, Levy flight, and genetic operators has led to remarkable results in the field of MOO, particularly in generating fair solutions.

1.3. Research gaps and contributions

Despite the of many achievements gained by multi-objective particle swarm optimization, there are still some weaknesses that exist. The modern algorithms implement such innovations as adaptive parameter tuning, multiple archives, mutations, population division, or hybridization. Nevertheless, all those features improve one particular stage of the search process, which is why particles behave identically, leading to premature convergence, stagnation of the population, and inability to explore the complex Pareto front. Besides, Pareto dominance criteria cannot be used to measure the performance of non-dominated solutions. Moreover, the leader selection mechanism in the current methods may choose a poorly converged solution. Additionally, current methods do not have any local search procedure, and the neighborhood strategies based on permutations are inappropriate for the decision variables in the continuous domain, as they change the semantic meaning of the variables' positions.

This research is aimed at addressing the above-mentioned gaps and proposing a new algorithm called multi-strategy multi-objective particle swarm optimization. Three major innovative techniques are implemented in this algorithm: The organization of cooperative sub-populations of particle swarm optimization and grey wolf optimization, and the implementation of various update rules with information exchange through a common external archive; the new leader selection technique based on Pareto dominance and Fréchet distance; and continuous variable neighborhood strategy consisting of single variable perturbation, subset perturbation, and vector perturbation.

Key scientific results of this research are listed below:

Multi-objective optimization technique with cooperation of two sub-swarms that are constructed according to the principles of particle swarm optimization and grey wolf optimization algorithms with different updating rules and communication via an external shared archive.

A leader selection technique based on Pareto-domination and permutation-averaged Fréchet distance is proposed for control of both sub-swarms, minimizing the effects of scaling and permutation on the optimization process.

A continuous variable neighborhood search technique is proposed using perturbation of decision variables according to single-variable, sub-vector, and full vector perturbation of decision variables to improve the quality of local search without changing the semantics of decision variables.

The remainder of this paper is organized as follows: In Section 2, we provide a discussion on the proposed algorithm and its operational principles. The experiments and results obtained will be explained in Section 3, whereas the conclusion, limitations, and future work will be provided in Section 4.

2. Realization of the particle swarm optimization algorithm combining strategies

2.1. Swarm optimization algorithm (PSO)

PSO was first proposed by Kennedy and Eberhart [25]. The idea behind the SI-based optimization method was gathered from the collective trait of birds or fish, in which the swarm's cooperation leads to finding optimal solutions in the environment. The SI-based optimization method proceeds through the imitation of particles traveling in a multidimensional search space, where the particles' individual and group behavior determines the best solution. The swarm comprises particles, with each particle being a potential solution to the optimization problem.

A particle in the swarm can be characterized by three key features: position, velocity, and fitness. Position indicates a possible solution in a multidimensional search space, while velocity specifies the direction and length of the particle's move. The fitness function represents the level at which the possible solution meets the required conditions for the optimization problem. The SI-based optimization method advances through the visualization of a set of particles moving through the multidimensional search space, where the particles' individual moves and the swarm behavior contribute to finding an optimal solution. PSO is one of the most effective SI-based optimization algorithms because of the neatness of the approach, the ease of implementation, and the capability to pinpoint the global optimum through the swarm behavior [26].

2.2. Grey Wolf Optimizer Algorithm (GWO)

GWO is an optimization algorithm created by Mirjalili et al. [27] that is based on the concept of SI; thus, it is a new method of evolutionary (bio-inspired) optimization algorithms. The algorithm is inspired by the social hierarchy of GW groups, simulating the process of hunting and solving complex optimization problems in the process. A GW pack consists of a well-structured social hierarchy that includes alpha, beta, and delta wolves, where alpha wolves are the most knowledgeable and delta wolves are the ones following the leader in the optimization process.

One of the most important advantages of the GWO is its ability to conduct efficient global search operations with the use of the least number of parameters, making the algorithm efficient and easy to use. One of the most talked-about topics surrounding optimization algorithms nowadays has been the GWO. Indeed, it has been an excellent tool in solving almost any kind of optimization problem, which is why the GWO has been widely used and has attracted significant attention from the optimization community. This is because the algorithm is capable of producing the best solutions with the right balance of exploration and exploitation in the hypothesis space, thus qualifying it as an efficient optimization algorithm in solving complex optimization problems, whether discrete or continuous in nature [28].

Because of the hierarchical leader-based update procedure of grey wolf optimization, it is suitable for cooperation with the velocity-based update procedure of the particle swarm optimization. The particle swarm optimization algorithm has good convergence speed and fast exchange of information between the particles; however, at times, particles may converge too soon, imitating the same trajectory. However, the grey wolf optimization has a few leaders: Alpha, beta, and delta, who can use different ways of searching. In addition, the algorithm itself is rather simple and has few parameters of control. Moreover, considering the No Free Lunch theorem, no optimization algorithm is better than any other in any case. Therefore, grey wolf optimization is chosen not because it is supposed to be more successful than any other meta-heuristic but because of its suitability for working with the particle swarm optimization algorithm.

2.3. Co-Evolutionary swarm strategy

In general, traditional population-based optimization algorithms often employ a single SI search mechanism, where all individuals in the swarm follow the same evolutionary pattern during the optimization process. With this traditional mechanism, the swarm population might show uniform behavior during the search process, which can lead to stagnation and degrade the entire efficiency of the search process, as all individuals in the swarm might converge and search around similar points in the search space [29]. To overcome the limitations and drawbacks of traditional SI-based population evolution and optimization, there is a significant trend and emphasis in more recent research on co-evolutionary SI and multi-SI, where multiple SI search mechanisms work together or in parallel. Thus, with such multi-SI, each different swarm can apply its own mechanism of the SI search, which can lead to a drastic increase in the efficiency of the overall SI search process. For example, there are proposals and research on developing adaptive multi-SI, where different sub-swarms can be employed to improve diversity and enhance the overall robustness of the SI optimization process [8].

Research on multi-swarm optimization focuses on the application of different algorithms with different and complementary search modalities in different swarms. PSO is the method that has the feature of a global search, whereas GWO is the one that has the feature of a local search and can achieve fast convergence. The synergy of both methods can bring great benefits in terms of optimization because it can provide diversity and quality solutions. An exterior archive stores all non-dominated solutions generated by both methods, and both methods can choose their leader from this archive. This helps them perform better, as shown in Figure 1.

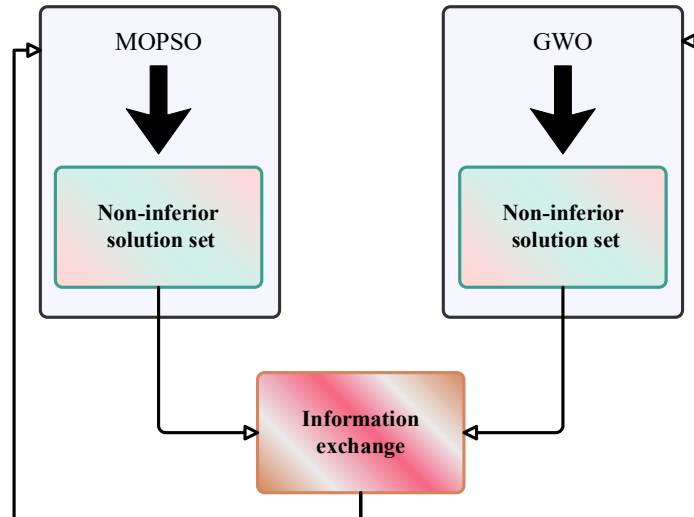


Figure 1. Collaborative search of a swarm.

2.4. Fréchet-distance-based leader selection mechanism

Selection of leaders plays a very significant role in MO-PARTICLE SWARM OPTIMIZATION because the Pareto Dominance Criterion is not able to differentiate the mutually non-dominated solutions present in the external archive. Traditionally, leader solutions in MO-PARTICLE SWARM OPTIMIZATION are selected using a grid density-based roulette wheel selection procedure. While this procedure ensures diversification in the selection of leaders, its stochasticity results in the selection of solutions that might be far off from the ideal area. Thus, in this study, the Fréchet Distance-based Ranking Procedure is used for similarity comparison of solutions.

$$A = \{X^1, X^2, \dots, X^K\} \quad (1)$$

denote the external archive containing KKK non-dominated solutions, and let

$$f(X^i) = (f_1(X^i), f_2(X^i), \dots, f_M(X^i)) \quad (2)$$

The M-objective vector of the i -th archived solution will be. All the objectives are treated as minimization objectives. As different objectives may have different scales, all objectives are normalized individually with respect to the current external archive:

$$f_m^{\min} = \min_{X^i \in A} f_m(X^i), \quad f_m^{\max} = \max_{X^i \in A} f_m(X^i) \quad (3)$$

$$\tilde{f}_{im} = \frac{f_m(X^i) - f_m^{\min}}{f_m^{\max} - f_m^{\min} + \varepsilon}, \quad m = 1, \dots, M \quad (4)$$

where $\varepsilon = 10^{-12}$ prevents division by zero when all archived solutions have the same value for an objective. The archive-based ideal reference vector is;

$$r = (f_1^{min}, f_2^{min}, \dots, f_M^{min}), \quad (5)$$

and its normalized form is consequently

$$\tilde{r} = (0, 0, \dots, 0) \quad (6)$$

To obtain the graph's shape, the objective index acts as the x-axis, and the objective value acts as the y-axis. In the case when $M > 1$, the objective index axis can be represented as:

$$\tau_m = \frac{m-1}{M-1} \quad (7)$$

The normalized objective profile of solution X^i and the corresponding reference profile are represented by the polygonal curves

$$C_i = [(\tau_1, \tilde{f}_{i1}), (\tau_2, \tilde{f}_{i2}), \dots, (\tau_M, \tilde{f}_{iM})] \quad (8)$$

$$C^* = [(\tau_1, 0), (\tau_2, 0), \dots, (\tau_M, 0)] \quad (9)$$

The above representation shows no causality present between the two adjacent objectives. It is merely the presentation of the objective vector as a whole, with all the objectives being taken together when the objective profile is being compared.

Let p_a and q_b denote the a -th and b -th vertices of C_i and C^* , respectively. Their Euclidean distance is:

$$\delta(p_a, q_b) = \|p_a, q_b\|_2 \quad (10)$$

The discrete Fréchet distance is calculated using the following dynamic-programming recursion:

$$D(1, 1) = \delta(p_1, q_1) \quad (11)$$

$$D(a, 1) = \max\{D(a-1, 1), \delta(p_a, q_1)\}, \quad a = 2, \dots, M \quad (12)$$

$$D(1, b) = \max\{D(1, b-1), \delta(p_1, q_b)\}, \quad b = 2, \dots, M \quad (13)$$

Additionally:

$$D(a, b) = \max\{\delta(p_a, q_b), \min[D(a-1, b), D(a-1, b-1), D(a, b-1)]\} \quad (14)$$

The discrete Fréchet distance between the two profiles is therefore:

$$d_F(C_i, C^*) = D(M, M) \quad (15)$$

A smaller value of d_F indicates that the normalized objective profile of a non-dominated solution is closer to the archive-based ideal profile.

