symmetry

Article

A Modification of the Imperialist Competitive Algorithm with Hybrid Methods for Multi-Objective Optimization Problems

Jianfu Luo, Jinsheng Zhou *, Xi Jiang and Haodong Lv

School of Economics and Management, China University of Geosciences (Beijing), Beijing 100083, China; 3007190015@cugb.edu.cn (J.L.); 3007180014@cugb.edu.cn (X.J.); 3007180013@cugb.edu.cn (H.L.)
* Correspondence: zhoujinsheng@cugb.edu.cn

Abstract: This paper proposes a modification of the imperialist competitive algorithm to solve multi-objective optimization problems with hybrid methods (MOHMICA) based on a modification of the imperialist competitive algorithm with hybrid methods (HMICA). The rationale for this is that there is an obvious disadvantage of HMICA in that it can only solve single-objective optimization problems but cannot solve multi-objective optimization problems. In order to adapt to the characteristics of multi-objective optimization problems, this paper improves the establishment of the initial empires and colony allocation mechanism and empire competition in HMICA, and introduces an external archiving strategy. A total of 12 benchmark functions are calculated, including 10 bi-objective and 2 tri-objective benchmarks. Four metrics are used to verify the quality of MOHMICA. Then, a new comprehensive evaluation method is proposed, called "radar map method", which could comprehensively evaluate the convergence and distribution performance of multi-objective optimization algorithm. It can be seen from the four coordinate axes of the radar maps that this is a symmetrical evaluation method. For this evaluation method, the larger the radar map area is, the better the calculation result of the algorithm. Using this new evaluation method, the algorithm proposed in this paper is compared with seven other high-quality algorithms. The radar map area of MOHMICA is at least 14.06% larger than that of other algorithms. Therefore, it is proven that MOHMICA has advantages as a whole.

Keywords: multi-objective optimization problems; hybrid methods; imperialist competitive algorithm; optimization

check for updates

Citation: Luo, J.; Zhou, J.; Jiang, X.; Lv, H. A Modification of the Imperialist Competitive Algorithm with Hybrid Methods for Multi-Objective Optimization Problems. *Symmetry* **2022**, *14*, 173. https://doi.org/10.3390/sym14010173

Academic Editors: Theodore E. Simos, Deming Lei and Ming Li

Received: 12 December 2021
Accepted: 13 January 2022
Published: 16 January 2022

1. Introduction

In the fields of production processes, engineering applications, management and decision-making within complex systems, multi-objective optimization problems are more common than single-objective problems. However, it is very difficult to achieve a solution to meet the requirement that all objective functions are optimal because of the conflict between various objective functions. Therefore, there is hardly a single global optimal solution, but a set of Pareto optimal solutions balanced by the values of various objective functions will be formed. In this case, the process of solving solutions becomes more complex than single-objective optimization, and it is difficult to obtain multiple uniformly distributed approximate Pareto optimal solution sets. Accordingly, it is of theoretical and practical significance to study the solution for such problems.

1.1. Description of Constrained Optimizaiton

Generally, a multi-objective optimization problem can be described by the Formula (1).

$$\begin{array}{ll} & \min \ \{f_1(x), f_2(x), \ldots, f_m(x)\} \\ s.t. & g_i(x) \le 0, i = 1,2\ldots,p \\ & h_j(x) = 0, j = 1,2,\ldots,q \\ & u_k \le x_k \le v_k, x \in R^n, k = 1,2,\ldots,n \end{array} \tag{1}$$

where, $\{f_1(x), f_2(x), \ldots, f_m(x)\}$ represents the individual objective function. $g_i(x) \le 0$ is the i-th inequality constraint in optimization problem in the Formula (1), and p is the number of inequality constraints. $h_j(x) = 0$ is the j-th equation constraint, and q is the number of equation constraints. u_k and v_k are the upper and lower bounds of x_k, respectively. The set $D = \{x \in S | g_i(x) \le 0, h_j(x) = 0, \ i = 1, 2 \ldots, p, j = 1, 2, \ldots, q\}$ that meets all inequality and equality constraints in the search space $S = \{u_k \le x_k \le v_k, x \in R^n, k = 1, 2, \ldots, n\}$ is called the feasible region of the constrained optimization problem in the Formula (1). If a group solution $x \in D$, x is called a feasible solution; otherwise, it is called an infeasible solution. For two group of solutions $x_1 = (x_{11}, x_{12}, \ldots, x_{1n})$ and $x_2 = (x_{21}, x_{21}, \ldots, x_{2n})$, if all components of x_1 are better than x_2, or some components of x_1 are better than x_2 and the others are equal, there is a dominant relationship between x_1 and x_2. Here, x_1 is the dominant solution and x_2 is the dominated solution. Otherwise, there is a non-dominant relationship between x_1 and x_2.

1.2. Related Work

This section can be divided into two parts, including multi-objective swarm and evolutionary algorithms and multi-objective imperialist competitive algorithms.

1.2.1. Multi-Objective Swarm and Evolutionary Algorithms

Swarm and evolutionary algorithms can use the population to search in the optimal direction, so as to make the whole population approach the Pareto front, and finally obtain the approximate Pareto front. There have been several studies about swarm and evolutionary algorithms for solving multi-objective optimization, since Schaffer [1] proposed the vector evaluated genetic algorithm (VEGA). Some well-known algorithms include multiple objective genetic algorithm (MOGA) proposed by Fonseca and Fleming [2], Pareto evolutionary selection algorithm II (PESA-II) proposed by Corne [3], non-dominated sorting in genetic algorithms (NSGA) [4] and non-dominated sorting in genetic algorithms II (NSGA-II) [5] proposed by Deb, multi-objective particle swarm optimization (MOPSO) proposed by Coello [6], multi-objective evolutionary algorithm based on decomposition (MOEA\D) proposed by Q. Zhang [7] and multi-objective artificial bee colony algorithm proposed by Akbari [8].

When solving complex multi-objective optimization problems, the above algorithms may have one or more of the following problems:

(1) With the increase of the number of objective functions, the proportion of non-dominated solutions in the population also increases, which would lead to the slowing down in the speed of search process;
(2) For high-dimensional target space, the computational complexity to maintain diversity is too high, and it is difficult to find the adjacent elements of the solution;
(3) The indexes for evaluating comprehensive performance of the algorithm are poor. Almost all evaluation indexes can only evaluate one of the convergence and distribution of the population in the algorithm; therefore, it is presently difficult to comprehensively evaluate the population convergence and distribution of the swarm and evolutionary algorithms for solving multi-objective optimization;
(4) For the high-dimensional target space, how to visualize the results is also a difficult problem.

In recent years, many new swarm and evolutionary algorithms and their improved algorithms have also been effectively applied in the process of solving multi-objective optimization. Mirjalili proposed the multi-objective grasshopper optimization algorithm (MOGOA) [9], the multi-objective ant lion optimizer (MOALO) [10] and the multi-objective grey wolf optimizer (MOGWO) [11], respectively. The MOGOA algorithm, based on the grasshopper optimization algorithm (GOA), has been proposed when solving multi-objective optimization. In order to solve multi-objective optimization, an archive and target selection mechanism was introduced into GOA. For most multi-objective optimization, MOGOA is a competitive algorithm with high distribution. In addition, the quality of convergence and distribution is competitive. The MOALO algorithm, based on ant lion optimizer (ALO), has also been proposed for solving multi-objective optimization. The algorithm was tested on 17 case studies, including 5 unconstrained functions, 5 constrained benchmarks and 7 engineering design optimizations. Most of the results achieved have been better than NSGA-II and MOPSO. The MOGWO algorithm, based on the grey wolf optimizer (GWO), is another algorithm proposed to solve multi-objective optimization. In this algorithm, in order to save the non-dominated solutions in the iterative process, a fixed-sized external archive was used. Meanwhile, a grid-based approach was employed to maintain and adaptively assess the Pareto front. After solving CEC 2009 [12] benchmarks, the results of MOGWO were compared with that of MOPSO and MOEA/D. Based on MOGWO, using an adaptive chaotic mutation strategy, a multiple search strategy based on the multi-objective grey wolf optimizer (MMOGWO) [13] has been proposed by Liu. An elitism strategy is also introduced into MMOGWO to search for more potential Pareto optimal solutions and store the diversity of solutions in the approximated solution set. Therefore, MMOGWO is verified by some benchmark functions of multi-objective optimization, and competitive calculation results are obtained. Based on stochastic Fractal Search (SFS), Khalilpourazari [14] proposed multi-objective stochastic Fractal Search (MOSFS) with two new components, including archive and leader selection mechanism. Then, this algorithm was applied in the welded beam design problem, obtaining better results than MOPSO and MOGWO. Got [15] extended the whale optimization algorithm (WOA) and proposed a new multi-objective algorithm called the guided population archive whale optimization algorithm (GPAWOA). This algorithm uses an external archive to store the non-dominated solutions searched in the process of solving the optimization problems. The leaders are selected from the archive to guide the population towards promising regions of the search space; also, the mechanism of crowding distance is incorporated into the WOA to maintain the diversity. The algorithm obtained good results, but there is room for improvement in the initialization. In the future, some new swarm and evolutionary algorithms, including aquila optimizer (AO) [16], reptile search algorithm (RSA) [17], and arithmetic optimization algorithm (AOA) [18], can be improved in order to solve multi-objective optimization.

1.2.2. Multi-Objective Imperialist Competitive Algorithms

Imperialist competitive algorithms (ICA) are a kind of evolutionary algorithm based on the colonies' competition mechanism of the imperialist method proposed by Atashpaz-Gargar and Lucas [19], which is a kind of social heuristic optimization algorithm. At present, ICA is widely applied in many different fields, including artificial intelligence [20,21], power electronic engineering [22], supply chain management [23–26], vehicle scheduling [27–29] and production process scheduling [30–32].

In recent years, there has been some research carried out regarding solving multi-objective optimization problems using ICA and all kinds of modified ICA. Enaytifar [33] proposed the multi-objective imperialist competitive algorithm (MOICA). The main calculation steps of MOICA are strictly carried out according to the ICA algorithm. Therefore, there are some problems, including premature convergence, because empires' competition can reduce the number of empires, and computing terminates before the number of iterations reaches the maximum. The reason for this is that convergence is too fast, leading to empires dying out in the process of empire competition. Moreover, there are several steps in the MOICA algorithm, and each step has space to improve, including in terms of the search ability and convergence speed. In order to solve these problems, researchers have proposed some form of modified MOICA. Ghasemi [34] proposed a bare-bones multi-objective imperialist competitive algorithm with a modified version (MGBICA). In that paper, a Gaussian bare-bones operator was introduced in empire assimilation in order to enhance the population diversity. Then, MGBICA is applied in the multi-objective optimal electric power planning, namely optimal power flow (OPF) and optimal reactive power dispatch (ORPD) problems. For this algorithm, the other steps, except for assimilation, have modified room.

Mohammad [35] improved MOICA, a new step that all countries move to the optimal imperialist; they use this algorithm to design variables of brushless DC motor to maximize efficiency and minimize total mass. For this algorithm, such algorithm design can enhance the convergence speed, but increase the possibility of falling into local optimization. At the same time, it cannot solve the problem that the number of empires may be reduced due to imperialist competition, and the iteration may be terminated before the number of iterations reaches the maximum. Piroozfard [36] designed an improved multi-objective imperialist competitive algorithm to solve multi-objective job shop scheduling optimization problem with low carbon emission. The algorithm obtains good calculation results for the model established in this paper, but the application scope has obvious limitations. When Khanali [37] researched multi-objective energy optimization and environmental emissions for a walnut production system, a new modified MOICA was proposed. This algorithm solved the multi-objective optimization for the walnut production system. The result of the most environmental and economic benefits of energy consumption was obtained. In order to solve flexible job shop scheduling problems with transportation, sequence-dependent setup times (FJSSP-TSDST), which is a complex multi-objective problem, Li [38] proposed a new MOICA named imperialist competitive algorithm with feedback (FICA). This algorithm proposed a new assimilation and adaptive revolution mechanism with feedback. Meanwhile, in order to improve the search ability, a novel competition mechanism is presented by solution transferring among empires.

In addition, some improved ICA algorithms that can only solve single objective optimization have the potential to solve multi-objective optimization problems through continuous improvement. A hybrid algorithm using ICA combining Harris Hawks Optimizer (HHO) [39] was proposed, called Imperialist Competitive Harris Hawks Optimization (ICHHO). This algorithm could solve some common optimization problems. Therefore, 23 benchmarks are calculated, and then the results are compared with ICA and HHO. This hybrid algorithm can obtain better results than two basic algorithms. In order to solve assembly flow shop scheduling problem, Li [40] proposed imperialist competitive algorithm with empire cooperation (ECICA). This algorithm uses a new imperialist competitive method through adaptive empire cooperation between the strongest and weakest empires. Tao [41] presented an improved ICA called a discrete imperialist competitive algorithm (DICA) to solve the resource-constrained hybrid flow-shop problem with energy consumption (RCHFSP-EC). A new decoding method considering the resource allocation was designed in this algorithm. Finally, a series of real shop scheduling system instances are calculated and compared with some other high-quality heuristic algorithms. DICA obtained satisfactory results.

1.3. The Main Content of This Paper

From the above literature on the improvement and application of multi-objective imperialist competitive algorithms, these kind of algorithms have the following three problems. First, most algorithms fail to solve the problem that the number of empires is reduced due to imperialist competition. When the number of empires is one, the calculation would not be carried out, which may lead to the early termination of iterative calculation. Second, in the operation process of each step of all kinds of modified imperialist competitive algorithms, most of the algorithms cannot consider both local search and global search. Third, when solving practical problems, some algorithms have limitations, which are only applicable to the problems to be solved, but not universal.

Therefore, in order to solve the above problems of multi-objective optimization using ICA, this paper proposes a new multi-objective imperialist competitive algorithm, called MOHMICA, based on a modification of the imperialist competitive algorithm, HMICA, in the literature [42].

The scientific contribution of this paper can be divided into the following two aspects, including algorithm theory and the evolution of algorithm performance:

(1) From the perspective of algorithm theory, this paper proposes a new scheme to solve multi-objective optimization problems based on HMICA. By calculating 12 multi-objective benchmarks and comparing with some high-quality algorithms in recent years, the algorithm proposed in this paper has certain advantages;
(2) From the perspective of algorithm performance evaluation, this paper proposes a comprehensive evaluation method of multi-objective optimization algorithm by using multiple evaluation metrics.

The second part of this paper will introduce MOHMICA, which is the proposed algorithm in this paper. The third part will introduce the relevant design of numerical simulation in this paper, including performance metrics, comparison algorithms, simulating setting and environment. The fourth part introduces the calculation results and discussion, and the fifth part is the conclusion and future research.

2. The Proposed Algorithm

The steps of MOHMICA include initialization of solutions, the establishment of the initial empires, the development of imperialists and assimilation of colonies, empire interaction, empire revolution, empire competition and external archive.

Among these steps, initialization of solutions, the development of imperialists and assimilation of colonies, empire interaction, and empire revolution are the same as HMICA. In this paper, the steps of the establishment of the initial empires, empire competition and external archive strategy are as follows.

2.1. The Establishment of the Initial Empires

Firstly, generate N initial solutions, namely N countries, using Halton sequences, and then sort these N initial solutions. The rules are as follows:

(1) The feasible solution is better than the infeasible solution. If both solutions are infeasible solutions, compare the value of the violation function. The smaller the value of the violation function is, the better the solution is;
(2) If both solutions are feasible solutions, first judge whether there is a dominant relationship between the two solutions. If one solution dominates the other, the dominant solution is the optimal solution and the dominated solution is the inferior solution;
(3) If the two solutions are mutually non-dominated feasible solutions, arrange the number of dominated solutions of the two solutions in the whole population. The less the number of dominated solutions is, the better the solution is;
(4) If the two solutions are mutually non-dominated feasible solutions, and the number of dominated solutions of the two solutions is the same in the whole population, the crowding distance is compared. The larger the crowding distance is, the better the solution is. The calculation process of the crowding distance can be seen in the literature [5].

After sorting the countries, they are divided into N_{imp} empires. Each empire is composed of an imperialist and several colonies, that is, all countries are composed of N_{imp} imperialists and N_{col} colonies. Here, $N = N_{imp} + N_{col}$. For the top $N_{imp} - 1$ imperialists, the number of colonies randomly assigned to each imperialist is carried out according to the Formula (2), and the remaining colonies are assigned to the last imperialist.

$$NC_i = round\left(\frac{N_{col}}{N_{imp}}\right) + \begin{cases} randi(0,1), & \text{when } i-\text{th imperialist is a non}-\text{dominated feasible solution} \\ 0, & \text{when } i-\text{th imperialist is a dominated feasible solution} \\ -1, & \text{when } i-\text{th imperialist is an infeasible solution} \end{cases} \quad (2)$$

where, NC_i means the number of colonies allocated to the i-th imperialist. $round(\bullet)$ is an integer closest to $\bullet$. $randi(0,1)$ is a random number of 0 or 1.

Allocating colonies such as this can avoid the disadvantage that the calculation formula of empires' power in the basic ICA cannot be used in multi-objective optimization and simplify the steps of colony allocation. Meanwhile, when, $N_{imp}^2 < N_{col}$, it ensures that each imperialist can be assigned to at least one colony.

2.2. Empire Competition

Competition among empires is a process of redistribution of the colonies owned by each empire. The steps are as follows:

Step 1. Compare the quality of each empire and rank them to find out the strongest empire and the weakest empire.

Step 2. If the weakest empire has colonies, find the weakest colony in the weakest empire as the annexed country. If there are no colonies in the weakest empire, the imperialist will be annexed by other empires.

Step 3. Randomly put the annexed countries into other empires.

The rules for ranking the strength of the empire are as follows:

(1) Comparing the number of infeasible solutions in each empire, where the empire with a smaller number is better;
(2) If the number of infeasible solutions of the two empires is the same, compare the number of dominated solutions. The lower the number of dominated solutions, the better empire is;
(3) If the above two are the same, compare the average crowding distance of each empire, where the larger the crowding distance is, the stronger empire is.

2.3. External Archive Strategy

When solving multi-objective optimization, it is necessary to compare the quality of solutions by using the distribution indexes, such as crowding distance, because non-dominated solutions cannot be directly compared. Since a certain number of non-dominated solutions would be generated in each iteration, in order to prevent these non-dominated solutions generated in each iteration from losing in the next iteration, it is necessary to establish an external archive, which could store these non-dominated solutions, merge the non-dominated solutions obtained in each iteration and delete the duplicate or dominated individuals in the external archive. Finally, the elite individuals in the calculation process are retained. The specific process of archiving strategy in this paper is as follows:

Step 1. Arrange the non-dominated solutions obtained in each iteration according to the crowding distance, place into the external archive and delete the duplicate solutions in the external archive;

Step 2. Update the external archive. Recalculate the number of dominated solutions and crowding distance of each solution in the external archive, and define the crowding distance of the D solutions with the minimum value in any specific sub vector as positive infinity. D is the number of objective functions;

Step 3. Delete the dominated solutions of the updated external archive and sort them by crowding distance. If the number of non-dominated solutions is larger than the maximum size of the external archive at this time, the part beyond the maximum size of the external archive will be deleted. In particular, in order to preserve more possible elite solutions, the size of the external archive can be enlarged to a certain extent, for example, twice the population;

Step 4. Find the country that the number of dominated solutions is the largest in all colonies, and replace the colony with the solution with the largest crowding distance in the external archive (excluding two solutions that the crowding distances are positive infinite), and then carry out the next iteration.

2.4. Implementation of the Proposed Algorithm

After the improvement of the hybrid method, the pseudo code of MOHMICA is obtained (Algortithm 1), as shown below.

Algorithm 1: Pseudocode of MOHMICA

Input:
Population total number N
The number of initial imperialists N_{imp} and colonies N_{col}
The number of optimization iterations *MaxIt*, archive size *EA*
Output: MOHMICA Pareto front

```
1   Initialize the MOHMICA population postions by Halton sequence
2   for i = 1: $N$ do
3       Calculate the function values, violation values (if the optimization with constraints) the number of dominated solutions and crowding distance of the initial countries.
4       Sort initial solutions according to the sorting rules in the Section 2.1.
5       Create empires: according to the clonies allocating rules in the Section 2.1.
6   end for
7   while $t \leq MaxIt$ do
8       for i = 1:$N$ do
9           The development of imperialists and the assimilation of colonies: according to literature [42].
10          Calculate the function values, violation values (if the optimization with constraints) the number of dominated solutions and crowding distance of the initial countries.
11          Empire interaction: according to literature [42].
12          Calculate the function values, violation values (if the optimization with constraints) the number of dominated solutions and crowding distance of the initial countries.
13          Empire revolution: according to literature [42].
14          Calculate the function values, violation values (if the optimization with constraints) the number of dominated solutions and crowding distance of the initial countries.
15          Empire interaction: according to literature [42].
16          Empire competition: according to the Section 2.2 of this paper.
17          Update external archive: according to the Section 2.3 of this paper.
18  end for
```

3. Experimental Design

This part will introduce the benchmark functions calculated in this paper, performance metrics, comparison algorithms, simulating setting and environment.

3.1. Benchmark Functions

In order to verify the effectiveness of the algorithm proposed in this paper, 12 benchmark functions are calculated by MOHMICA, including SCH [5], FON [5], ZDT1-ZDT4 in ZDT [5] series, and 6 benchmarks in UF of CEC 2009. Among them, UF8 and UF10 are three objective functions and the other benchmarks are double objective functions. The mathematical expressions of all benchmarks are shown in Table 1.

3.2. Performance Metrics

In order to evaluate the convergence and distribution of solutions, this paper uses four metrics: convergence metric (CM), diversity metric (DM), generational distance (GD) and inverted generational distance (IGD). The introduction of these four indicators is as follows.

Table 1. The mathematical expressions of all benchmarks.

Function Name	Mathematical Expressions	Dimensions	Bounds
SCH	$f_1 = x^2, \quad f_2 = (x-2)^2$	1	$[0, 2]$
FON	$f_1 = 1 - \exp\left[-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right], \quad f_2 = 1 - \exp\left[-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right]$	3	$x_i \in [-4, 4]$
ZDT1	$f_1(x) = x_1, \quad f_2(x) = g(x) \cdot \left[1 - \sqrt{\frac{x_1}{g(x)}}\right], \quad g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$	30	$x_i \in [0, 1]$
ZDT2	$f_1(x) = x_1, \quad f_2(x) = g(x) \cdot \left\{1 - \left[\frac{x_1}{g(x)}\right]^2\right\}, \quad g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$	30	$x_i \in [0, 1]$
ZDT3	$f_1(x) = x_1, \quad f_2(x) = g(x) \cdot \left[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)} \cdot \sin(10 \cdot \pi \cdot x_1)\right], \quad g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$	30	$x_i \in [0, 1]$
ZDT4	$f_1(x) = x_1, \quad f_2(x) = g(x) \cdot \left[1 - \sqrt{\frac{x_1}{g(x)}}\right], \quad g(x) =$ $1 + 10 \cdot (n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10 \cdot \cos(4 \cdot \pi \cdot x_i)\right]$	10	$x_i \in [0, 1]$
UF1	$f_1 = x_1 + \frac{2}{\lvert J_1 \rvert}\sum_{j \in J_1}\left[x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)\right]^2, \quad f_2 = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2 \rvert}\sum_{j \in J_2}\left[x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)\right]^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, \quad J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$	30	$[0, 1] \times [-1, 1]^{n-1}$
UF2	$f_1 = x_1 + \frac{2}{\lvert J_1 \rvert}\sum_{j \in J_1} y_j^2, \quad f_2 = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2 \rvert}\sum_{j \in J_2} y_j^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, \quad J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$ $y_j = \begin{cases} x_j - \left[0.3x_1^2 \cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ x_j - \left[0.3x_1^2 \cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases}$	30	$[0, 1] \times [-1, 1]^{n-1}$
UF3	$f_1 = x_1 + \frac{2}{\lvert J_1 \rvert}\left[4\sum_{j \in J_1} y_j^2 - 2\prod_{j \in J_1}\cos\left(\frac{20 y_j \pi}{\sqrt{j}}\right) + 2\right], \quad f_2 = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2 \rvert}\left[4\sum_{j \in J_2} y_j^2 - 2\prod_{j \in J_2}\cos\left(\frac{20 y_j \pi}{\sqrt{j}}\right) + 2\right]$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, \quad J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$ $y_j = x_j - x_1^{0.5\left[1.0 + \frac{3(j-2)}{n-2}\right]}, \quad j = 2, \ldots, n$	30	$[0, 1]^n$
UF7	$f_1 = x_1 + \frac{2}{\lvert J_1 \rvert}\sum_{j \in J_1} y_j^2, \quad f_2 = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2 \rvert}\sum_{j \in J_2} y_j^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, \quad J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$ $y_j = x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), \quad j = 2, \ldots, n$	30	$[0, 1] \times [-1, 1]^{n-1}$

Table 1. *Cont.*

Function Name	Mathematical Expressions	Dimensions	Bounds
UF8	$f_1(x) = \cos(0.5\pi x_1) \cdot \cos(0.5\pi x_2) + \frac{2}{\lvert J_1 \rvert} \sum_{j \in J_1} \left(x_j - 2x_2 \sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_2(x) = \cos(0.5\pi x_1) \cdot \sin(0.5\pi x_2) + \frac{2}{\lvert J_2 \rvert} \sum_{j \in J_2} \left(x_j - 2x_2 \sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_3(x) = \sin(0.5\pi x_1) + \frac{2}{\lvert J_3 \rvert} \sum_{j \in J_3} \left(x_j - 2x_2 \sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $J_1 = \{j \mid 3 \le j \le n, \text{ and } j-1 \text{ is a multiplication of } 3\}$ $J_2 = \{j \mid 3 \le j \le n, \text{ and } j-2 \text{ is a multiplication of } 3\}$ $J_3 = \{j \mid 3 \le j \le n, \text{ and } j \text{ is a multiplication of } 3\}$	30	$[0,1]^2 \times [-1,1]^{n-2}$
UF10	$f_1(x) = \cos(0.5\pi x_1) \cdot \cos(0.5\pi x_2) + \frac{2}{\lvert J_1 \rvert} \cdot \sum_{j \in J_1} \left[4 \cdot y_j^2 - \cos\left(8 \cdot \pi \cdot y_j\right) + 1\right]$ $f_2(x) = \cos(0.5\pi x_1) \cdot \sin(0.5\pi x_2) + \frac{2}{\lvert J_2 \rvert} \cdot \sum_{j \in J_1} \left[4 \cdot y_j^2 - \cos\left(8 \cdot \pi \cdot y_j\right) + 1\right]$ $f_3(x) = \sin(0.5\pi x_1) + \frac{2}{\lvert J_3 \rvert} \cdot \sum_{j \in J_1} \left[4 \cdot y_j^2 - \cos\left(8 \cdot \pi \cdot y_j\right) + 1\right]$ $J_1 = \{j \mid 3 \le j \le n, \text{ and } j-1 \text{ is a multiplication of } 3\}$ $J_2 = \{j \mid 3 \le j \le n, \text{ and } j-2 \text{ is a multiplication of } 3\}$ $J_3 = \{j \mid 3 \le j \le n, \text{ and } j \text{ is a multiplication of } 3\}$ $y_j = x_j - 2x_2 \sin\left(2\pi x_1 + \frac{j\pi}{n}\right), j = 3, \ldots, n$	30	$[0,1]^2 \times [-2,2]^{n-2}$

(1) Convergence metric

This metric reflects the distance between the approximate Pareto front and the real Pareto front. The smaller the value is, the closer the individual of the solutions is to the real Pareto front, and the better its convergence is. The calculating method is as shown in Equation (3):

$$CM(PF, PF^*) = \frac{1}{n_{nd}} \cdot \sum_{i=1}^{n_{nd}} (\min\|PF_i, PF^*\|) \tag{3}$$

where, *PF* is the calculated approximate Pareto front. PF^* is real Pareto front. n_{nd} is the number of non-dominated solutions. $\|\bullet\|$ means Euclidean distance. Particularly, if $CM = 0$, that means the calculated Pareto front is true Pareto front.