• Effect of objective ordering

In calculating the Fréchet distance, the ordering of point sequences is taken into account. Consequently, any ordering of the objectives will influence the distance value calculated despite the

unchangeable structure of the objective vector. To eliminate the effect of the arbitrary objective numbering, this implementation evaluates all possible objective permutations for the three-objective DTLZ problems.

Let Π_M be the set of all permutations of the M objectives. For a permutation $\pi \in \Pi_M$, let C_i^π , and $C^{*,\pi}$ denote the candidate and reference curves constructed using the same permuted objective order. The final leader-selection score is defined as:

$$S_i = \frac{1}{|\Pi_M|} \sum_{\pi \in \Pi_M} d_F(C_i^\pi, C^{*,\pi}) \quad (16)$$

For this experiment, when there are three objectives, $\Pi_3 = 3! = 6$, thus all six cases are considered. In this way, it will be ensured that the result is independent of the initial labeling of f_1, f_2 , and f_3 . However, when there are multiple objectives, one may consider a fixed number of permutations chosen randomly.

After obtaining the value of S_i for all archived solutions, then all archived solutions are sorted according to increasing values. The solution that has the minimum value is selected as the best solution of the PSO sub-swarm. The first three highest-ranked solutions are taken as α , β , and δ of the GWO sub-swarm. If there is a tie between any two solutions with respect to the Fréchet score of 10^{-12} , then the one that has a larger crowding distance is selected.

The use of this method takes place when the Pareto-dominance filter is used. Thus, Pareto dominance determines whether a solution will be put into the archive or not, and the Fréchet number plays the role of an extra factor in evaluating non-dominated solutions.

For the calculation of the discrete Fréchet distance between two curves having M number of vertices using an archive of K elements, $O(M^2)$ time is required. The overall time required for the whole permutation method is equal to:

$$O(KM! M^2) \quad (17)$$

and not including $O(KM)$ for the normalizing process and $O(K \log K)$ for sorting. In this case, since $M = 3$, only six permutations are needed.

Algorithm 1. Fréchet-distance-based leader selection.

Input:

External archive A containing K non-dominated solutions
 Number of objectives M
 Numerical tolerance ε

Output:

PSO leader gbest
 GWO leaders $x\alpha$, $x\beta$, and $x\delta$

1. For each objective $m = 1, \dots, M$:
 Calculate f_{m_min} and f_{m_max} from the current archive.
 2. For each archived solution x_i :
 Normalize each objective using Eqs. (3)–(4).
 3. Construct the ideal and candidate profiles using Eqs. (5)–(9).
 4. Calculate the discrete Fréchet distance using Eqs. (10)–(15), and calculate the permutation-averaged score using Eq. (16).
 For the three-objective experiments, ΠM contains all six permutations.
 5. For each archived solution x_i :
 Set $S_i = 0$.
 For each permutation π in ΠM :
 Construct C_i^π and C_*^π .
 Calculate $d_F(C_i^\pi, C_*^\pi)$ using Eqs. (10)–(15)
 $S_i = S_i + d_F(C_i^\pi, C_*^\pi)$.
 End for
 $S_i = S_i / |\Pi M|$.
 End for
 6. Rank the archived solutions in ascending order of S_i .
 Resolve score ties using descending crowding distance.
 7. Select:
 gbest = first ranked solution;
 $x\alpha$ = first ranked solution;
 $x\beta$ = second ranked distinct solution;
 $x\delta$ = third ranked distinct solution.
 8. Return gbest, $x\alpha$, $x\beta$, and $x\delta$.
-

2.5. Continuous variable neighborhood search

However, due to the nature of the decision variables in DTLZ test problems, which are real-coded, the variables may contribute differently to the objective function at each position. Therefore, permutation-based operators like SWAP, INSERT, and INVERSE are not considered in the proposed algorithm, and only three neighborhoods that increase perturbation in steps are considered to generate the neighbor solutions.

A neighbor solution for a given solution can be obtained by making some perturbations to the selected decision variables within their permissible range. The first neighborhood makes a perturbation to one decision variable that is selected randomly to obtain a small neighborhood. The second

neighborhood makes a perturbation to a randomly selected group of decision variables to get a medium perturbation, and in the third neighborhood, all the decision variables are perturbed.

The general perturbation rule is:

$$x'_j = \text{clip}(x_j + \rho_k(u_j - l_j)r_j, l_j, u_j) \quad (18)$$

where l_j and u_j represent the lower and upper bounds, respectively, of the j th variable; r_j is randomly selected from the interval -1 and 1 ; and ρ_k determines the size of perturbation for the k th neighborhood. The size of the neighborhoods has been decided in such a way that the first structure exploits the problem space, while the other two explore.

The three neighborhoods are defined as follows:

- **N1—single-variable perturbation:** One randomly selected variable is modified.
- **N2—subset perturbation:** A randomly selected subset of variables is modified.
- **N3—full-vector perturbation:** All decision variables are modified simultaneously.

The probability of occurrence of each structure is identical, i.e., $1/3$, $1/3$, and $1/3$. If the neighboring structure dominates the incumbent, then it will be accepted; otherwise, it will be rejected if the neighboring structure is dominated by the incumbent and will be accepted with probability 0.5 if neither dominates the other.

Acceptance criterion. If the solution dominates the current solution, then it is accepted; on the other hand, if the current solution dominates the new one, then it is rejected. In the event that neither dominates the other, the solution is accepted with a probability of 0.5 . This kind of VNS maintains the semantics of the decision variables, and therefore it is appropriate for continuous DTLZ problems.

The local investigation component of the optimization algorithm is crucial in deciding the overall performance of the optimization approach. In the overall procedure of optimization, the efficiency of the local search is significant in determining the quality of the overall solutions generated by the mechanism, particularly for problems where the objective function space is complex. A major factor influencing the efficiency of the local search is the neighborhood structure. Essentially, the neighborhood structure defines a mechanism whereby a given solution is modified to generate new candidate solutions. In the context of the Variable Neighborhood Search (VNS), the neighborhood structure is modified in the overall process of the algorithm, thus enhancing the possibility of reaching the global optimum compared to the situation where a constant neighborhood structure is used. When the algorithm changes the neighborhood structure, it lowers the chance of getting stuck in local optima and improves the thoroughness of the search in the solution space [30].

The classic variable neighborhood search algorithms make use of operators like SWAP, INSERT, and INVERSE. The descriptions of these operators will be given briefly further below to establish the distinction between the classic neighborhoods and the continuous perturbation neighborhoods used in the novel approach. Nevertheless, it should be noted that the use of SWAP, INSERT, and INVERSE operators is not common for the algorithm proposed since swapping the places of real-valued decision variables changes their meanings in continuous functions.

Figure 2 illustrates the steps involved in the proposed approach of the continuous variable neighborhood search. The next step is performed for each of the selected solution when the VNS algorithm is executed randomly with the probability p_{VNS} .

One of the three continuous neighborhood structure approaches, namely single variable change, subset change, or complete vector change, is selected at random. In this situation, a solution is created

in the variable space and is evaluated. When the solution dominates the current solution, the current solution is replaced with the new solution. However, if the current solution dominates the new solution, then the new solution will be rejected. On the other hand, if neither solution dominates the other one, the new solution will be accepted with probability 0.5.

Figure 2 summarizes the activation, neighborhood selection, candidate generation, evaluation, and acceptance stages of the continuous VNS procedure.

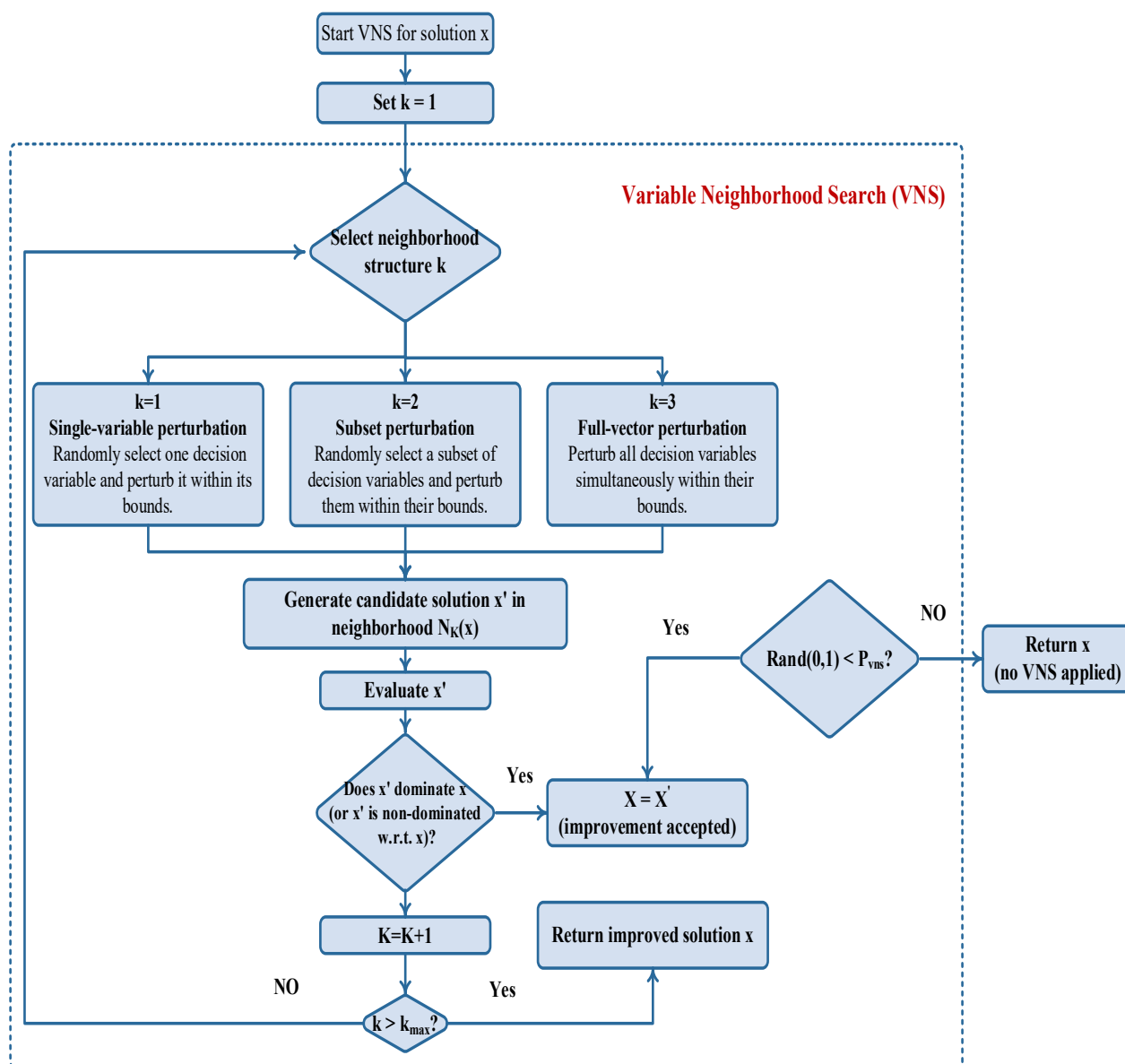


Figure 2. Flowchart of the continuous variable neighborhood search procedure used in MS-MOPSO.