(2) Diversity metric

This metric is used to measure the distribution of non-dominated solutions. The smaller its value is, the better distribution of non-dominated solutions is. The calculation method is shown in Equation (4).

$$DM(PF, PF^*) = \frac{d_f + d_l + \sum\limits_{i=1}^{n_{nd}-1} \left| \|PF_i, PF_{i+1}\| - \frac{\sum\limits_{i=1}^{n_{nd}-1} \|PF_i, PF_{i+1}\|}{n_{nd}-1} \right|}{d_f + d_l + \sum\limits_{i=1}^{n_{nd}-1} \|PF_i, PF_{i+1}\|} \tag{4}$$

where, d_f and d_l are the Euclidean distance between the extreme non-dominated solution and the boundary solutions of the obtained non-dominated solution set.

(3) Generational distance

This metric refers to the distance between the whole approximate Pareto front obtained by the algorithm and the real Pareto front. The smaller the GD is, the closer solutions are to the real Pareto front, and the better the convergence of the algorithm. The calculation method of this metric is shown in Equation (5):

$$GD(PF, PF^*) = \frac{1}{n_{nd}} \cdot \sqrt{\sum_{i=1}^{n_{nd}} (\min\|PF_i, PF^*\|)^2} \tag{5}$$

(4) Inverted generational distance

This metric refers to the distance between the real Pareto front and the approximate Pareto front obtained by the algorithm. To some extent, it is a comprehensive metric that can measure both convergence and diversity of an algorithm. The smaller the IGD, the better quality of algorithm is. The calculation method of IGD is shown in Equation (6):

$$IGD(PF, PF^*) = \frac{1}{n_{PF}} \cdot \sum_{i=1}^{n_{PF}} \min\|PF_i^*, PF\| \tag{6}$$

where, n_{PF} is the number of points of real Pareto front.

3.3. Comparison Algorithm and Simulation Setting

In this paper, each benchmark function is run independently 20 times by using the MOHMICA algorithm, and then compared with some multi-objective algorithms that have achieved good results in solving these kind of problems in recent years, including PESA-II, MOEA\D, NSGA-II, MOABC, MOALO, MOGOA and MMOGWO. In particular, the related parameter settings of PESA-II and MOEA\D are the same as in [3,7]. The related data of the other algorithms are from [13].

The simulation environment is Windows 10, Intel® Core (TM) i7-10875H CPU @ 2.30 GHz with a 16.00 GB RAM memory with a running environment of MATLAB 2017b.

The initial population size of the MOHMICA algorithm is set to 100, and the size of the external archive is set to 200. For SCH and FON, the maximum number of iterations is 50, meaning the maximum number of evaluations is 5000. The maximum number of iterations of other two objective functions is 250, that is, the maximum number of evaluations is 25,000. For the three objective benchmark functions, the maximum number of iterations is 500, that is, the maximum number of evaluations is 50,000. In order to ensure that the comparison results of different algorithms are fair when calculating the same function, the population number, maximum iteration times and maximum evaluation times of all comparison algorithms are the same as those of MOHMICA.

4. Results and Discussion

4.1. Calculation Results and Discussion of Benchmark Functions

The results of MOHMICA and other comparison algorithms are shown in Tables 2–5. These four tables show the mean value and standard deviation (SD) of CM, DM, GD and IGD respectively. Meanwhile, the ranking of mean values of each algorithm are counted in these tables. Then, the results of average of ranking values of four metrics on each algorithm are in Table 6. From this table, MOHMICA ranked first more than all the other algorithms.

In Tables 2–6, some rules about relevant metrics can be obtained when calculating each benchmark function of MOHMICA. For the convergence metrics including CM and GD, MOHMICA has an obvious advantage in general. For the distribution metric DM, the amount of times that MOHMICA ranked first was the most among all algorithms. For the metric of IGD, the comprehensive ranking of the algorithm proposed in this paper was slightly lower than MOALO and MMOGWO, but significantly higher than other algorithms. The reason for this is the low ranking of SCH and UF2 functions. On the whole, the more complex a benchmark function is, the better the result obtained by MOHMICA is. The results of all benchmark functions calculated by different algorithms from Tables 2–5 can be quantitatively verified by the Wilcoxon test on the four metrics of each algorithm. This test is conducted with three levels of significance, namely, $\alpha = 0.01$, $\alpha = 0.05$ and $\alpha = 0.1$. The statistical hypotheses for the Wilcoxon test are as follows:

(1) H_0: The results of the two algorithms are homogenous;
(2) H_1: The results of the two algorithms are heterogenous.

According to the results of the Wilcoxon test in Table 7, the conclusions that can be obtained as follows:

(1) From the perspective of R+, MOHMICA has advantages over the other algorithms. Moreover, most of the results can pass the level of significance of $\alpha = 0.01$;
(2) For the convergence metric CM, only two comparing algorithms, including MOGOA and MMOGWO, cannot pass the level of significance of $\alpha = 0.01$, but can pass the level of significance of $\alpha = 0.05$. For the other convergence metric GD, the performance of MOHMICA is worse than that of CM, with three algorithms including MOALO, MOGOA and MMOGWO falling the level of significance of $\alpha = 0.01$. Moreover, the latter two cannot pass the level of significance of $\alpha = 0.1$, although MOHMICA has advantages over them;
(3) For the distribution metric DM, except PESA-II failing to achieve the level of significance of $\alpha = 0.1$, MOHMICA outperforms other algorithms with a level of significance of $\alpha = 0.01$;
(4) For the comprehensive metric IGD, MOHMICA has some advantages over MOALO and MOGOA, but these are not significant. The results of MOHMICA and MMOGWO are equal. It has obvious advantages over other algorithms with a level of significance of $\alpha = 0.05$.

Table 2. The results of convergence metric (CM) for all benchmark functions.

Benchmark Functions		MOHMICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
	Mean	**1.328×10^{-3}**	1.373×10^{-3}	2.870×10^{-3}	8.03×10^{-3}	8.38×10^{-3}	7.40×10^{-3}	8.28×10^{-3}	8.18×10^{-3}
SCH	**SD**	**1.332×10^{-4}**	2.441×10^{-4}	3.546×10^{-3}	5.41×10^{-4}	4.72×10^{-4}	1.27×10^{-3}	6.96×10^{-4}	6.85×10^{-4}
	Rank	**1**	2	3	5	8	4	7	6
	Mean	2.706×10^{-3}	**2.249×10^{-3}**	3.182×10^{-3}	9.97×10^{-3}	3.65×10^{-2}	1.11×10^{-2}	9.23×10^{-2}	1.06×10^{-2}
FON	**SD**	**2.154×10^{-4}**	2.380×10^{-4}	2.199×10^{-3}	3.87×10^{-4}	8.89×10^{-3}	2.08×10^{-3}	1.12×10^{-2}	8.29×10^{-4}
	Rank	2	**1**	3	4	7	6	8	5
	Mean	2.639×10^{-3}	7.745×10^{-2}	6.369×10^{-2}	4.61×10^{-2}	2.94×10^{-1}	5.04×10^{-3}	7.79×10^{-2}	**1.23×10^{-3}**
ZDT1	**SD**	1.075×10^{-3}	1.731×10^{-2}	7.270×10^{-2}	4.33×10^{-2}	5.59×10^{-2}	9.67×10^{-3}	2.33×10^{-1}	**4.01×10^{-4}**
	Rank	2	6	5	4	8	3	7	**1**
	Mean	2.341×10^{-3}	1.253×10^{-1}	8.943×10^{-1}	7.52×10^{-2}	3.05×10^{-1}	**5.40×10^{-4}**	4.02×10^{-3}	8.52×10^{-4}
ZDT2	**SD**	6.979×10^{-4}	2.924×10^{-2}	4.823×10^{-1}	4.28×10^{-2}	7.19×10^{-2}	**7.52×10^{-5}**	6.95×10^{-3}	1.06×10^{-4}
	Rank	3	6	8	5	7	**1**	4	2
	Mean	3.965×10^{-3}	7.376×10^{-2}	8.962×10^{-1}	5.31×10^{-2}	1.87×10^{-1}	7.67×10^{-3}	3.83×10^{-2}	**4.69×10^{-4}**
ZDT3	**SD**	**3.885×10^{-4}**	1.550×10^{-2}	7.403×10^{-1}	5.42×10^{-2}	5.94×10^{-2}	3.27×10^{-3}	6.39×10^{-2}	6.19×10^{-4}
	Rank	2	6	8	5	7	3	4	**1**
	Mean	**2.003×10^{-3}**	2.515	1.011	7.08	2.25×10^{-5}	20.1	15.3	4.25
ZDT4	**SD**	**2.899×10^{-4}**	1.613	5.481×10^{-1}	2.85	8.90×10^{-1}	5.24	3.37×10^{-1}	4.15
	Rank	**1**	4	2	6	3	8	7	5
	Mean	**3.810×10^{-2}**	3.814	3.982	2.22×10^{-1}	7.95×10^{-2}	6.76×10^{-2}	9.04×10^{-2}	4.43×10^{-2}
UF1	**SD**	**8.746×10^{-3}**	1.990×10^{-1}	3.816×10^{-1}	9.24×10^{-2}	2.10×10^{-2}	5.15×10^{-2}	3.65×10^{-2}	3.80×10^{-2}
	Rank	**1**	7	8	6	4	3	5	2
	Mean	4.716×10^{-2}	7.390×10^{-2}	6.105×10^{-2}	7.92×10^{-2}	4.12×10^{-2}	1.23×10^{-1}	**2.21×10^{-2}**	5.13×10^{-2}
UF2	**SD**	9.735×10^{-3}	1.487×10^{-2}	2.064×10^{-2}	2.51×10^{-2}	**7.31×10^{-3}**	4.25×10^{-2}	2.47×10^{-2}	1.44×10^{-2}
	Rank	3	6	5	7	2	8	**1**	4
	Mean	**1.112×10^{-1}**	1.879	4.122	3.11×10^{-1}	3.39×10^{-1}	2.15×10^{-1}	1.72×10^{-1}	2.54×10^{-1}
UF3	**SD**	1.469×10^{-1}	1.215	9.176×10^{-1}	8.20×10^{-2}	6.92×10^{-2}	8.63×10^{-2}	**4.74×10^{-2}**	6.05×10^{-2}
	Rank	**1**	7	8	5	6	3	2	4
	Mean	3.172×10^{-2}	3.913×10^{-2}	4.450×10^{-2}	2.54×10^{-1}	7.08×10^{-2}	5.46×10^{-2}	3.33×10^{-2}	**2.15×10^{-2}**
UF7	**SD**	1.074×10^{-2}	$1.636 \times 10{-}^{2}$	2.496×10^{-2}	1.55×10^{-1}	2.30×10^{-2}	4.69×10^{-2}	1.91×10^{-2}	**5.04×10^{-3}**
	Rank	3	5	6	2	8	7	4	**1**
	Mean	6.497×10^{-2}	9.886×10^{-1}	6.195×10^{-1}	4.65	**2.59×10^{-2}**	1.91×10^{-1}	4.53×10^{-1}	1.96
UF8	**SD**	4.238×10^{-2}	7.686×10^{-1}	4.503×10^{-1}	9.59×10^{-1}	**1.83×10^{-2}**	1.42×10^{-1}	5.97×10^{-1}	7.10×10^{-1}
	Rank	2	6	5	8	**1**	3	4	7
	Mean	**2.562×10^{-1}**	37.75	25.44	12.7	6.68×10^{-1}	3.46	2.48	4.79
UF10	**SD**	**5.884×10^{-2}**	12.31	13.11	2.62	3.40×10^{-1}	7.75×10^{-1}	3.40×10^{-1}	1.86
	Rank	**1**	8	7	3	2	5	4	6

Table 3. The results of diversity metric (DM) for all benchmark functions.

Benchmark Functions		MOHMICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
	Mean	8.881×10^{-1}	7.445×10^{-1}	1.004	**4.14×10^{-1}**	9.35×10^{-1}	1.53	1.05	9.68×10^{-1}
SCH	**SD**	5.480×10^{-2}	8.738×10^{-1}	8.104×10^{-1}	**4.43×10^{-2}**	7.03×10^{-2}	1.11×10^{-1}	6.94×10^{-2}	1.35×10^{-1}
	Rank	4	2	6	**1**	3	8	7	5
	Mean	**2.300×10^{-1}**	9.964×10^{-1}	8.855×10^{-1}	3.92×10^{-1}	7.19×10^{-1}	1.36	1.44	8.97×10^{-1}
FON	**SD**	1.603×10^{-1}	1.381×10^{-1}	3.006×10^{-1}	**4.40×10^{-2}**	1.14×10^{-1}	1.16×10^{-1}	1.37×10^{-1}	8.04×10^{-2}
	Rank	**1**	6	5	2	4	7	8	3
	Mean	7.849×10^{-1}	4.924×10^{-1}	6.744×10^{-1}	**4.56×10^{-1}**	8.08×10^{-1}	1.11	1.20	1.09
ZDT1	**SD**	1.005×10^{-1}	3.133×10^{-1}	4.230×10^{-1}	5.10×10^{-2}	8.04×10^{-2}	**4.71×10^{-2}**	7.41×10^{-2}	1.24×10^{-1}
	Rank	4	2	3	**1**	5	7	8	6
	Mean	**2.339×10^{-1}**	5.747×10^{-1}	1.101	5.01×10^{-1}	8.50×10^{-1}	1.02	1.00	9.89×10^{-1}
ZDT2	**SD**	1.175×10^{-1}	3.249×10^{-1}	5.375×10^{-1}	6.90×10^{-2}	9.17×10^{-2}	7.40×10^{-3}	**2.30×10^{-4}**	1.44×10^{-1}
	Rank	**1**	3	8	2	4	7	6	5
	Mean	6.281×10^{-1}	**5.084×10^{-1}**	8.692×10^{-1}	5.28×10^{-1}	8.16×10^{-1}	1.30	1.28	9.78×10^{-1}
ZDT3	**SD**	1.150×10^{-1}	2.534×10^{-1}	7.403×10^{-1}	1.02×10^{-1}	**9.78×10^{-2}**	1.09×10^{-1}	1.19×10^{-1}	1.06×10^{-1}
	Rank	3	**1**	5	2	4	8	7	6
	Mean	7.841×10^{-1}	**5.215×10^{-1}**	1.011	9.36×10^{-1}	1.01	1.04	9.81×10^{-1}	1.04
ZDT4	**SD**	8.153×10^{-2}	4.647×10^{-1}	5.481×10^{-1}	3.25×10^{-2}	1.1×10^{-1}	4.24×10^{-2}	**0.00**	6.61×10^{-2}
	Rank	2	**1**	5	3	5	7	4	7
	Mean	**5.859×10^{-1}**	9.075×10^{-1}	6.372×10^{-1}	8.11×10^{-1}	7.16×10^{-1}	1.14	1.07	9.48×10^{-1}
UF1	**SD**	2.984×10^{-1}	5.877×10^{-1}	5.285×10^{-1}	7.58×10^{-2}	1.05×10^{-1}	1.31×10^{-1}	**5.99×10^{-2}**	9.99×10^{-2}
	Rank	**1**	5	4	3	2	8	7	6
	Mean	2.886×10^{-1}	8.539×10^{-1}	1.044	5.92×10^{-1}	6.49×10^{-1}	1.34	1.05	**1.00×10^{-1}**
UF2	**SD**	2.145×10^{-1}	5.205×10^{-1}	7.942×10^{-1}	6.26×10^{-2}	9.13×10^{-2}	1.24×10^{-1}	**2.98×10^{-2}**	9.03×10^{-2}
	Rank	2	5	6	3	4	8	7	**1**
	Mean	**8.723×10^{-2}**	5.565×10^{-1}	3.369×10^{-1}	8.61×10^{-1}	8.76×10^{-1}	1.50	1.08	1.19
UF3	**SD**	2.201×10^{-1}	5.321×10^{-1}	2.178×10^{-1}	**8.62×10^{-2}**	1.12×10^{-1}	1.77×10^{-1}	2.76×10^{-2}	2.52×10^{-1}
	Rank	**1**	3	2	4	5	8	6	7
	Mean	**3.881×10^{-1}**	4.977×10^{-1}	1.252	8.87×10^{-1}	8.85×10^{-1}	1.38	1.18	1.11
UF7	**SD**	2.363×10^{-1}	3.726×10^{-1}	8.852×10^{-1}	**7.93×10^{-2}**	1.01×10^{-1}	1.81×10^{-1}	8.02×10^{-2}	1.68×10^{-1}
	Rank	**1**	2	7	4	3	8	6	5
	Mean	6.238×10^{-1}	**6.185×10^{-1}**	1.276	7.36×10^{-1}	1.01	1.15	1.07	8.40×10^{-1}
UF8	**SD**	2.687×10^{-1}	3.627×10^{-1}	6.372×10^{-1}	3.96×10^{-2}	1.18×10^{-1}	7.79×10^{-2}	6.88×10^{-2}	**4.15×10^{-2}**
	Rank	2	**1**	8	3	5	7	6	4
	Mean	5.439×10^{-1}	**5.024×10^{-1}**	5.950×10^{-1}	7.39×10^{-1}	8.60×10^{-1}	1.06	1.09	8.99×10^{-1}
UF10	**SD**	2.459×10^{-1}	3.404×10^{-1}	6.405×10^{-1}	4.49×10^{-2}	1.20×10^{-1}	6.67×10^{-2}	4.23×10^{-2}	**3.74×10^{-2}**
	Rank	2	**1**	3	4	5	7	8	6

Table 4. The results of generational distance (GD) for all benchmark functions.

Benchmark functions		MOHMICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
	Mean	$\mathbf{1.837 \times 10^{-4}}$	2.245×10^{-4}	2.752×10^{-4}	9.41×10^{-4}	9.94×10^{-4}	8.78×10^{-4}	9.63×10^{-4}	9.44×10^{-4}
SCH	SD	$\mathbf{1.686 \times 10^{-5}}$	2.865×10^{-5}	1.847×10^{-5}	5.13×10^{-5}	4.73×10^{-5}	1.16×10^{-4}	6.85×10^{-5}	6.86×10^{-5}
	Rank	**1**	2	3	5	8	4	7	6
	Mean	3.369×10^{-4}	$\mathbf{2.648 \times 10^{-4}}$	4.661×10^{-4}	1.18×10^{-3}	1.06×10^{-2}	1.28×10^{-3}	1.01×10^{-2}	1.23×10^{-3}
FON	SD	3.695×10^{-5}	$\mathbf{2.591 \times 10^{-5}}$	4.613×10^{-4}	3.75×10^{-5}	1.96×10^{-3}	2.10×10^{-4}	1.03×10^{-3}	7.99×10^{-5}
	Rank	2	**1**	3	4	8	6	7	5
	Mean	3.192×10^{-4}	8.538×10^{-3}	6.435×10^{-3}	4.78×10^{-3}	9.73×10^{-2}	6.70×10^{-4}	8.55×10^{-3}	$\mathbf{2.34 \times 10^{-4}}$
ZDT1	SD	$\mathbf{1.083 \times 10^{-4}}$	2.564×10^{-3}	7.331×10^{-3}	4.47×10^{-3}	2.37×10^{-2}	1.32×10^{-3}	2.65×10^{-2}	1.37×10^{-4}
	Rank	2	6	5	4	8	3	7	**1**
	Mean	3.134×10^{-4}	1.303×10^{-2}	8.962×10^{-2}	7.58×10^{-3}	1.21×10^{-1}	$\mathbf{6.12 \times 10^{-5}}$	2.25×10^{-3}	9.79×10^{-5}
ZDT2	SD	8.934×10^{-5}	2.948×10^{-3}	4.810×10^{-2}	4.26×10^{-3}	3.52×10^{-2}	$\mathbf{6.83 \times 10^{-6}}$	5.75×10^{-3}	1.09×10^{-5}
	Rank	3	6	7	5	8	**1**	4	2
	Mean	$\mathbf{5.102 \times 10^{-4}}$	1.022×10^{-2}	2.626×10^{-2}	6.98×10^{-3}	6.56×10^{-2}	1.22×10^{-3}	4.70×10^{-3}	6.16×10^{-4}
ZDT3	SD	$\mathbf{5.726 \times 10^{-5}}$	3.748×10^{-3}	7.629×10^{-3}	5.77×10^{-3}	2.21×10^{-2}	6.85×10^{-4}	6.78×10^{-3}	9.65×10^{-5}
	Rank	**1**	6	7	5	8	3	4	2
	Mean	$\mathbf{2.498 \times 10^{-4}}$	2.603×10^{-1}	2.601×10^{-1}	7.13×10^{-1}	11.9	2.06	14.8	6.10×10^{-1}
ZDT4	SD	$\mathbf{3.085 \times 10^{-5}}$	1.645×10^{-1}	1.840×10^{-1}	2.84×10^{-1}	5.59×10^{-1}	6.65×10^{-1}	2.11	6.53×10^{-1}
	Rank	**1**	3	2	5	8	6	7	4
	Mean	6.493×10^{-3}	4.351×10^{-1}	4.918×10^{-1}	3.21×10^{-2}	2.72×10^{-2}	8.32×10^{-3}	1.14×10^{-2}	$\mathbf{5.42 \times 10^{-3}}$
UF1	SD	$\mathbf{9.354 \times 10^{-4}}$	8.314×10^{-2}	1.727×10^{-1}	1.46×10^{-2}	9.75×10^{-3}	5.29×10^{-3}	6.19×10^{-3}	4.17×10^{-3}
	Rank	2	7	8	6	5	3	4	**1**
	Mean	7.537×10^{-3}	9.090×10^{-3}	8.073×10^{-3}	1.42×10^{-2}	9.60×10^{-3}	1.43×10^{-2}	$\mathbf{2.66 \times 10^{-3}}$	7.37×10^{-3}
UF2	SD	$\mathbf{1.917 \times 10^{-3}}$	2.782×10^{-3}	2.195×10^{-3}	5.33×10^{-3}	2.80×10^{-3}	4.77×10^{-3}	2.76×10^{-3}	2.35×10^{-3}
	Rank	3	5	4	7	6	8	**1**	2
	Mean	2.766×10^{-2}	2.635×10^{-1}	5.185×10^{-1}	3.20×10^{-2}	1.39×10^{-1}	2.94×10^{-2}	$\mathbf{1.87 \times 10^{-2}}$	3.24×10^{-2}
UF3	SD	6.751×10^{-2}	1.350×10^{-1}	1.510×10^{-1}	8.99×10^{-3}	3.07×10^{-1}	6.64×10^{-3}	4.53×10^{-3}	$\mathbf{3.46 \times 10^{-3}}$
	Rank	2	7	8	4	6	3	**1**	5
	Mean	3.498×10^{-3}	4.398×10^{-3}	5.600×10^{-2}	2.87×10^{-2}	2.75×10^{-2}	5.84×10^{-3}	5.37×10^{-3}	$\mathbf{2.53 \times 10^{-3}}$
UF7	SD	1.584×10^{-3}	1.566×10^{-3}	2.156×10^{-1}	1.76×10^{-2}	1.18×10^{-2}	5.01×10^{-3}	1.85×10^{-3}	$\mathbf{6.57 \times 10^{-4}}$
	Rank	2	3	8	7	6	5	4	**1**
	Mean	$\mathbf{1.503 \times 10^{-2}}$	9.907×10^{-2}	7.081×10^{-2}	5.26×10^{-1}	1.85×10^{-2}	1.99×10^{-2}	5.70×10^{-2}	2.03×10^{-1}
UF8	SD	1.600×10^{-2}	7.682×10^{-2}	6.364×10^{-2}	1.14×10^{-1}	1.52×10^{-2}	$\mathbf{1.45 \times 10^{-2}}$	7.03×10^{-2}	7.12×10^{-2}
	Rank	**1**	6	5	8	2	3	4	7
	Mean	$\mathbf{2.892 \times 10^{-2}}$	3.883	2.070	1.34	3.85×10^{-1}	3.49×10^{-1}	2.88×10^{-1}	4.93×10^{-1}
UF10	SD	$\mathbf{7.995 \times 10^{-3}}$	1.296	1.485	2.95×10^{-1}	2.25×10^{-1}	7.70×10^{-2}	4.31×10^{-2}	1.89×10^{-1}
	Rank	**1**	8	7	6	4	3	2	5

Table 5. The results of inverted generational distance (IGD) for all benchmark functions.