While permutation-based neighborhood generation involves modification of values as well as positions of selected continuous variables, the proposed algorithm only modifies only values without

changing positions. Hence, the proposed method can be applied for real-valued encoding of DTLZ benchmark problems.

Nevertheless, though these classical operators may help to resolve the problem using permutations successfully, they are not applicable when dealing with real-valued decision vectors. That is why the proposed algorithm employs the following neighborhoods based on perturbation.

2.5.1. Continuous neighborhood structures used in the proposed algorithm

The proposed continuous variable neighborhood search applies three neighborhoods. The first neighborhood will be the one in which there is perturbation of one of the decision variables chosen randomly. The second neighborhood will be the one where there is perturbation of several decision variables chosen randomly, and the third neighborhood is that where there is perturbation of all the decision variables. The three neighborhoods will lead to an increasing degree of perturbation, but there is no change in the place and importance of the decision variables. Each of the neighborhoods has a probability of being selected of $1/3$.

2.6. *Process of multi-strategy multi-objective particle swarm optimization algorithm*

The proposed MS-MOPSO algorithm contains two search populations. One search population is the PSO search population, while the other is the GWO search population. Both of these search populations utilize different update rules; however, cooperation between both of these populations is achieved through a common external archive that contains all non-dominated solutions of both of these search populations.

Figure 3 depicts the complete process of the proposed algorithm, beginning with the initialization of parameters, populations, and archive. Following this, the particle swarm and grey wolf swarms are separately evolved with the help of leaders having shared knowledge from the external archive. Next, a continuous variable neighborhood search is conducted after sub-swarm evolution, followed by archive merging, Pareto filtering, and truncation. The process goes on until the desired number of function evaluations is reached, and the final output is an archive [31].

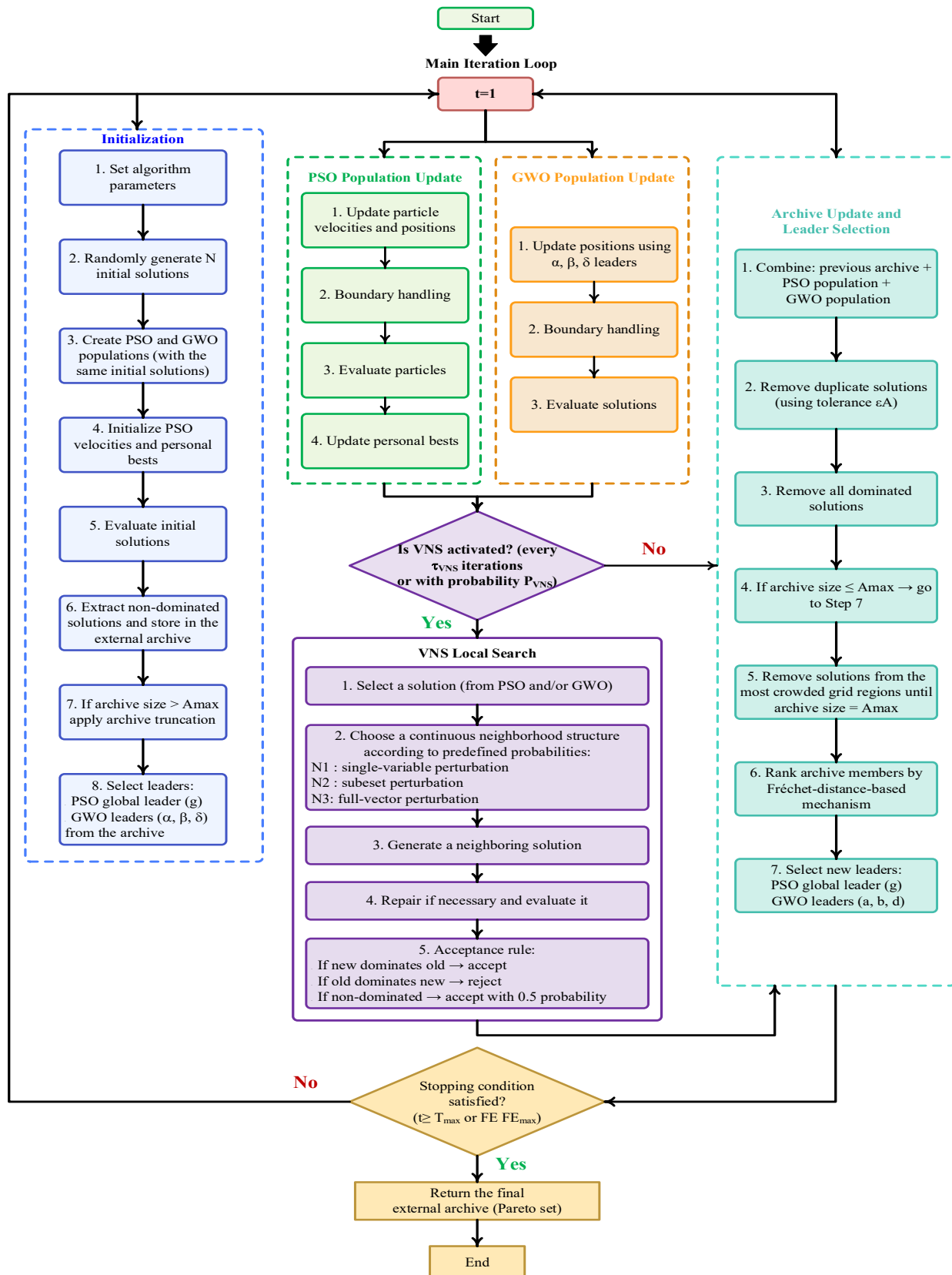


Figure 3. Workflow of the proposed multi-strategy multi-objective particle swarm optimization algorithm.

2.6.1. Initialization and Solution encoding

Initialization of algorithm parameters, decision variable bounds, population size, archiving threshold value, and stopping criteria happens at the beginning of each experiment. A set of N feasible real-coded solutions is generated randomly within the decision variable bounds. These same initial solutions are copied for PSO and GWO population initialization. By doing this, both searches can commence from the same search state and proceed independently after this initialization step.

All solutions have an associated decision vector of $x = (x_1, \dots, x_D)$, where $x_j \in [l_j, u_j]$. The initial population is generated based on a uniform distribution in the range of decision variables, whereas the initial position of the particle swarm and the grey wolf sub-swarm particles is initialized as being equal. The velocities of the particles are initially set to the predefined ranges of velocities, and the infeasible variables due to any update become feasible variables.

The velocity vectors and personal best solutions for each individual in PSO are initialized by this step. Fitness evaluation of the initial solutions happens first. After that, using the Pareto domination idea, the non-dominated solutions obtained from this are archived. In case the initial non-dominated solution set becomes larger than the archive threshold value, the archiving procedure takes place.

The decision vector for each solution is represented by the real vector $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$, where D is the number of decision variables, and $x_{ij} \in [l_j, u_j]$. In this paper, in all the test functions in the DTLZ set of test functions, the decision variables lie in the range of $[0,1]$. The population size remains constant at 100, and the maximum size of the external archive will also be kept at 100. Initially, the solutions are randomly created.

$$x_{ij} = l_j + \text{rand}(0,1) (u_j - l_j) \quad (19)$$

where $\text{rand}(0,1)$ stands for the randomly generated value from the uniform distribution. The same initial population is then used in the particle swarm and grey wolf optimization sub-swarms so that both of them can start their search process from the same initial point. The initial velocities of all particles are assigned to zero, and the personal best values of particles are also assigned to the initial value. The initial solutions are checked only once, and the solutions that are not dominated are added into the common external archive. If the number of non-dominated solutions in the initial solutions is greater than the size of the archive, then the adaptive grid truncation method, as described in Section 2.6.4, is carried out.

2.6.2. Cooperative updating of the two populations

Both the populations in PSO and GWO evolve independently in every iteration.

Particles within the PSO population update their velocities and positions according to the personal best solution and a global leader selected from the external archive. The global leader is selected by following the Fréchet distance-based algorithm presented in Section 2.4. Infeasible decision variable values are mapped back to the nearest feasible values after the position update of particles. The objective's function values and personal best solutions are updated subsequently. In the case of the GWO population, the three highest ranked solutions from the external archive are regarded as alpha, beta, and delta leaders, respectively. Positions of members of the GWO population are updated, taking into account the above-mentioned three leaders. Any boundary constraint violation is resolved before evaluating the objective function. No direct cooperation between the two populations exists in terms

of exchanging their positions/velocities. The cooperation of the two populations is established via the shared archive. All solutions generated by the two populations are fed into the external archive at the end of every iteration. This means that an excellent solution found by any of the two populations can become a leader for both populations in the next iteration.

2.6.3. Variable neighborhood search

The variable neighborhood search algorithm is carried out after the standard PSO/GWO updating algorithms but prior to the updating of the external archive.

The Variable Neighborhood Search algorithm is applied after the Particle Swarm and Grey Wolf population updates and before the External Archive updates. When the Variable Neighborhood Search is implemented, one of the three neighborhoods, either single variable neighborhood, subset neighborhood, or full vector neighborhood perturbation, is selected randomly with equal probability. If the solution obtained through Variable Neighborhood Search violates any of the bounds of decision variables, then the solution is fixed and evaluated. The obtained solution is accepted if it dominates the present solution; rejected if the current solution dominates the candidate solution; and otherwise accepted with probability 0.5.

Table 1 shows VNS activation probability, three equivalent neighborhood structures, and their sizes of perturbation. The solutions are accepted by Pareto dominance, and two mutually non-dominated solutions are accepted with the same probability (0.5). Boundary violation repair is employed, and all VNS invocations contribute to the computation budget.

Table 1. VNS neighborhood-selection and acceptance settings.