Benchmark Functions		MOHMICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
	Mean	3.071×10^{-2}	8.118×10^{-2}	4.315×10^{-2}	2.00×10^{-3}	2.07×10^{-3}	$\mathbf{1.83 \times 10^{-3}}$	2.10×10^{-3}	2.02×10^{-3}
SCH	SD	1.025×10^{-2}	7.402×10^{-2}	1.123×10^{-2}	1.35×10^{-4}	$\mathbf{1.18 \times 10^{-4}}$	3.18×10^{-4}	1.74×10^{-4}	1.71×10^{-4}
	Rank	6	8	7	2	4	**1**	5	3
	Mean	$\mathbf{7.468 \times 10^{-3}}$	2.263×10^{-1}	1.063×10^{-1}	1.01×10^{-2}	3.75×10^{-2}	1.11×10^{-2}	9.70×10^{-2}	1.08×10^{-2}
FON	SD	8.528×10^{-4}	2.592×10^{-3}	6.293×10^{-2}	3.96×10^{-4}	9.09×10^{-3}	2.13×10^{-3}	1.14×10^{-2}	8.48×10^{-4}
	Rank	**1**	8	7	2	5	4	6	3
	Mean	7.408×10^{-3}	8.114×10^{-2}	6.826×10^{-2}	3.72×10^{-2}	3.02×10^{-1}	1.57×10^{-3}	2.58×10^{-2}	$\mathbf{1.18 \times 10^{-3}}$
ZDT1	SD	1.136×10^{-3}	2.040×10^{-2}	7.086×10^{-2}	4.33×10^{-2}	5.59×10^{-2}	9.67×10^{-3}	2.33×10^{-1}	$\mathbf{4.01 \times 10^{-4}}$
	Rank	3	7	6	5	8	2	4	**1**
	Mean	6.696×10^{-3}	1.776×10^{-1}	9.824×10^{-1}	8.31×10^{-2}	3.02×10^{-1}	$\mathbf{5.40 \times 10^{-4}}$	8.79×10^{-4}	8.52×10^{-4}
ZDT2	SD	6.855×10^{-4}	1.910×10^{-1}	5.170×10^{-1}	4.28×10^{-2}	5.17×10^{-3}	$\mathbf{7.52 \times 10^{-5}}$	6.95×10^{-3}	1.06×10^{-4}
	Rank	4	6	8	5	7	**1**	3	2
	Mean	5.766×10^{-3}	2.589×10^{-2}	8.298×10^{-2}	2.91×10^{-2}	1.21×10^{-1}	4.40×10^{-3}	1.46×10^{-2}	$\mathbf{2.74 \times 10^{-3}}$
ZDT3	SD	8.433×10^{-4}	6.404×10^{-3}	1.750×10^{-2}	3.86×10^{-2}	3.43×10^{-2}	2.80×10^{-3}	4.68×10^{-2}	$\mathbf{3.71 \times 10^{-4}}$
	Rank	3	5	7	6	8	2	4	**1**
	Mean	$\mathbf{6.061 \times 10^{-3}}$	2.367	2.558	6.22	2.17	18.5	15.3	4.04
ZDT4	SD	$\mathbf{6.388 \times 10^{-4}}$	1.484	1.586	2.85	8.92×10^{-1}	5.25	3.38×10^{-1}	4.16
	Rank	**1**	3	4	6	2	8	7	5
	Mean	1.055×10^{-1}	3.852	4.076	2.11×10^{-1}	7.65×10^{-2}	5.12×10^{-2}	7.82×10^{-2}	$\mathbf{2.42 \times 10^{-2}}$
UF1	SD	$\mathbf{3.389 \times 10^{-3}}$	1.779×10^{-1}	4.040×10^{-1}	9.24×10^{-2}	2.10×10^{-2}	5.15×10^{-2}	3.65×10^{-2}	3.80×10^{-2}
	Rank	5	7	8	6	3	2	4	**1**
	Mean	1.000×10^{-1}	9.146×10^{-2}	1.419×10^{-1}	7.14×10^{-2}	3.92×10^{-2}	1.07×10^{-1}	$\mathbf{1.28 \times 10^{-2}}$	4.70×10^{-2}
UF2	SD	$\mathbf{8.393 \times 10^{-3}}$	2.784×10^{-2}	1.840×10^{-2}	2.51×10^{-2}	7.31×10^{-3}	4.25×10^{-2}	2.47×10^{-2}	1.44×10^{-2}
	Rank	6	5	8	4	2	7	**1**	3
	Mean	2.508×10^{-1}	1.739	4.167	3.02×10^{-1}	3.31×10^{-1}	1.91×10^{-1}	$\mathbf{1.56 \times 10^{-1}}$	2.53×10^{-1}
UF3	SD	6.304×10^{-2}	8.475×10^{-1}	8.612×10^{-1}	8.20×10^{-2}	6.92×10^{-2}	8.63×10^{-2}	$\mathbf{4.74 \times 10^{-2}}$	6.05×10^{-2}
	Rank	3	7	8	5	6	2	**1**	4
	Mean	5.428×10^{-2}	9.423×10^{-2}	1.486×10^{-1}	2.54×10^{-1}	7.20×10^{-2}	3.03×10^{-2}	2.66×10^{-2}	$\mathbf{2.11 \times 10^{-2}}$
UF7	SD	1.141×10^{-2}	1.164×10^{-2}	5.324×10^{-2}	1.55×10^{-1}	2.30×10^{-2}	4.69×10^{-2}	1.91×10^{-2}	$\mathbf{5.04 \times 10^{-3}}$
	Rank	4	6	7	8	5	3	2	**1**
	Mean	1.649×10^{-1}	1.343	9.484×10^{-1}	4.64	1.89×10^{-2}	$\mathbf{1.50 \times 10^{-1}}$	2.11×10^{-1}	1.95
UF8	SD	$\mathbf{4.118 \times 10^{-2}}$	6.953×10^{-1}	4.026×10^{-1}	9.59×10^{-1}	1.83×10^{-2}	1.42×10^{-1}	5.97×10^{-1}	7.10×10^{-1}
	Rank	3	6	5	8	**1**	2	4	7
	Mean	$\mathbf{2.562 \times 10^{-1}}$	37.96	25.61	11.7	5.63×10^{-1}	3.28	2.71	4.93
UF10	SD	$\mathbf{5.884 \times 10^{-2}}$	12.28	13.10	2.62	3.40×10^{-1}	7.75×10^{-1}	4.30×10^{-1}	1.86
	Rank	**1**	8	7	6	3	2	4	5

Table 6. Average ranking of each algorithm on each metric.

Metrics	MOHMICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
CM	**1.467**	4.267	4.533	4	4.2	3.6	3.8	2.933
DM	**1.6**	2.133	4.133	2.133	3.267	6	5.333	4.067
GD	**1.4**	4	4.667	4.4	5.133	3.2	3.467	2.733
IGD	2.667	5.133	5.467	4.2	3.533	2.533	2.933	**2.333**

Table 7. Results of the Wilcoxon test results of each metric by MOHMICA and other multi-objective algorithms.

MOHMICA vs.	CM					DM				
	R+	R−	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	R+	R−	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$
PESA-II	76	2	H1	H1	H1	53	25	H0	H0	H0
MOEA\D	78	0	H1	H1	H1	75	3	H1	H1	H1
NSGA-II	78	0	H1	H1	H1	58	20	H1	H1	H1
MOABC	74	4	H1	H1	H1	78	0	H1	H1	H1
MOALO	77	1	H1	H1	H1	78	0	H1	H1	H1
MOGOA	74	4	H1	H1	H1	78	0	H1	H1	H1
MMOGWO	65	13	H0	H1	H1	76	2	H1	H1	H1
MOHMICA vs.	**GD**					**IGD**				
	R+	R−	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	R+	R−	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$
PESA-II	76	2	H1	H1	H1	77	1	H1	H1	H1
MOEA\D	78	0	H1	H1	H1	78	0	H1	H1	H1
NSGA-II	78	0	H1	H1	H1	74	4	H1	H1	H1
MOABC	78	0	H1	H1	H1	65	13	H0	H1	H1
MOALO	77	1	H0	H1	H1	43	35	H0	H0	H0
MOGOA	65	13	H0	H1	H1	44	34	H0	H0	H0
MMOGWO	60	18	H0	H0	H0	39	39	H0	H0	H0

In order to show the advantages of MOHMICA in calculating these benchmark functions intuitively, the approximate Pareto front of each benchmark function calculated by MOHMICA, PESA-II, MOEA\D are compared with the real Pareto front, as shown from Figures 1–12. It is easy to see that the MOHMICA algorithm has obvious advantages in benchmark functions from these function images.

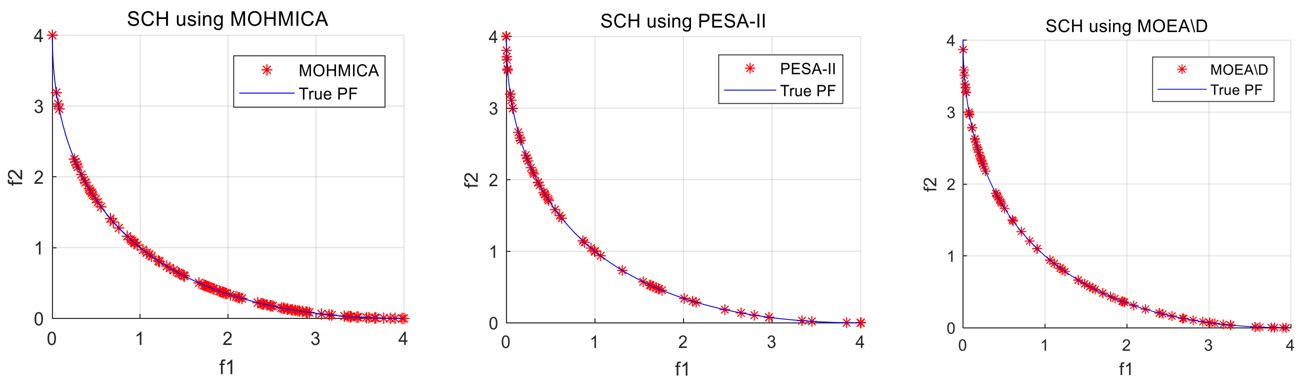

Figure 1. Pareto frontiers of SCH benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

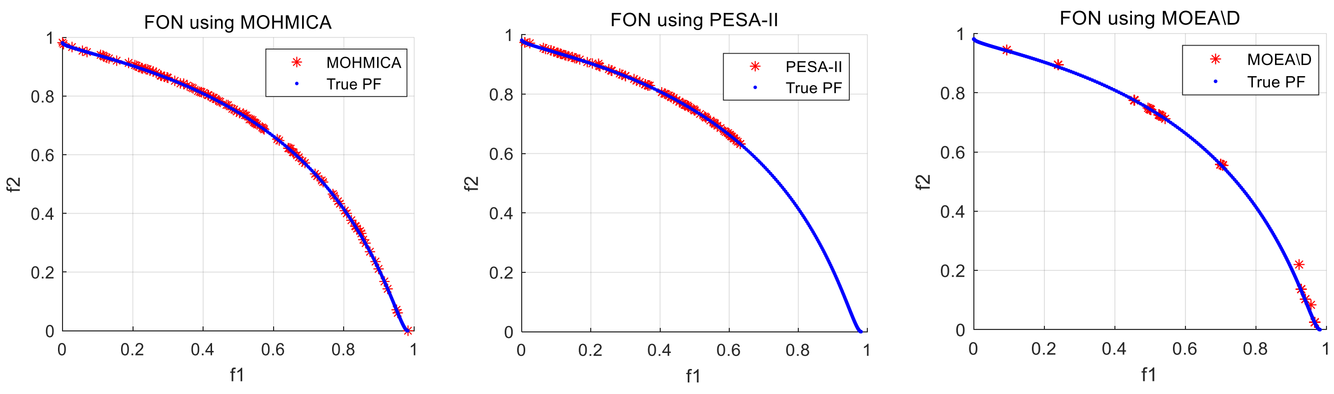

Figure 2. Pareto frontiers of FON benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

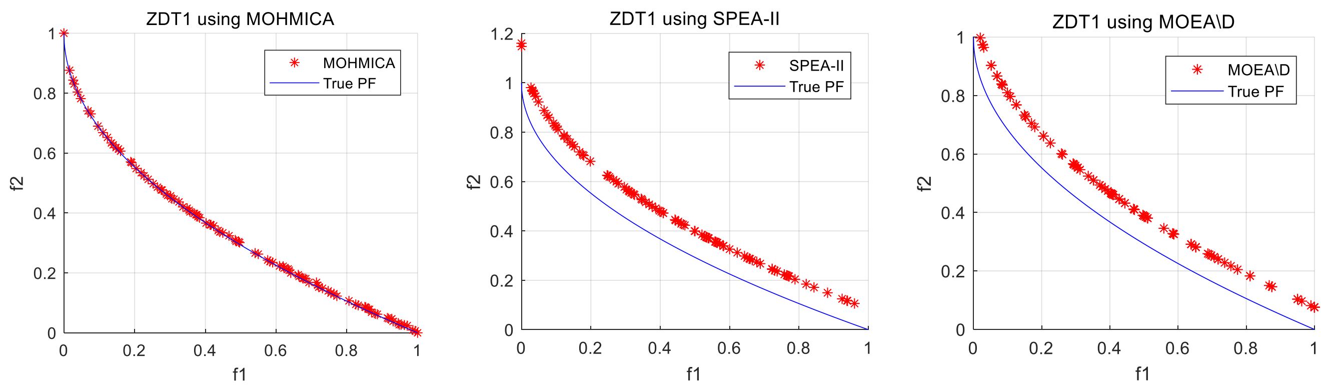

Figure 3. Pareto frontiers of ZDT1 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