Item	Symbol/condition	Description	Value/action
VNS activation probability	p_{VNS}	Probability of applying VNS to a selected solution	0.20
Single-variable perturbation	N_1	One randomly selected variable is modified	Selection probability 1/3; $\rho_1 = 0.05$
Subset perturbation	N_2	A randomly selected subset of variables is modified	Selection probability 1/3; $\rho_2 = 0.10$
Full-vector perturbation	N_3	All decision variables are modified	Selection probability 1/3; $\rho_3 = 0.20$
Candidate dominates incumbent	$x' < x$	The candidate is better by Pareto dominance	Accept candidate
Incumbent dominates candidate	$x < x'$	The incumbent is better by Pareto dominance	Reject candidate
Mutually non-dominated	$x' \parallel x$	Neither solution dominates the other	Accept candidate with probability 0.5
Boundary handling	—	Repair any variable outside its allowable interval	Clip to the nearest feasible bound
Evaluation accounting	—	Objective-function evaluations generated by VNS	Included in total evaluation budget

The continuous neighborhood search algorithm is applied after updating the PSO-GWO sub-swarms and before updating the common external archive. The solution vectors produced in each step are selected for the neighborhood search with probability $p_{VNS}=0.20$. At the start of the procedure, one out of the three neighborhoods is chosen randomly with probability:

$$P(N_1) = P(N_2) = P(N_3) = \frac{1}{3} \quad (20)$$

For a selected solution x , a neighboring solution x' is generated according to:

$$x'_j = \text{clip} [x_j + \rho_k(u_j - l_j)r_j, l_j, u_j] \quad (21)$$

where r_j is independently sampled from the uniform distribution over $[-1,1]$, ρ_k is the perturbation scale associated with the neighborhood, N_k , and $\text{clip}(\cdot)$ restricts the perturbed variable to its feasible interval.

The following are the neighborhood structures used: N_1 , a randomly chosen decision variable is slightly modified using $\rho_1 = 0.05$. In N_1 , $[D/3]$ randomly chosen decision variables are slightly modified using $\rho_1 = 0.02$. In N_3 , all decision variables are simultaneously modified using $\rho_1 = 0.20$. The aforementioned neighborhood structures ensure that the roles of the continuous decision variables remain unchanged and do not affect changes in position, unlike permutation operators.

A neighboring solution is accepted if it Pareto-dominates the current solution. However, if the current solution dominates the neighboring solution, the neighboring solution is rejected. When both solutions do not dominate each other, the neighboring solution will be accepted with a probability of 0.5. The function evaluations resulting from the application of the continuous variable neighborhood search become part of the computational effort needed for solving MS-MOPSO problems. In this case, the maximum-iteration termination criterion is used; hence, the importance of taking into account the above-mentioned evaluations.

To decrease dependence of the algorithm on tuning parameters and to do it once, independently of the chosen benchmark, all the parameters applied in continuous variable neighborhood search are fixed for all DTLZ problems. Activation probability was set to $p_{VNS} = 0.20$. Such an activation probability enables us to take advantage of the local search ability of the additional search procedure, but does not let the additional search dominate. Probability distribution over the various neighborhoods is identical, thus avoiding domination of some perturbation pattern. A perturbation magnitude of 0.05, 0.10, and 0.20 indicates a small, medium, and large perturbation, respectively, regarding the range of the corresponding decision variable.

2.6.4. External archive maintenance

The initialization process for the external archive involves the non-dominated solutions from the initial population. This is because at each iteration, the previous archive is added to the solutions generated by the two sub-swarms and the local search algorithm. Redundant and dominated solutions are then removed. However, in cases where the number of non-dominated solutions exceeds the capacity of the archive, the solutions are iteratively removed from the densest regions through the adaptive grid truncation approach.

Let us assume that N represents the size of each sub-swarm, D is the number of decision variables, M is the number of objectives, K is the maximum size of the archive, P is the number of

objective permutations, and N_v is the number of solutions to be evaluated using the variable neighborhood search in one iteration. Moreover updating of positions of particles in PSO and GWOM requires $O(ND)$ computations each, while a continuous variable neighborhood search requires $O(N_v D)$. Let C_f denote the time required to evaluate one objective function. The time required to evaluate the new solutions are $O((2N + N_v)C_f)$.

Normalizing K solutions in the archive requires $O(KM)$ computations. Calculating the discrete Fréchet distance between two objective profiles requires $O(M^2)$ computations; hence, the computation of the permutation-averaged leader requires $O(KPM^2)$, followed by sorting of the solutions in $O(K \log K)$. Let $S = K + 2N + N_v$ be the total number of solutions to be archived. The worst-case computational complexity of archive maintenance using the concept of pairwise Pareto domination is $O(MS^2)$. The computational complexity of one iteration will thus be:

$$O((2N + N_v)D + (2N + N_v)C_f + M(K + 2N + N_v)^2 + KPM^2 + K \log K) \quad (22)$$

For the instances with three objectives, $P = 6$, and thus, the permutation cost does not become expensive. However, it becomes expensive to generate all $M!$ permutations when M increases. Therefore, to solve the many-objective problem instances, it is recommended to keep a fixed set of permutations such that the Fréchet ranking cost will be $O(KP_s M^2)$, where P_s represents the set of permutations taken as samples. The space cost will be $O((2N + K)(D + M))$.

For every step performed, the previously external archive is joined with the populations of particle swarm optimization and grey wolf optimization, as well as the solutions received due to continuous variable neighborhood search. At first, all duplicates with identical decision and objective vectors are removed. Then, Pareto-dominance filtering is done, and dominated solutions are discarded.

If the total number of non-dominated solutions is no larger than the archive size, which equals to 100, then all solutions are retained. Otherwise, the objective space is divided into grids with adaptive partitioning of eight grids for every dimension. Next, the density of each grid cell is determined according to the number of archive members located in it. Should there be too many solutions remaining after this step compared to the required archive size of 100 solutions, then solutions are deleted from the crowded grid cells one by one until the archive size reduces to 100. If there are multiple solutions in a single crowded grid cell, then one solution from them is deleted randomly to avoid any determinism.

The sorted solutions from the archive, sorted in increasing order of their normalized and permutation-averaged Fréchet distance values, serve as leaders for both populations. The solution from the archive with the minimum value of Fréchet distance serves as the leader of the particle swarm optimization sub-swarm. The three first unique solutions from the archive are considered to be the alpha, beta, and delta leaders of the grey wolf optimization sub-swarm, respectively. Moreover, if two solutions from the archive have the same Fréchet distance values, then the one with the maximum crowding distance wins. Because both sub-swarms receive their leaders from the same archive, a good solution found by one of the searches can lead both populations in the next step.

2.7. Multi-Objective mode

In the field of MOO, test problems such as the ZDT and the DTLZ are commonly used as benchmark test cases. The ZDT test problems involve the optimization of two objectives, while the

DTLZ test problems are more versatile and are commonly used in the optimization of high-dimensional MOPs. In addition, the DTLZ test problems can be easily extended to incorporate a higher number of objectives, making it the most commonly used test problem in the field of MOO [32]. DTLZ1 is a standard benchmark from the DTLZ series that is broadly recognized as the initial test for assessing multi-objective optimization algorithms. The objective functions in DTLZ1 are defined in such a way that they illustrate complex relationships between variables and objectives. In this benchmark function, the goal of the optimization is to minimize M objective functions, and n stands for the count of variables. The general form of the objective function and constraints is as follows:

Test case: DTLZ1

The optimization objectives for the DTLZ1 test case are defined as follows:

$$\min f_1(X) = \frac{1}{2} x_1 x_2 \dots x_{M-1} (1 + g(X_M)) \quad (23)$$

$$\min f_2(X) = \frac{1}{2} x_1 x_2 \dots (1 - x_{M-1}) (1 + g(X_M)) \quad (24)$$

$$\min f_{M-1}(X) = \frac{1}{2} x_1 (1 - x_2) (1 + g(X_M)) \quad (25)$$

$$\min f_M(X) = \frac{1}{2} x_1 (1 - x_1) (1 + g(X_M)) \quad (26)$$

where the function $g(X_M)$ is defined as:

$$g(X_M) = 100[|X_M| + \sum_{x_i \in X_M} ((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)))] \quad (27)$$

Subject to: $0 \leq x_i \leq 1$ for $(i=1, 2, \dots, n)$

In this formulation, X_M represents the set of the last k variables in the n -dimensional decision vector, where $k=(n-m+1)$. The function $g(X_M)$ incorporates the last k variables of the decision vector.

In this study, the test case with $M=3$ and $n=10$ is used to evaluate the MS-MOPSO approach's function. The specific configurations of the test cases employed are provided in Table 2.

Table 2. Test case set.

Test case	Objective function (3 objectives)	Constraints
DTLZ2	$f_1(X) = (1 + g(X_M)) \cos(x_1 \pi/2) \cos(x_2 \pi/2)$	$0 \leq x_i \leq 1 \ (i=1, 2, \dots, n)$ $g(X_M) = \sum_{x_i \in X_M} (x_i - 0.5)^2$
	$f_2(X) = (1 + g(X_M)) \cos(x_1 \pi/2) \sin(x_2 \pi/2)$	
	$f_3(X) = (1 + g(X_M)) \sin(x_1 \pi/2)$	
DTLZ3	$f_1(X) = (1 + g(X_M)) \cos(x_1 \pi/2) \cos(x_2 \pi/2)$	Consistent with DTLZ1
	$f_2(X) = (1 + g(X_M)) \cos(x_1 \pi/2) \sin(x_2 \pi/2)$	
	$f_3(X) = (1 + g(X_M)) \sin(x_1 \pi/2)$	

Continued on next page

Test case	Objective function (3 objectives)	Constraints
DTLZ4	$f_1(X) = (1+g(X_M)) \cos(x_1^\alpha \pi/2) \cos(x_2 \pi/2)$	$0 \leq x_i \leq 1 \ (i=1,2,\dots,n)$
	$f_2(X) = (1+g(X_M)) \cos(x_1^\alpha \pi/2) \sin(x_2 \pi/2)$	$g(X_M) = \sum_{x_i \in X_M} (x_i - 0.5)^2$
	$f_3(X) = (1+g(X_M)) \sin(x_1^\alpha \pi/2)$	$\alpha = 100$
DTLZ5	$f_1(X) = (1+g(X_M)) \cos(\theta_1 \pi/2) \cos(\theta_2 \pi/2)$	$\theta_i = \frac{\pi}{4(1+g(X_M))} (1+2g(X_M)x_i)$
	$f_2(X) = (1+g(X_M)) \cos(\theta_1 \pi/2) \sin(\theta_2 \pi/2)$	$0 \leq x_i \leq 1 \ (i=1,2,\dots,n)$
	$f_3(X) = (1+g(X_M)) \sin(\theta_1 \pi/2)$	$g(X_M) = \sum_{x_i \in X_M} (x_i - 0.5)^2$
DTLZ6	$f_1(X) = (1+g(X_M)) \cos(\theta_1 \pi/2) \cos(\theta_2 \pi/2)$	$\theta_i = \frac{\pi}{4(1+g(X_M))} (1+2g(X_M)x_i)$
	$f_2(X) = (1+g(X_M)) \cos(\theta_1 \pi/2) \sin(\theta_2 \pi/2)$	$0 \leq x_i \leq 1 \ (i=1,2,\dots,n)$
	$f_3(X) = (1+g(X_M)) \sin(\theta_1 \pi/2)$	$g(X_M) = \sum_{x_i \in X_M} (x_i)^{0.1}$
DTLZ7	$f_1(X) = x_1$	$g(X_M) = 1 + 9/ X_M \sum_{x_i \in X_M} x_i$
	$f_2(X) = x_2$	$h(f_1, f_1, g) = M - \sum_{i=1}^{M-1} \left(\frac{f_i}{1} + g * p \right)$
	$f_3(X) = (1+g(X_M)) h(f_1, f_1, g)$	$p = (1 + \sin(3\pi f_i))$
		$0 \leq x_i \leq 1 \ (i=1,2,\dots,n)$

3. Experiment and result analysis

First, to check how well the MS and MOPSO algorithms worked, a series of tests was done with DTLZ test problems (17), which are well known as benchmark problems for MOPs. The performance evaluation of the MS-MOPSO algorithm was carried out against MOPSO, MOGWO, and the multi-objective evolutionary algorithm (SPEA2). Three major performance evaluation criteria were used for the evaluation: convergence index DM, diversity index SP, and coverage index C [33].

3.1. Parameter setting

The experimental comparison of the suggested MS-MOPSO algorithm was carried out using MOPSO, MOGWO, and SPEA2. It can be noted that this selection of algorithms is appropriate since each of these three algorithms implements one of the following approaches: particle swarm approach, grey wolf approach, or an evolutionary archive.

The final values of the population size, archive capacity, variable neighborhood search activation probability, perturbation amplitudes, and stopping budget are summarized in Table 3.

Table 3. Parameter setting.

Algorithm	Parameter setting
SPEA2	$P_c=0.7$, $P_m=0.2$, Group=100, REP=100
MOPSO	$\omega=0.5$, $\omega_{damp}=0.5$, $C_1=C_2=1$, nGrid=7, $P_m=0.1$, Group=100, REP=100
MOGWO	nGrid=10, Group=100, REP=100
MS-MOPSO	$\omega=0.6$, $C_1=C_2=1$, $\omega_{damp}=0.9$, $P_m=0.2$, nGrid=8, Group=100, REP=100

In Table 3, Group refers to the population size, while the value REP refers to the maximum external archive size. The values of the parameters related to each of the algorithms have been chosen from the source literature where the algorithms were proposed or from the recommendation of the algorithm authors. The sizes of the populations, the archives, the ranges of the decision variables, and random seeds have been kept identical.

To examine the robustness of the selected parameter settings, a sensitivity analysis was conducted by varying each principal parameter individually while keeping all other parameters unchanged.

3.1.1. Comparison protocol and stopping condition

All algorithms were tested using the same population size, archive size, benchmark parameters, repetition, random seeds, and number of iterations. Therefore, the convergence behavior of all algorithms is illustrated by means of the number of iterations. On the other hand, one iteration of MS-MOPSO consists of updating particle swarm sub-swarm, grey wolf sub-swarm, and neighborhood search of continuous variables. Consequently, the number of the same iterations does not necessarily imply that the number of function evaluations is the same too. The following results should not be viewed as the comparison under the same function evaluations budget.

3.2. Statistical analysis

We could not present statistical information related to the algorithms, criteria, and DTLZ test problems, due to insufficient information. Consequently, this research does not contain any global statistics related to the values of the best, worst, mean, and standard deviation, as well as to Wilcoxon and Friedman tests for all DTLZ test problems. Statistical significance has been taken into account for DM criterion of the DTLZ4 test problem, where 40 separate tests have been conducted for all four algorithms. comparison of four algorithms was carried out using a Kruskal-Wallis test at 0.05 level of significance.

The statistical test was carried out separately for each metric of performance. Measures that were low in value were favored for inversion generation gap, spacing, and error metrics, whereas high values were favored for hypervolume and coverage metrics. The results of the descriptive statistics and significance tests, together with the average ranks of algorithms, are shown in the corresponding tables. The statistical test is more effective than evaluation based on single-run convergence graphs and average performance measures.

Since different solutions can be obtained in separate runs by metaheuristics, the efficiency of every algorithm was estimated with respect to 30 runs of each algorithm. Best, worst, mean, median, and standard deviation were calculated for each benchmark problem and each criterion. Mean and median indicate the average efficiency of the algorithm, whereas standard deviation and the difference

between the best and the worst value illustrate the variability of the algorithm. The lower standard deviation value implies higher stability and consistency of the algorithm's performance.

To conduct a comprehensive analysis of the algorithms' efficiencies, convergence and diversity criteria were used. This fact is explained by the inability of the convergence criterion to completely characterize the quality of the obtained Pareto set. Therefore, the statistics provided below consider not only the proximity of solutions to the reference Pareto set but also their distribution and coverage in objective space. Full descriptive statistics are provided below in the result tables; the best mean values are highlighted.

3.3. Analysis of Results

- (1) The convergence index DM was used to measure the extent to which the non-inferior solution set, which was computed by the approach, approached the true Pareto front. When the DM is small, it shows that the solutions are closer to the true Pareto front, which is a measure of the convergence performance of the algorithm. Figures 4 to 10 show the DM indices for different test cases.

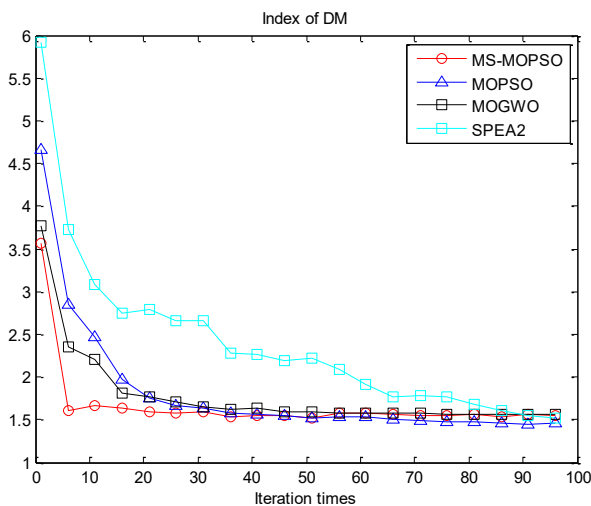


Figure 4. DM Index of DTLZ1.

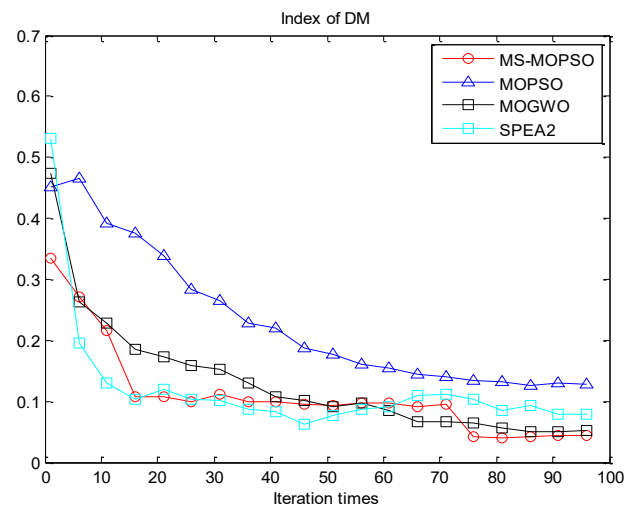


Figure 5. DM Index of DTLZ2.

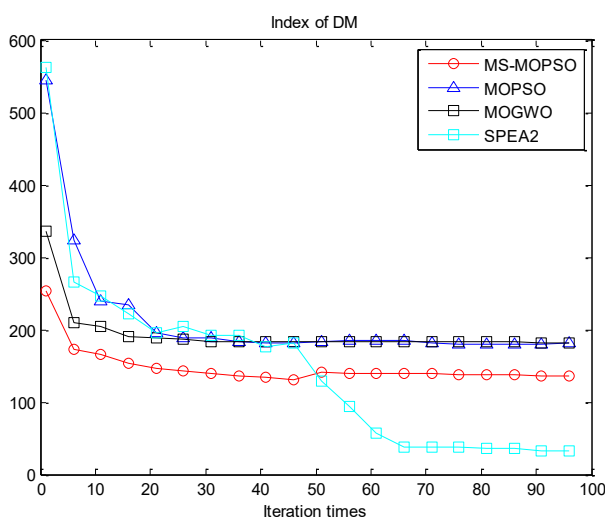


Figure 6. DM Index of DTLZ3.

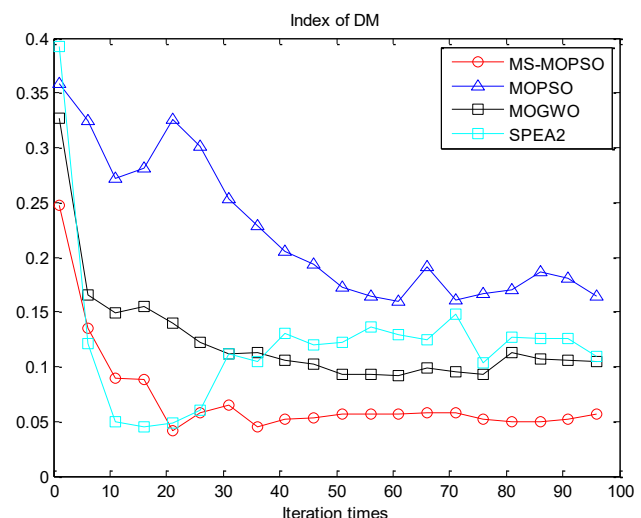


Figure 7. DM Index of DTLZ4.

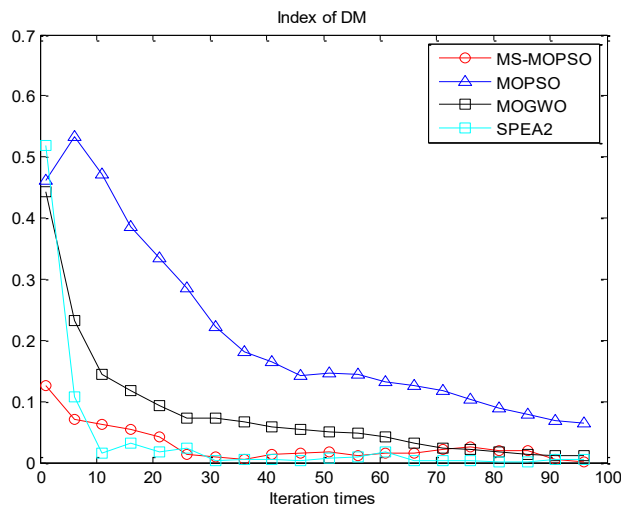


Figure 8. DM Index of DTLZ5.