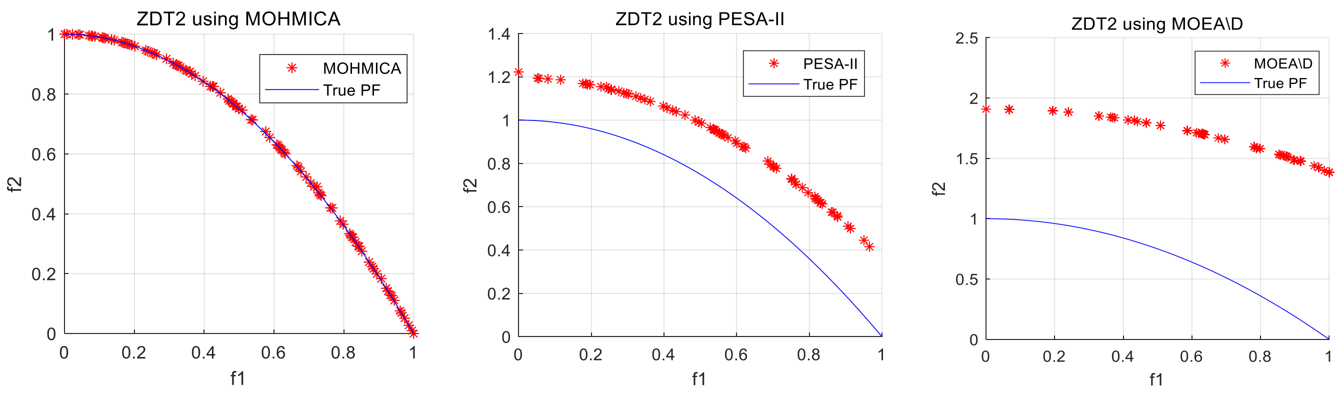

Figure 4. Pareto frontiers of ZDT2 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

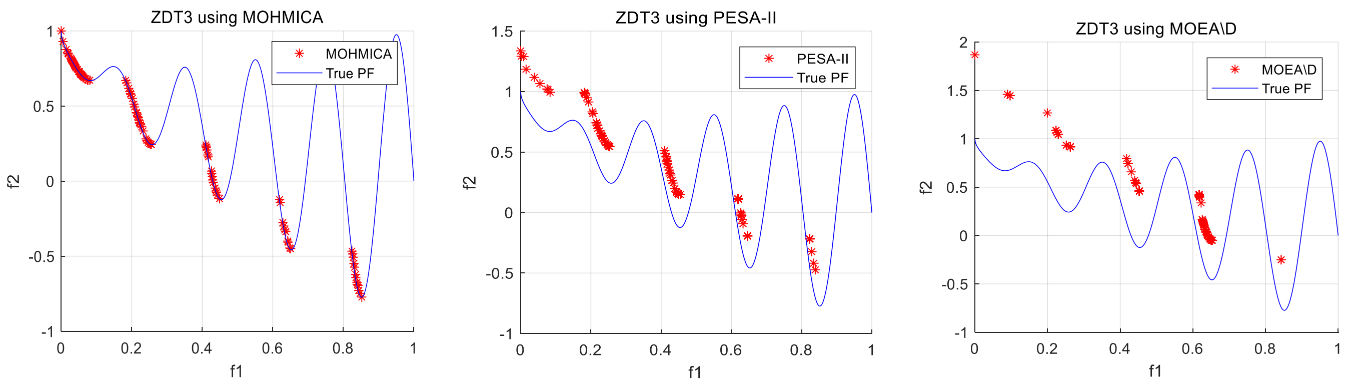

Figure 5. Pareto frontiers of ZDT3 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

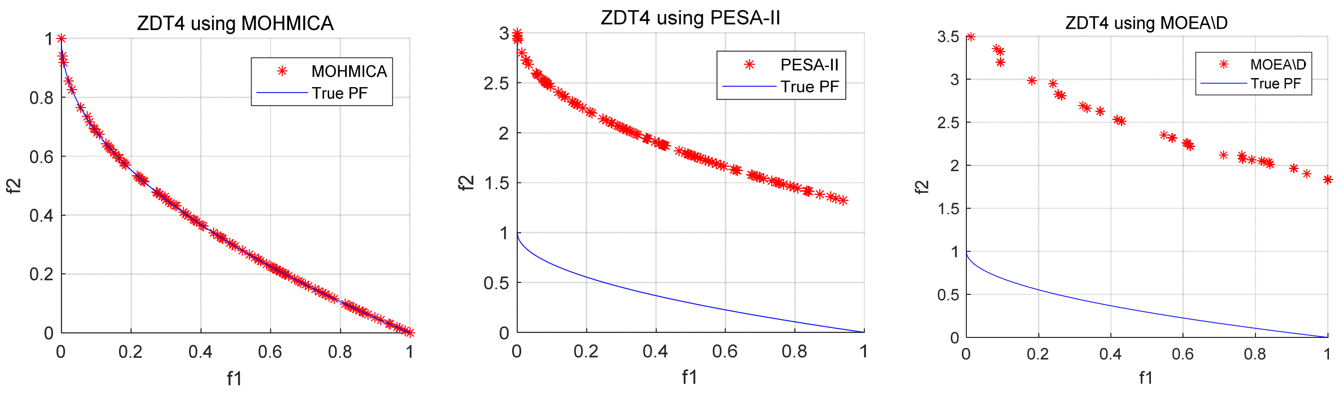

Figure 6. Pareto frontiers of ZDT4 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

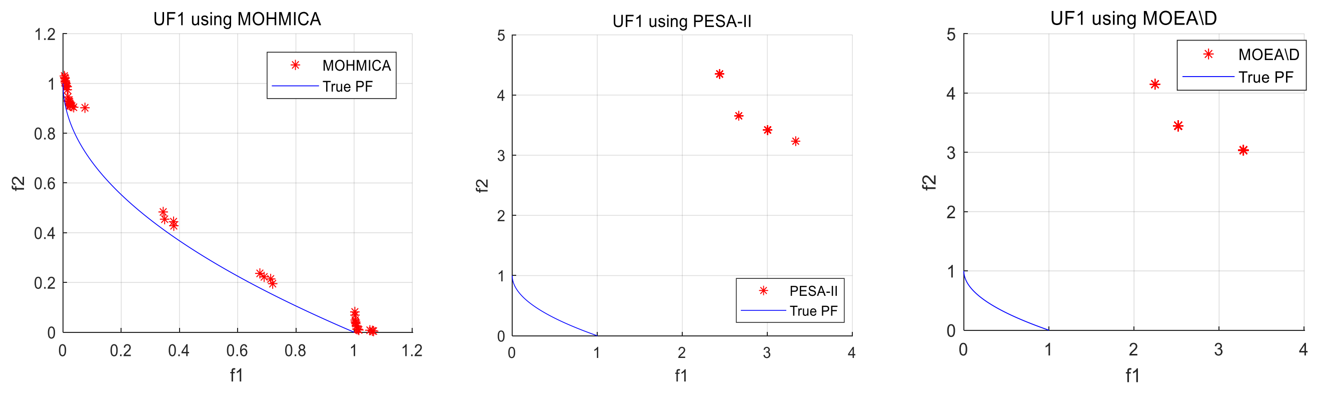

Figure 7. Pareto frontiers of UF1 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

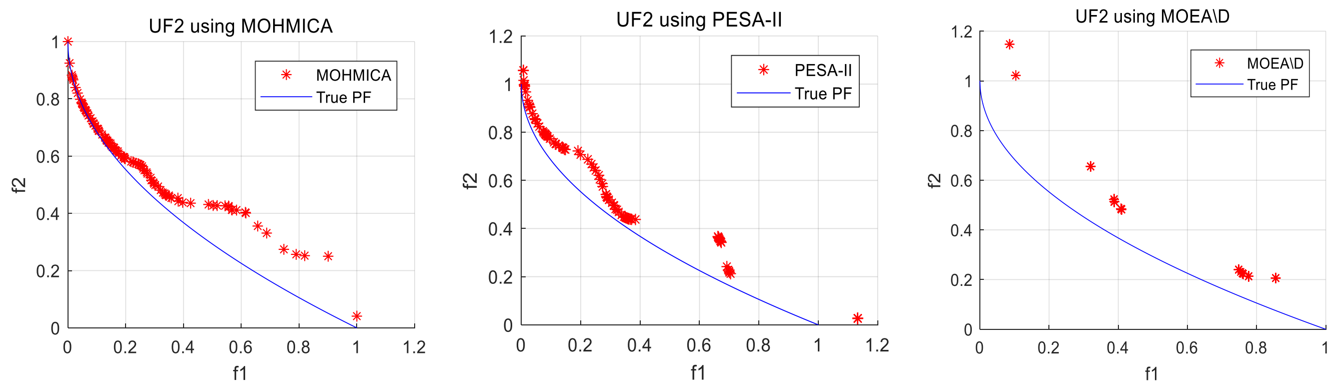

Figure 8. Pareto frontiers of UF2 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

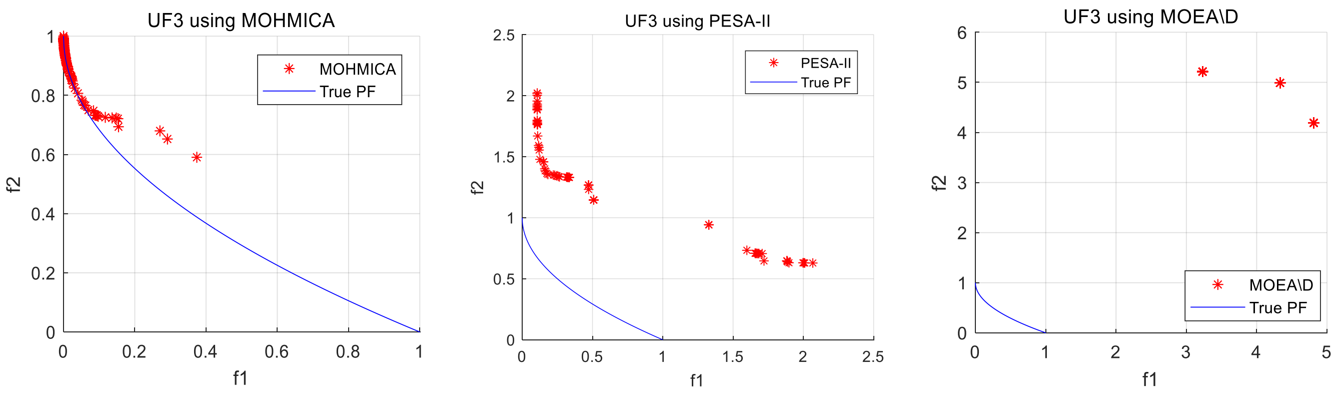

Figure 9. Pareto frontiers of UF3 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

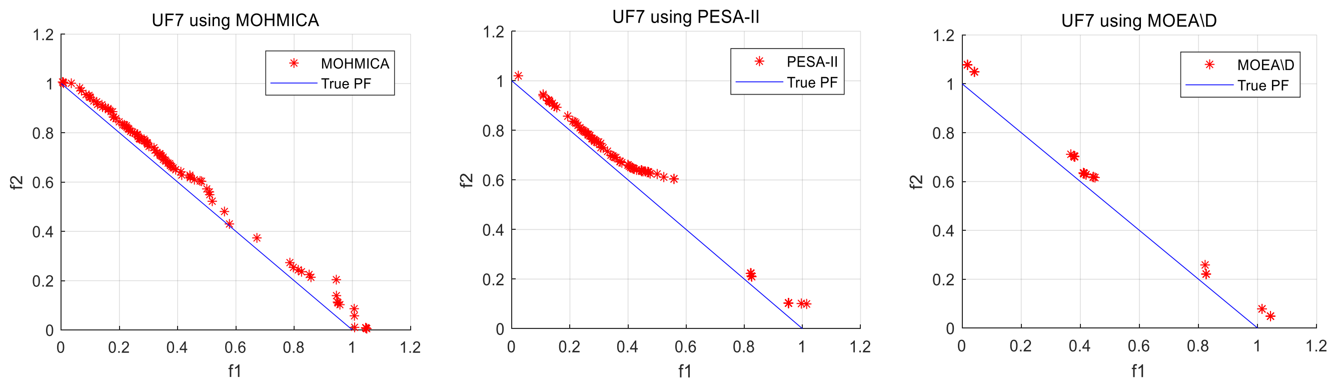

Figure 10. Pareto frontiers of UF7 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