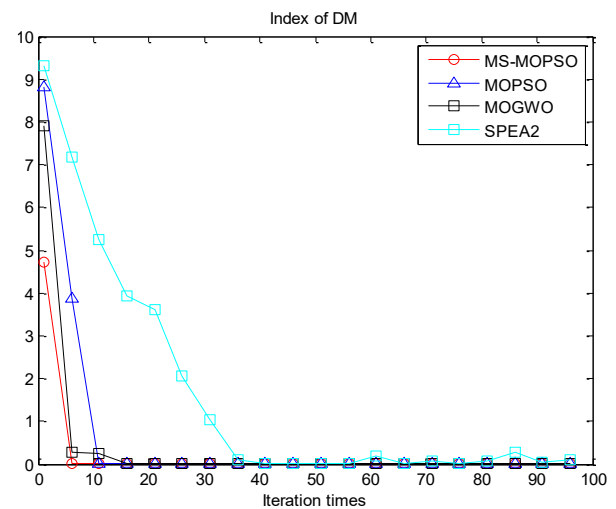


Figure 9. DM Index of DTLZ6.

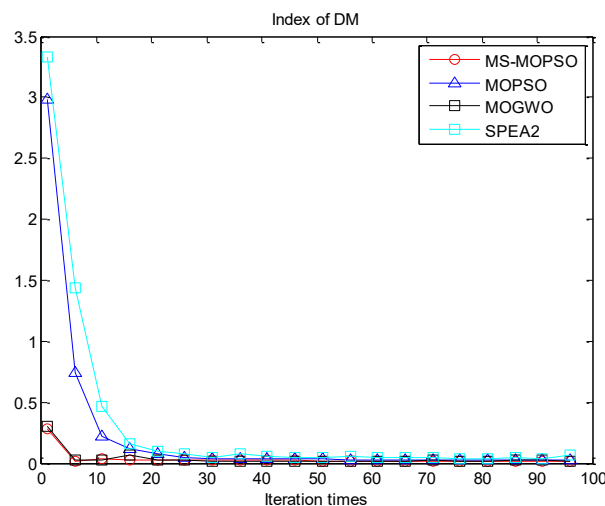


Figure 10. DM Index of DTLZ7.

From the DM as shown in Figures 4 to 10, it can be seen that four algorithms perform equivalently during the iteration process in the DTLZ (1, 5, 6, 7) test cases, but the algorithm in this paper is optimal in converging to the real Pareto in DTLZ2 and DTLZ4, and its convergence in DTLZ3 is only next to SPEA2. In general, the MS-MOPSO algorithm in this paper has certain advantages over the other three algorithms in terms of fast convergence, as the ideal adaptive point in its selection mechanism effectively guides the whole swarm.

Based on the convergence function of the four optimization algorithms, as shown in Figures 4 to 10, it can be noted that all four optimization algorithms, including MS-MOPSO, MOPSO, MOGWO, and SPEA2, obtain similar results for the DTLZ test cases (1, 5, 6, and 7). The convergence of all four optimization algorithms to the Pareto front is almost the same throughout the iteration process. Hence, it can be understood that all four optimization algorithms produce very similar results in terms of convergence to the Pareto front. However, it can be noted that the convergence of MS-MOPSO to the real Pareto front in DTLZ2 and DTLZ4 test cases is better than that of the other three optimization algorithms. Thus, it can be noted that the proposed optimization algorithm, MS-MOPSO, is more

effective in solving optimization problems, especially those characterized by complexities in DTLZ2 and DTLZ4 test cases.

Moreover, it is evident from the graph that in DTLZ3, the convergence of MS and MOPSO to the Pareto front comes only after SPEA2 in the ranking. Therefore, it is argued that the newly proposed optimization algorithm, MS, and MOPSO have a higher problem-solving capability for optimization problems, even though it comes second in DTLZ3.

(2) The diversity index, also known as SP, is a criterion for the distribution characteristics of the objective values attained within the objective space. It is considered an effectiveness measure of the search mechanism employed by the algorithm, where a higher value of SP indicates a better distribution with more diversity in the objective space. Figures 11 to 17 show the SP indices for different test cases.

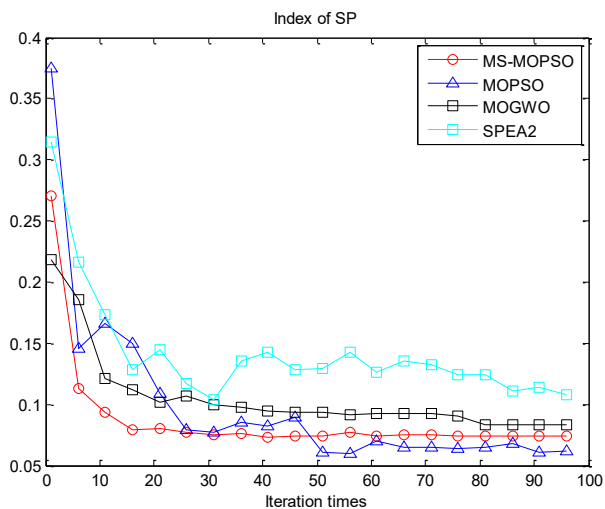


Figure 11. SP Index of DTLZ1.

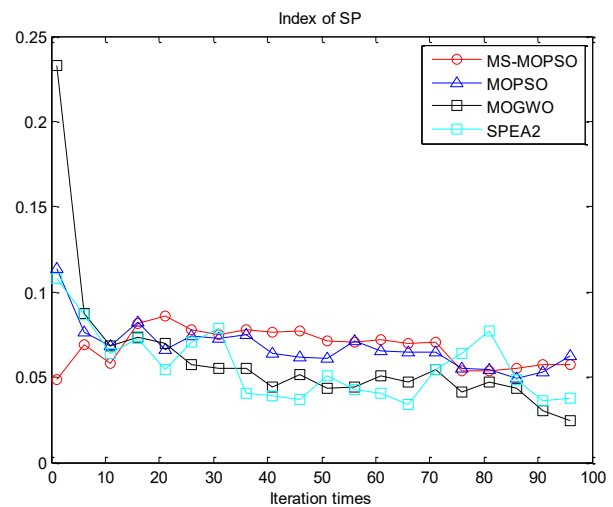


Figure 12. SP Index of DTLZ2.

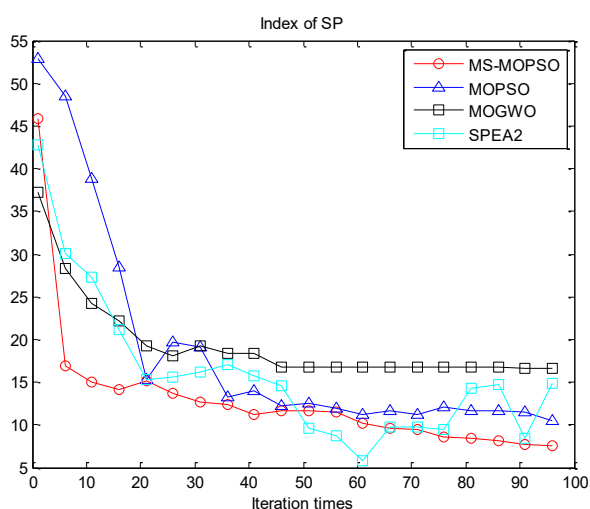


Figure 13. SP Index of DTLZ3.

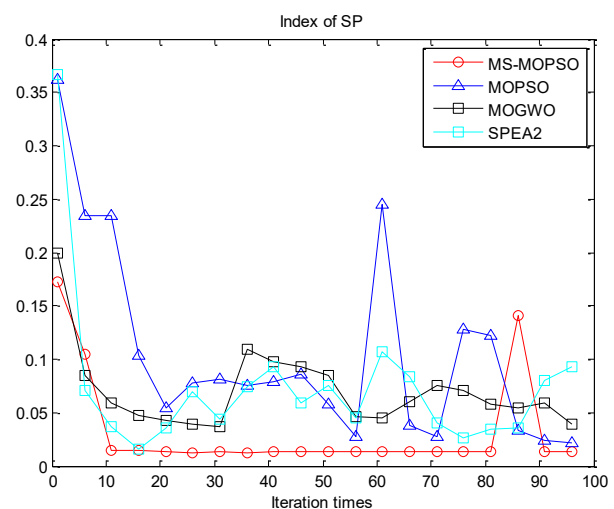


Figure 14. SP Index of DTLZ4.

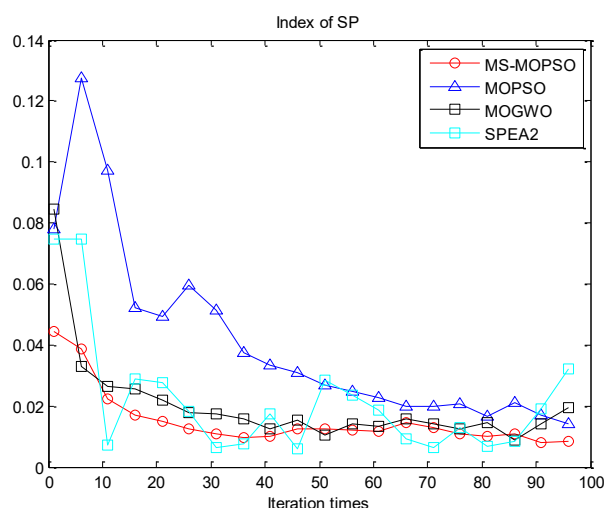


Figure 15. SP Index of DTLZ.

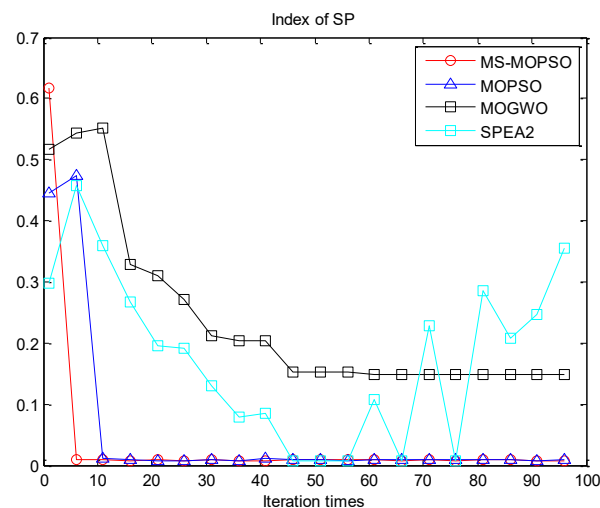


Figure 16. SP Index of DTLZ6.

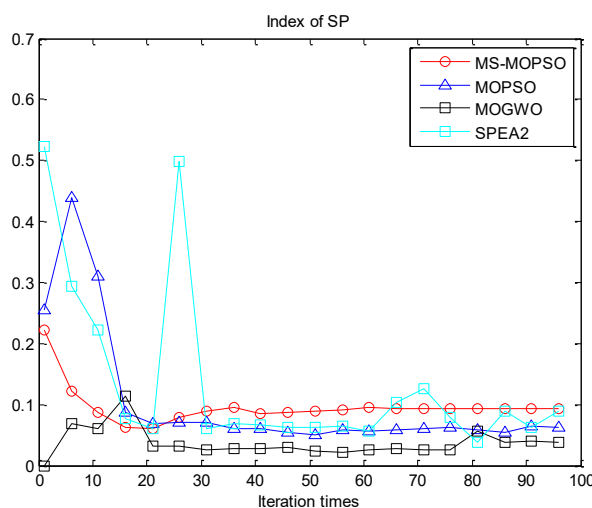


Figure 17. SP Index of DTLZ7.

As can be seen from the distribution diagrams presented in Figures 11 to 17, the MS-MOPSO algorithm attains optimal uniform distribution for four of the seven test case problems (DTLZ1, DTLZ2, DTLZ3, and DTLZ6). This means that for the aforementioned four test cases, the distribution uniformity obtained by the MS-MOPSO algorithm for the Pareto optimal solution set is superior to that obtained by the other three algorithms, MOPSO, MOGWO, and SPEA2. This suggests that the MS-MOPSO algorithm can maintain the diversity and obtain a uniform distribution for the solutions.

However, for the remaining three test cases, DTLZ4, DTLZ5, and DTLZ7, the distribution uniformity obtained by the MS-MOPSO algorithm does not always have a higher value compared to the distribution uniformity obtained by the other three algorithms. This could be due to the guidance mode used by the MS-MOPSO selection mechanism, which, although effective, does not always ensure the best possible distribution uniformity for the solutions.

On the other hand, the MOGWO and MOPSO algorithms have a grid density adaptation mechanism, which increases the probability of finding the optimally distributed solution set and can obtain a better distribution uniformity in some cases. The grid-based methods used by the two

algorithms can ensure a better exploration of the objective space and obtain a competitive distribution uniformity in some cases.

(3) Coverage index (C) acts as an essential performance index to evaluate the coverage of the solution space for distinct Pareto solution sets achieved through optimization techniques. The coverage index provides useful information regarding the variety of solutions obtained by different optimization methods under various test cases. Table 4 shows the C indices for distinct test cases, comparing the performance of MS-MOPSO, MOPSO, MOGWO, and SPEA2.

As shown in Table 4, the function of the MS-MOPSO approach in terms of coverage is consistently good for most of the DTLZ test functions, especially if compared with other algorithms like MOPSO and MOGWO. In many cases, the coverage of the proposed algorithm is the best, implying that the algorithm is capable of exploring the solution space more efficiently and finding diverse POS.

Nevertheless, the suggested approach cannot consistently beat the SPEA2 algorithm when it comes to coverage. For example, in the case of the DTLZ2 and DTLZ5 functions, the coverage of the Pareto solutions by the SPEA2 algorithm is significantly higher. This is because the Pareto dominance approach of the SPEA2 algorithm enables quick identification of the true Pareto front. In these cases, the coverage of the Pareto solutions by the proposed algorithm is good but not as uniform as that of the SPEA2 algorithm.

Table 4. C index.

Test function	C index			
DTLZ1	(MOPSO, MS-MOPSO)	0.13	(MS-MOPSO, MOPSO)	0.35
	(MOGWO, MS-MOPSO)	0.09	(MS-MOPSO, MOGWO)	0.44
	(SPEA2, MS-MOPSO)	0.44	(MS-MOPSO, SPEA2)	0.38
DTLZ2	(MOPSO, MS-MOPSO)	0.025	(MS-MOPSO, MOPSO)	0.73
	(MOGWO, MS-MOPSO)	0	(MS-MOPSO, MOGWO)	0.063
	(SPEA2, MS-MOPSO)	0.013	(MS-MOPSO, SPEA2)	0.55
DTLZ3	(MOPSO, MS-MOPSO)	0.05	(MS-MOPSO, MOPSO)	0.56
	(MOGWO, MS-MOPSO)	0	(MS-MOPSO, MOGWO)	0.98
	(SPEA2, MS-MOPSO)	0.98	(MS-MOPSO, SPEA2)	0.43
DTLZ4	(MOPSO, MS-MOPSO)	0	(MS-MOPSO, MOPSO)	0.28
	(MOGWO, MS-MOPSO)	0	(MS-MOPSO, MOGWO)	1
	(SPEA2, MS-MOPSO)	0	(MS-MOPSO, SPEA2)	0.51
DTLZ5	(MOPSO, MS-MOPSO)	0	(MS-MOPSO, MOPSO)	0.94
	(MOGWO, MS-MOPSO)	0.08	(MS-MOPSO, MOGWO)	0.42
	(SPEA2, MS-MOPSO)	0.07	(MS-MOPSO, SPEA2)	0.29

Continued on next page

Test function	C index			
DTLZ6	(MOPSO, MS-MOPSO)	0.12	(MS-MOPSO, MOPSO)	0.15
	(MOGWO, MS-MOPSO)	0.2	(MS-MOPSO, MOGWO)	0.15
	(SPEA2, MS-MOPSO)	0	(MS-MOPSO, SPEA2)	0
DTLZ7	(MOPSO, MS-MOPSO)	0.04	(MS-MOPSO, MOPSO)	0.21
	(MOGWO, MS-MOPSO)	0	(MS-MOPSO, MOGWO)	0.09
	(SPEA2, MS-MOPSO)	0.01	(MS-MOPSO, SPEA2)	0.38

3.4. Significance test

In solving the resolution of the DTLZ benchmark problem, the function metrics of the proposed MS-MOPSO algorithm show promising results. However, it is not sufficient to conclude that the proposed algorithm outperforms other comparative algorithms in solving the DTLZ benchmark problem. Therefore, to more certainly confirm the function boost of the suggested approach, a significance test should be used. For this purpose, DTLZ4 is used as an instance for to perform the significance test. The convergence index, denoted by DM, is used as the performance metric for statistically analyzing the optimization results of the proposed algorithm and other comparative algorithms, such as SPEA2, MOPSO, MOGWO, and MS-MOPSO. The Kruskal-Wallis test, which is a non-parametric test, is used to evaluate whether there is any substantial difference in the performance of these four optimization algorithms.

This test comes in handy, especially when one has to determine if there is any substantial difference in the performance of several optimization algorithms. In this study, each of the four optimization algorithms is run 40 times for DTLZ4, and then the DM value of each run is collected. A total of 160 data points are collected. Table 5 displays the outcomes of the Kruskal-Wallis test; the test statistic is 21.8, and the corresponding p-value is 0.001. It is clear from the table that the p-value is far less than 0.05. The conclusion drawn from the results is that there exists an algorithm with a distribution of DM values that is different from others for DTLZ4. However, Kruskal-Wallis cannot be used to identify such pairs of algorithms.

Table 5. Kruskal-Wallis test results.

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	130976	3	65488	21.8	0.001
Error	1473070	156	5558.8		
Total	1604047	159			

3.5. Discussion

MOPs are very common in engineering and other fields, where multiple objectives are to be optimized simultaneously. Traditional methods such as PSO often encounter issues of stagnation at early stages and diminishing diversity in the objective space. In this work, a new algorithm, MS-

MOPSO, is proposed, which combines the conventional PSO with GWO and incorporates a new leader selection strategy along with VNS strategies.

The performance of MS-MOPSO seems to be problem-dependent. This could be due to the combination of components such as Fréchet-distance-based leader selection, a shared archive, a cooperative PSO-GWO optimization technique, and a continuous variable neighborhood search. Leader Selection chooses individuals near the ideal objective vector, whereas the archive facilitates the sharing of information between individuals through a neighborhood search with small perturbations. However, since there has been no ablation study of MS-MOPSO, it is not known how much the individual components contribute to the performance.

Furthermore, there are certain limitations of the method : Maintaining multiple sub-swarms, archive, and variable neighborhood search adds complexity to the problem. The costs associated with Pareto sorting and archive management keep increasing as the number of individuals and the archive increases, whereas the permutation-averaged Fréchet distance may not work well for many of objectives. As the number of objectives keeps on increasing, the Pareto dominance relation loses discrimination power, resulting in larger archives and inefficiencies in leader selection. In the case of a multi-dimensional decision space, full-vector perturbation does not seem to be efficient for a partial variable neighborhood search. Last, this research has been carried out for the three objectives' DTLZ1-7 problem, and the performance of other problems is not known.

It can also be argued that the comparison presented herein is not entirely fair from the standpoint of computational effort, as the number of iterations is the same for all of the algorithms tested, while MS-MOPSO has relatively higher computational effort per iteration due to its sub-swarms and local search approach. In future studies, ablation tests, the same number of function evaluations, same random seed, repeated runs, and non-parametric statistical tests will be employed. Further future work will also include test cases for ZDT1-ZDT4 and ZDT6, as well as WFG1-WFG9 with 3, 5, and 10 objective functions, respectively. Up to the point where these tests are carried out, the conclusions must be confined to the 3-objective DTLZ test cases.

4. Conclusion

Our key aim of this work was to create a novel MOO technique that combines several strategies in a single framework, which was named MS-MOPSO. The proposed MS-MOPSO was designed to mitigate premature convergence and loss of diversity by combining a cooperative PSO-GWO search, archive-based leader selection, and continuous variable neighborhood search. To enhance local search, a variable neighborhood search with three different neighborhood structures was also included.

In the evaluation of the technique, several benchmark problems, especially the DTLZ series of problems, were used to compare the performance of the named MS-MOPSO technique in terms of convergence, diversity, and coverage. During the evaluation, the method was pitted against several methods, including MOPSO, MOGWO, and SPEA2.

In the evaluation of the technique, the following results were obtained:

- In the evaluation of the technique for the DTLZ2 and DTLZ4 problems, the MS-MOPSO technique was observed to perform better in terms of convergence, as the solutions obtained were closer to the true Pareto front than those of the existing techniques, MOPSO, MOGWO, and SPEA2.