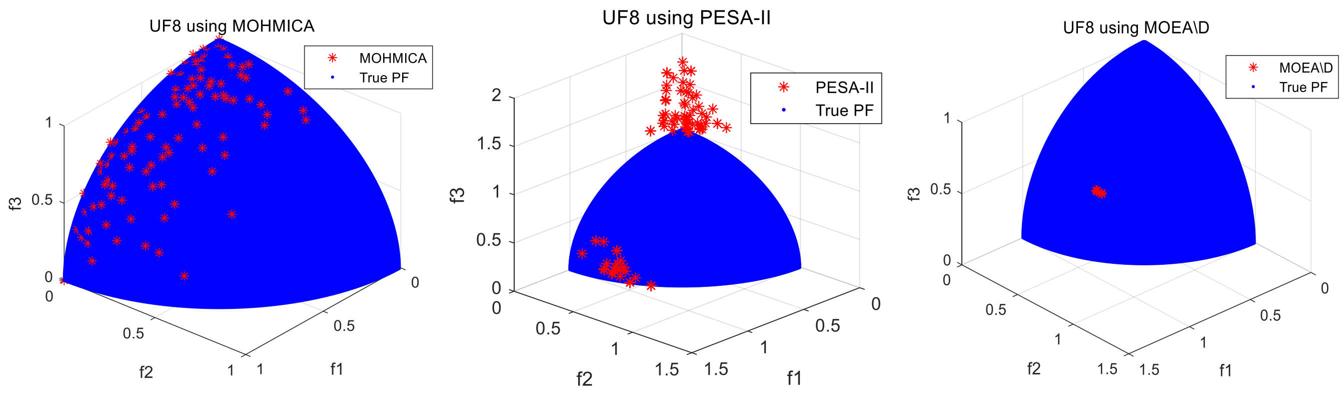

Figure 11. Pareto frontiers of UF8 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

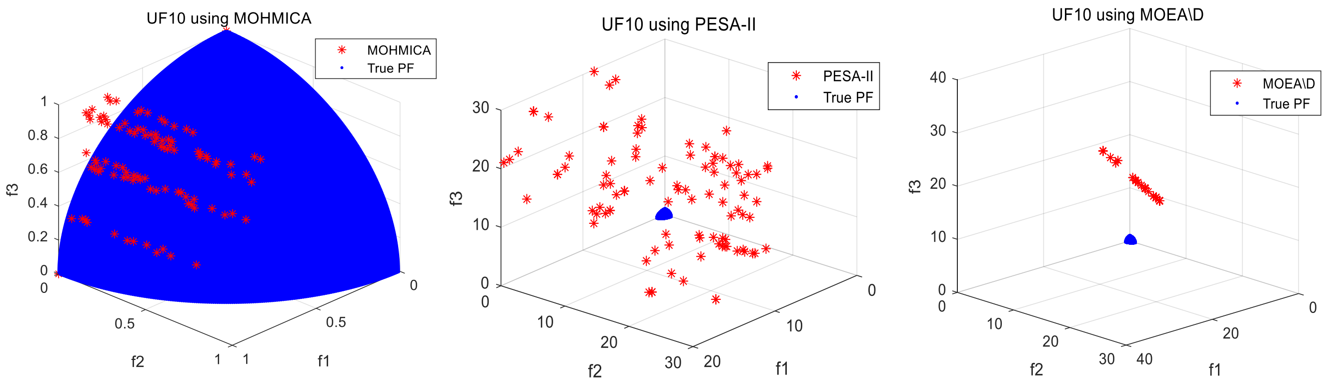

Figure 12. Pareto frontiers of UF10 benchmark function obtained by MOHMICA, PESA-II and MOEA\D.

4.2. A New Method for Evaluating Multi-Objective Optimization Algorithm

For the common metrics evaluating the quality of multi-objective optimization algorithms at present, CM, DM, GD and IGD all have some limitations. Specifically, CM and GD are convergence metrics from different perspectives. DM is a metric to evaluate the distribution of solutions in the approximate Pareto front. Although IGD is generally considered to be a comprehensive evaluation metric that can take into account the convergence and distribution of the solutions, it also has some limitations. On the one hand, a different number of the sampling points on the real Pareto front may affect the results of IGD; on the other hand, for those optimization problems with more than three objective functions, the convergence and distribution of the solutions obtained by algorithms cannot be seen from IGD because those solutions cannot be expressed visually. Therefore, it is of some theoretical significance to combine multiple metrics representing the convergence and distribution of the solutions of multi-objective optimization algorithms and propose a comprehensive evaluation method that can be expressed visually. The specific methods are as follows.

Firstly, each metric result of benchmark functions calculated by different algorithms is processed by logarithm. The specific calculation method is shown in Equation (7):

$$w = u - \lg v \tag{7}$$

In Equation (7), v represents the mean value of CM, DM, GD and IGD, respectively. $u = |[-\lg v]_{\min}| + 1$, where $[\bullet]$ represents the integer of $\bullet$. w is logarithmic processed data. Then, draw the radar map of each benchmark functions using w_{CM}, w_{DM},w_{GD} and w_{IGD} of different algorithms, as shown in Figures 13–16. The drawing method of radar maps is as follows. Starting from the origin point, the length of w_{CM}, w_{DM}, w_{GD} and w_{IGD} are the half diagonal respectively. w_{CM} and w_{DM} forms a diagonal of the quadrilateral of the radar map, because these two are the metrics that directly characterize convergence degree and distribution degree of the approximate Pareto front, respectively. w_{GD} and w_{IGD} constitutes another diagonal, because these two metrics represent the distance from the approximate Pareto fronts obtained by different algorithms to real Pareto fronts and the distance from real Pareto fronts to the approximate Pareto fronts obtained by different algorithms. The larger the area of the radar map is, the better the comprehensive result of the benchmark function obtained by each algorithm. For the 12 benchmark functions calculated by MOHMICA and comparing other algorithms in this paper, the larger the average area of the 12 radar maps of each algorithm, the stronger the comprehensive ability to calculate the multi-objective optimization problems. Moreover, from the actual value after logarithmic transformation, when the radar map areas of two algorithms calculating the same benchmark function, there is little performance difference between different algorithms. The calculation results are shown in Table 8.

From the results in Table 8, comparing with the average area of radar maps of different algorithms in this paper, MOHMICA is the largest, being at least 14.06% larger than the total area of other algorithms. It shows that the comprehensive ability of MOHMICA is also the strongest when calculating benchmark functions. Meanwhile, the number of times the radar maps with the largest area of MOHMICA is the most among all algorithms.

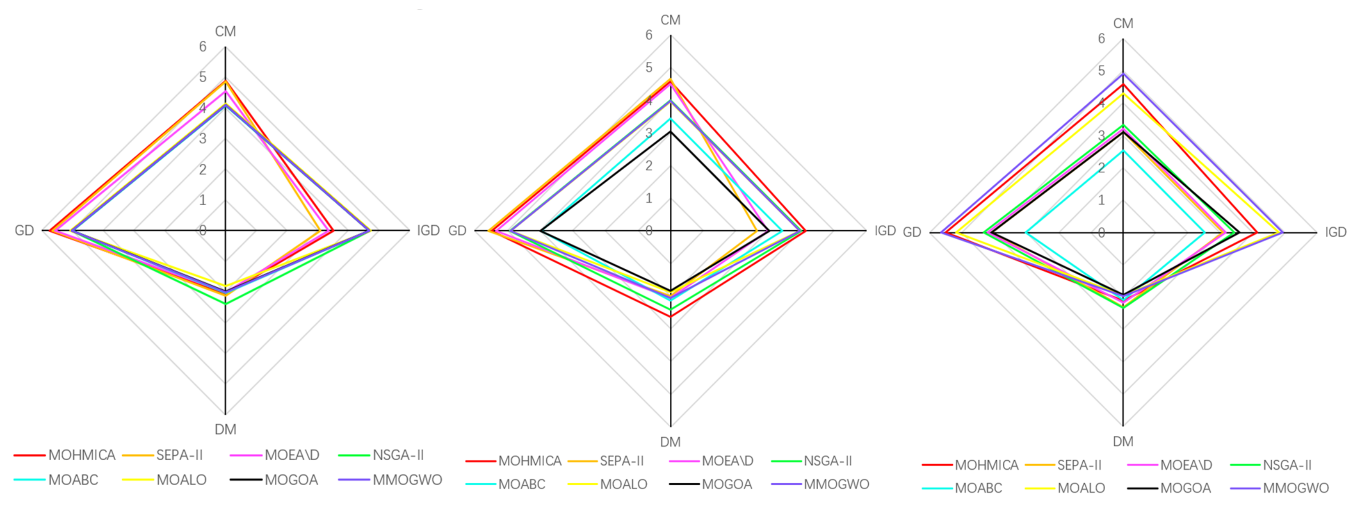

Figure 13. Comprehensive evaluation radar maps of SCH function (**left**), FON function (**center**) and ZDT1 function (**right**) calculated by eight different algorithms.

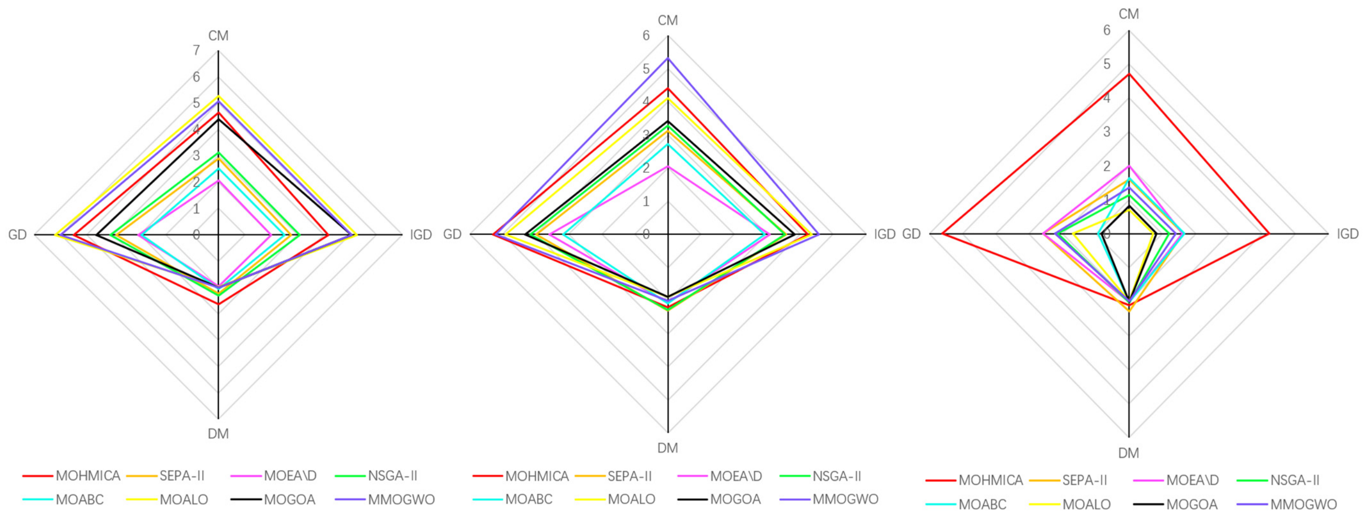

Figure 14. Comprehensive evaluation radar maps of ZDT2 function (**left**), ZDT3 function (**center**) and ZDT4 function (**right**) calculated by eight different algorithms.

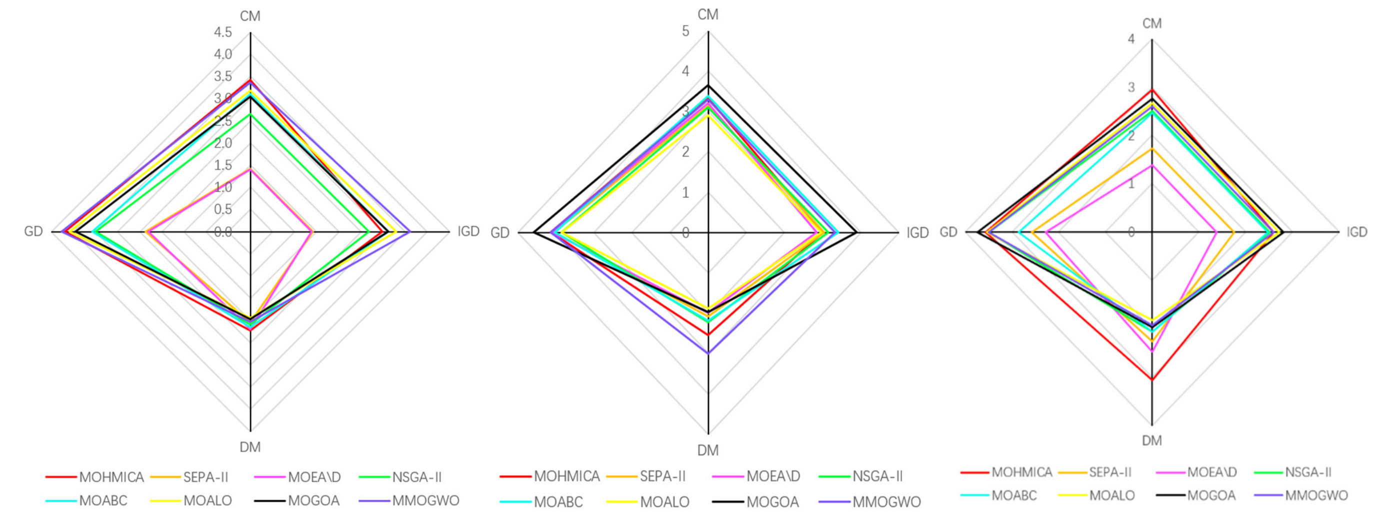

Figure 15. Comprehensive evaluation radar maps of UF1 function (**left**), UF2 function (**center**) and UF3 function (**right**) calculated by eight different algorithms.