- In the evaluation of the technique for the DTLZ1, DTLZ2, DTLZ3, and DTLZ6 problems, the diversity index showed that the MS-MOPSO technique provided a more uniform distribution of POS than the existing techniques, MOPSO, MOGWO, and SPEA2.
- In the evaluation of the technique for the DTLZ2, DTLZ5, and DTLZ7 problems, the coverage index showed that the MS-MOPSO technique outperformed the existing techniques, MOPSO and MOGWO, in terms of efficiency in exploring the search space.

The efficiency of the MS-MOPSO algorithm is highly dependent on the nature of the problem. Based on the presented experimental results, the algorithm shows high convergence for DTLZ2 and DTLZ4, while diversity is equal in terms of the variety of test problems. It is necessary to point out that at the moment the study was carried out using three-objective DTLZ test functions only.

Competing of interests

The authors declare no competing interests.

Authorship Contribution Statement

Zhe Luo: Writing-Original draft preparation, Conceptualization, Supervision, Project administration. Guang-Yu Zhu: Methodology. Zhi-feng Yang: Software, Validation. Si-jie Wu: Formal analysis. Xin-Yu Wang: Language review.

Declarations

Not applicable.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Author Statement

The manuscript has been read and approved by all the authors; the requirements for authorship, as stated earlier in this document, have been met, and each author believes that the manuscript represents honest work.

Ethical approval

All authors have been personally and actively involved in substantial work leading to the paper, and will take public responsibility for its content.

Use of Generative-AI tools declaration

The authors declare that they did not utilize any artificial intelligence (AI) tools in the creation of this article.

Project Funding

This project was supported by the Hunan Provincial Social Science Achievement Review Committee Subjects, China (Grant No. XSP26YBC439), the Scientific Research Program of Hunan Provincial Education Department, China (No. 23B0757), the Natural Science Foundation of Hunan Province, China (No. 2023 JJ50071), and the Natural Science Foundation of Fujian Province, China (No. 2023J01256).

References

1. A. K. Singh, A. Kumar, Multi-objective: hybrid particle swarm optimization with firefly algorithm for feature selection with Leaky ReLU, *Discov. Artif. Intell.*, **5** (2025), 192. <https://doi.org/10.1007/s44163-025-00428-0>
2. X. F. Xu, K. Wang, W. H. Ma, C. L. Wu, X. R. Huang, Z. X. Ma, et al., Multi-objective particle swarm optimization algorithm based on multi-strategy improvement for hybrid energy storage optimization configuration, *Renew. Energy*, **223** (2024), 120086. <https://doi.org/10.1016/j.renene.2024.120086>
3. Y. Zhang, B. Li, W. Hong, A. Zhou, MOCPSO: A multi-objective cooperative particle swarm optimization algorithm with dual search strategies, *Neurocomputing*, **562** (2023), 126892. <https://doi.org/10.1016/j.neucom.2023.126892>
4. X. Li, Y. Zhang, W. Hu, Ensemble learning training strategy based on multi-objective particle swarm optimization and chasing method, *Expert Syst. Appl.*, **281** (2025), 127777. <https://doi.org/10.1016/j.eswa.2025.127777>
5. X. Shu, Y. Liu, J. Liu, M. Yang, Q. Zhang, Multi-objective particle swarm optimization with dynamic population size, *J. Comput. Des. Eng.*, **10** (2023), 446–467. <https://doi.org/10.1093/jcde/qwac139>
6. Y. Zhang, Y. Liu, X. Zhang, Q. Song, A. Ouyang, J. Yang, Multi-objective particle swarm optimization with integrated fireworks algorithm and size double archiving, *Int. J. Comput. Intell. Syst.*, **18** (2025) 2. <https://doi.org/10.1007/s44196-024-00722-2>
7. H. Han, L. Zhang, A. Yinga, J. Qiao, Adaptive multiple selection strategy for multi-objective particle swarm optimization, *Inf. Sci.*, **624** (2023) 235–251. <https://doi.org/10.1016/j.ins.2022.12.077>
8. T. Ton-That, M. Hoang-Le, C. Ngo-Huu, T. Cuong-Le, An adaptive multi-swarm optimization algorithm inspired by migration for solving multimodal problems, *Cluster Comput.*, **29** (2026) 129. <https://doi.org/10.1007/s10586-025-05881-1>
9. Y. Cui, X. Meng, J. Qiao, A multi-objective particle swarm optimization algorithm based on two-archive mechanism, *Appl. Soft Comput.*, **119** (2022) 108532. <https://doi.org/10.1016/j.asoc.2022.108532>
10. X. Wu, Y. Pan, Y. Chen, M. Kustudic, J. Luo, *Enhanced multi-objective particle swarm optimization algorithms*, International Conference on Swarm Intelligence, Springer, 2025. https://doi.org/10.1007/978-981-95-0985-0_1
11. Q. Liu, R. Sun, X. Bu, N. Hanajima, W. Ding, Optimal trajectory planning method for handling robots based on multi-objective particle swarm optimization guided by evolutionary information, *Eksploat. Niezawodn.*, 2025. <http://doi.org/10.17531/ein/202990>

12. P. Jangir, H. Buch, S. Mirjalili, P. Manoharan, MOMPA: Multi-objective marine predator algorithm for solving multi-objective optimization problems, *Evol. Intell.*, **16** (2023), 169–195. <https://doi.org/10.1007/s12065-021-00649-z>
13. E. H. Houssein, M. A. Mahdy, M. Kayed, W. M. Mohamed, An efficient Warlus optimizer for solving multi-objective optimization problems: case study with action recognition, *Cluster Comput.*, **28** (2025), 671. <https://doi.org/10.1007/s10586-025-05340-x>
14. N. Khodadadi, E. Khodadadi, B. Abdollahzadeh, E. S.M. El-Kenawy, P. Mardanpour, W. Zhao, et al., Multi-objective generalized normal distribution optimization: A novel algorithm for multi-objective problems, *Cluster Comput.*, **27** (2024), 10589–10631. <https://doi.org/10.1007/s10586-024-04467-7>
15. E. Quan, Y. Liu, H. Gao, H. You, G. Xue, An enhanced NSGA-II algorithm combining lévy flight and simulated annealing and its application in electric winch trajectory planning: A complex multi-objective optimization study, *Machines*, **13** (2025), 1017. <https://doi.org/10.3390/machines13111017>
16. M. Song, J. Lin, X. Liu, H. Jia, S. Luo, Octopus optimization algorithm: a novel single-and multi-objective optimization algorithm for optimization problems, *Cluster Comput.*, **28** (2025), 484. <https://doi.org/10.1007/s10586-025-05141-2>
17. M. S. Rajan, Optimizing photovoltaic panel performance: a comparative study of meta-heuristic algorithms, *Adv. Eng. Intell. Syst.*, **3** (2024), 83–105. <https://doi.org/10.22034/aeis.2024.460309.1201>
18. S. R. Rajeyagari, S. Nowduri, Optimizing solid oxide fuel cell performance using advanced meta-heuristic algorithms, *Adv. Eng. Intell. Syst.*, **3** (2024), 106–126. <https://doi.org/10.22034/aeis.2024.460563.1202>
19. L. Mrkela, Z. Stanimirović, A Multi-objective variable neighborhood search for the maximal covering location problem with customer preferences, *Cluster Comput.*, **25** (2022), 1677–1693. <https://doi.org/10.1007/s10586-021-03524-9>
20. J. Sánchez-Oro, A. D. López-Sánchez, J. M. Colmenar, A multi-objective parallel variable neighborhood search for the bi-objective obnoxious p-median problem, *Optim. Lett.*, **16** (2022), 301–331. <https://doi.org/10.1007/s11590-020-01690-0>
21. Y. Zhang, W. Hu, W. Yao, X. Li, J. Hu, A multi-objective particle swarm optimization based on local ideal points, *Appl. Soft Comput.*, **161** (2024), 111707. <https://doi.org/10.1016/j.asoc.2024.111707>
22. Y. Zhang, B. Li, W. Hong, A. Zhou, MOCPSO: A multi-objective cooperative particle swarm optimization algorithm with dual search strategies, *Neurocomputing*, **562** (2023), 126892. <https://doi.org/10.1016/j.neucom.2023.126892>
23. Y. Tang, K. Huang, Z. Tan, M. Fang, H. Huang, Multi-subswarm cooperative particle swarm optimization algorithm and its application, *Inf. Sci.*, **677** (2024), 120887. <https://doi.org/10.1016/j.ins.2024.120887>
24. Q. Xu, Y. Chen, C. Shi, J. Huang, W. Li, Two-stage particle swarm optimization with dual-indicator fusion ranking for multi-objective problems, *Inf. Sci.*, **679** (2024), 121032. <https://doi.org/10.1016/j.ins.2024.121032>
25. J. Kennedy, R. Eberhart, Particle swarm optimization, *Proc. ICNN'95*, **4** (1995), 1942–1948. <https://doi.org/10.1109/ICNN.1995.488968>

26. K. Tang, C. Meng, Particle swarm optimization algorithm using velocity pausing and adaptive strategy, *Symmetry*, **16** (2024), 661. <https://doi.org/10.3390/sym16060661>
27. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.*, **69** (2014), 46–61. <https://doi.org/10.1016/j.advengsoft.2013.12.007>
28. M. H. Nadimi-Shahraki, H. Zamani, Z. A. Varzaneh, A. S. Sadiq, S. Mirjalili, A systematic review of applying grey wolf optimizer, its variants, and its developments in different Internet of Things applications, *Int. of Things*, **26** (2024), 101135. <https://doi.org/10.1016/j.iot.2024.101135>
29. L. Cheng, J. Cao, W. Wang, L. Cheng, Multiple learning strategies and a modified dynamic multiswarm particle swarm optimization algorithm with a master slave structure, *Appl. Sci.*, **14** (2024), 7035. <https://doi.org/10.3390/app14167035>
30. D. Urošević, R. Todosijević, N. Mladenović, J. Brimberg, Variable neighborhood search, Springer Optimization and Its Applications, 2023. https://doi.org/10.1007/978-3-031-38310-6_8
31. D. Mohan, D. Ghosh, *A neighborhood search algorithm for the tool indexing problem without tool duplication*, Optimization Essentials, Springer, 2024. https://doi.org/10.1007/978-981-99-5491-9_21
32. A. Katkar, V. Metkari, H. T. Jadhav, Performance evaluation of hybrid MOABC–NSGA-II algorithms for constrained multi-objective optimization using benchmark test problems, 2025. Available from: <http://dx.doi.org/10.2139/ssrn.5443555>.
33. C. Audet, J. Bignon, D. Cartier, S. Le Digabel, L. Salomon, Performance indicators in multiobjective optimization, *Eur. J. Oper. Res.*, **292** (2020), 397–422. <https://doi.org/10.1016/j.ejor.2020.11.016>



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0>)