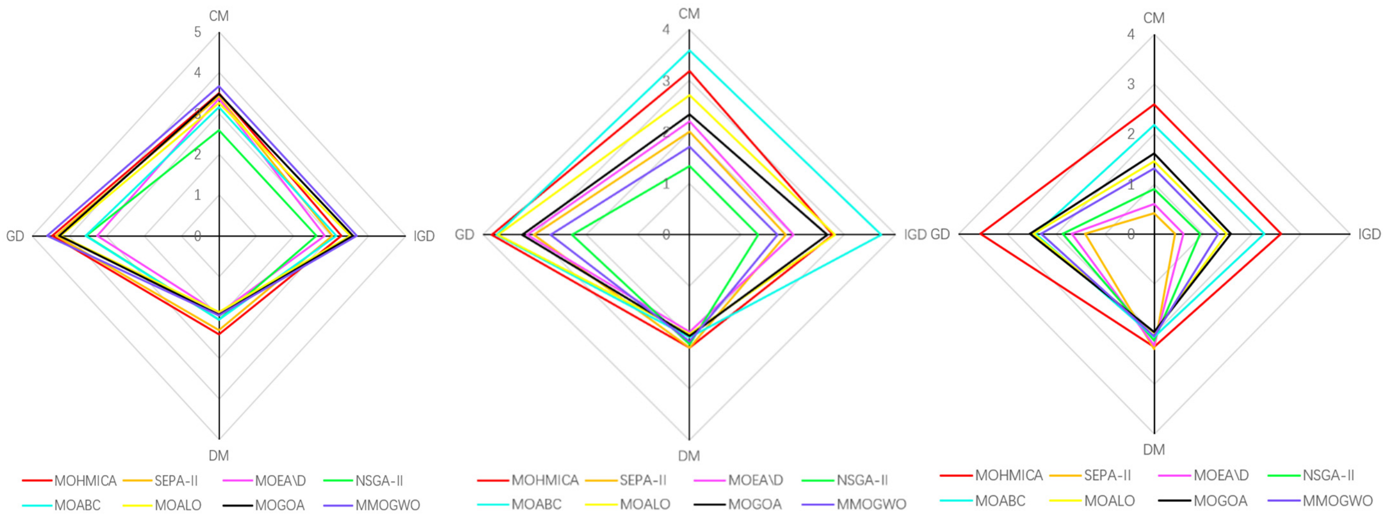

Figure 16. Comprehensive evaluation radar maps of UF7 function (**left**), UF8 function (**center**) and UF10 function (**right**) calculated by eight different algorithms.

Table 8. Comparison of radar map area calculated by eight algorithms for benchmark function.

Benchmark Functions		MOHMIICA	PESA-II	MOEA\D	NSGA-II	MOABC	MOALO	MOGOA	MMOGWO
SCH	Area	**32.039**	30.546	29.188	31.502	29.573	29.118	29.377	29.652
	Rank	**1**	3	7	2	5	8	6	4
FON	Area	**34.586**	27.338	27.199	28.592	20.651	25.751	17.089	26.727
	Rank	**1**	3	4	2	7	6	8	5
ZDT1	Area	32.170	19.397	19.743	22.000	12.790	31.193	19.253	36.284
	Rank	2	6	5	4	8	3	7	**1**
ZDT2	Area	35.139	17.062	10.128	19.528	12.468	41.670	31.032	**39.188**
	Rank	3	6	8	5	7	3	4	1
ZDT3	Area	31.471	20.557	13.685	21.355	14.691	27.816	21.673	**35.858**
	Rank	2	6	8	5	7	3	4	**1**
ZDT4	Area	**33.407**	8.173	8.334	5.329	4.715	3.241	2.323	6.052
	Rank	**1**	3	2	6	5	7	8	4
UF1	Area	20.244	6.533	6.648	14.635	17.522	18.844	17.676	**21.190**
	Rank	2	8	7	6	5	3	4	**1**
UF2	Area	20.892	18.08	18.031	18.635	20.688	16.299	**23.856**	23.463
	Rank	3	6	7	5	4	8	**1**	2
UF3	Area	**18.517**	8.636	7.069	13.751	12.081	14.037	15.459	13.754
	Rank	**1**	7	8	5	6	3	2	4
UF7	Area	22.816	21.079	15.972	14.261	17.439	19.857	21.024	**23.255**
	Rank	2	4	8	7	6	5	3	**1**
UF8	Area	17.810	10.273	10.610	6.260	**20.812**	15.198	12.771	8.328
	Rank	2	6	5	8	**1**	3	4	7
UF10	Area	**14.884**	2.493	3.209	4.246	9.889	6.770	7.328	6.083
	Rank	**1**	8	7	6	2	4	3	5
Mean area		**26.164**	15.847	14.151	16.674	16.109	20.816	18.238	22.486
The rank of mean area		**1**	7	8	5	6	3	4	2

5. Conclusions and Future Research

This paper aimed to address the shortcomings of HMICA that can only solve single-objective optimization problems and proposes the MOHMICA algorithm. In order to adapt to the characteristics of multi-objective optimization problems, MOHMICA updates the colony allocation strategy during the empire creation on the basis of HMICA, and increases the step of external archive.

In order to verify the performance of MOHMICA, this paper calculated 12 common benchmark functions, including 10 bi-objective benchmarks and 2 tri-objective benchmarks. Then, seven high-quality algorithms were compared to the proposed algorithm using four metrics: CM, DM, GD and IGD. After ranking and performing the Wilcoxon test, the proposed algorithm was found to have certain advantages over other algorithms for most metrics, but it is not enough to prove that the algorithm proposed in this paper has obvious advantages for each function. Therefore, a new comprehensive evaluating method called "radar map method" is proposed as the other knowledge contribution of this paper, which is used to evaluate comprehensive ability, including that of convergence and distribution of the approximate Pareto fronts obtained by different algorithms. The coordinate axis of the radar map includes CM, DM, GD and IGD. After evaluating algorithms that compare with MOHMICA using the radar map method, the comprehensive ability of MOHMICA was found to be the best among all algorithms.

For future research, there are three problems recommended to improve upon. First, in order to make the Pareto front distribution better than the algorithm proposed in this paper, when solving the optimization problem with more than two objective functions, the external archive strategy may need to be further improved. Second, in order to reduce time consumption and complexity when using MOHMICA to solve optimization problems, the operators in some of the steps may need to be replaced with simpler operators. Lastly, the application field needs to be considered. Using MOHMICA to solve real-world problems, including vehicle routing, industrial production management and production process scheduling optimization, are also important to explore in future research.

Author Contributions: Conceptualization, J.L.; methodology, J.L.; writing—original draft preparation, J.L.; writing—review and editing, J.L.; discussion, J.L., J.Z., X.J. and H.L.; supervision, J.Z.; project administration, J.Z.; funding acquisition, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Any information does not involve this endorsement.

Informed Consent Statement: This article does not contain any research on humans.

Data Availability Statement: The relevant data of this paper have been shown in the manuscript.

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

References

1. Schaffer, J.D. Multiple objective optimization with vector evaluated genetic algorithms. In Proceedings of the first international conference on genetic algorithms. In Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, 1 July 1985; Lawrence Erlbaum: Hillsdale, NJ, USA, 1985; pp. 93–100.
2. Fonseca, C.; Fleming, P. Genetic algorithms for multiobjective optimization: Formulation discussion and generalization. In Proceedings of the Fifth International Conference on Genetic Algorithms, Urbana, IL, USA, 1 June 1993; Morgan Kauffman Publishers: San Francisco, CA, USA, 1993; pp. 34–44.
3. Corne, D.W.; Jerram, N.R.; Knowles, J. PESA-II: Region-based selection in evolutionary multiobjective optimization. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), San Francisco, CA, USA, 7–11 July 2001.
4. Srinivas, N.; Deb, K. Multiobjective optimization using non-dominated sorting in genetic algorithms. *IEEE Trans. Evol. Comput.* **1994**, *2*, 221–248.
5. Deb, K.; Pratap, A.; Agarwal, S. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
6. 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]

7. Zhang, Q.; Li, H. MOEA\D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
8. Akbari, R.; Hedayatzadeh, R. A multi-objective artificial bee colony algorithm. *Swarm Evol. Comput.* **2012**, *2*, 39–52. [CrossRef]
9. Mirjalili, S. Grasshopper optimization algorithm for multi-objective optimization problems. *Appl. Intell.* **2018**, *48*, 805–820. [CrossRef]
10. Mirjalili, S.; Jangir, P. Multi-objective ant lion optimizer: A multi-objective optimization algorithm for solving engineering problems. *Appl. Intell.* **2017**, *46*, 79–95. [CrossRef]
11. Mirjalili, S.; Saremi, S.; Mirjalili, S. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Syst. Appl.* **2016**, *47*, 106–119. [CrossRef]
12. 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. *Mech. Eng.* **2008**, *57*, 722–748.
13. Liu, J.; Yang, Z.; Li, D. A multiple search strategies based grey wolf optimizer for solving multi-objective optimization problems. *Expert Syst. Appl.* **2020**, *145*, 113134. [CrossRef]
14. Khalilpourazari, S.; Naderi, B.; Khalilpourazary, S. Multi-Objective Stochastic Fractal Search: A powerful algorithm for solving complex multi-objective optimization problems. *Soft. Comput.* **2020**, *24*, 3037–3066. [CrossRef]
15. Got, A.; Moussaoui, A.; Zouache, D. A guided population archive whale optimization algorithm for solving multiobjective optimization problems. *Expert Syst. Appl.* **2020**, *141*, 112972. [CrossRef]
16. Abualigah, L.; Yousri, D.; Elaziz, M.A.; Ewees, A.A.; Al-Qaness, M.A.; Gandomi, A.H. Aquila Optimizer: A novel meta-heuristic optimization Algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
17. Abualigah, L.; Elaziz, M.A.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **2022**, *191*, 116158. [CrossRef]
18. Abualigah, L.; Diabat, A.; Mirjalili, S.; Elaziz, M.A.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
19. Atashpaz-Gargari, E.; Lucas, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 4661–4667. [CrossRef]
20. Aliniya, Z.; Keyvanpour, M.R. CB-ICA: A crossover-based imperialist competitive algorithm for large-scale problems and engineering design optimization. *Neural Comput. Appl.* **2019**, *31*, 7549–7570. [CrossRef]
21. Iyer, V.H.; Mahesh, S.; Malpani, R.; Sapre, M.; Kulkarni, A. Adaptive Range Genetic Algorithm: A hybrid optimization approach and its application in the design and economic optimization of Shell-and-Tube Heat Exchanger. *Eng. Appl. Artif. Intell.* **2019**, *85*, 444–461. [CrossRef]
22. Elsisi, M. Design of neural network predictive controller based on imperialist competitive algorithm for automatic voltage regulator. *Neural Comput. Appl.* **2019**, *31*, 5017–5027. [CrossRef]
23. Arya, Y. Impact of ultra-capacitor on automatic generation control of electric energy systems using an optimal FFOID controller. *Int. J. Energy Res.* **2019**, *43*, 8765–8778. [CrossRef]
24. Hosseinzadeh, A.Z.; Razzaghi, S.R.S.; Amiri, G.G. An iterated IRS technique for cross-sectional damage modelling and identification in beams using limited sensors measurement. *Inverse Probl. Sci. Eng.* **2019**, *27*, 1145–1169. [CrossRef]
25. Hajiaghaei-Keshteli, M.; Fard, A.M.F. Sustainable closed-loop supply chain network design with discount supposition. *Neural Comput. Appl.* **2019**, *31*, 5343–5377. [CrossRef]
26. Karimi, B.; Hassanlu, M.G.; Niknamfar, A.H. An integrated production-distribution planning with a routing problem and transportation cost discount in a supply chain. *Assem. Autom.* **2019**, *39*, 783–802. [CrossRef]
27. Fakhrzad, M.B.; Goodarzian, F. A Fuzzy Multi-Objective Programming Approach to Develop a Green Closed-Loop Supply Chain Network Design Problem under Uncertainty: Modifications of Imperialist Competitive Algorithm. *RAIRO Res. Oper.* **2019**, *53*, 963–990. [CrossRef]
28. Gharib, M.; Mohammad, S. A dynamic dispatching problem to allocate relief vehicles after a disaster. *Eng Optimiz.* **2020**, *53*, 1999–2016. [CrossRef]
29. Marandi, F.; Fatemi Ghomi, S.M.T. Integrated multi-factory production and distribution scheduling applying vehicle routing approach. *Int. J. Prod. Res.* **2019**, *57*, 722–748. [CrossRef]
30. Wang, S.; Liu, G.; Gao, S. A hybrid discrete imperialist competition algorithm for fuzzy job-shop scheduling problems. *IEEE Access* **2017**, *7*, 9320–9331. [CrossRef]
31. Zhang, H. Balancing Problem of Stochastic Large-Scale U-Type Assembly Lines Using a Modified Evolutionary Algorithm. *IEEE Access* **2018**, *6*, 78414–78424. [CrossRef]
32. Lei, D.; Li, M.; Wang, L. A two-phase meta-heuristic for multi-objective flexible job shop scheduling problem with total energy consumption threshold. *IEEE Trans. Cybern.* **2018**, *49*, 1097–1109. [CrossRef]
33. Enayatifar, R.; Yousefi, M.; Abdullah, A.H. MOICA: A novel multi-objective approach based on imperialist competitive algorithm. *Appl. Math. Comput.* **2013**, *219*, 8829–8841. [CrossRef]
34. Ghasemi, M.; Ghavidel, S.; Ghanbarian, M.M. Multi-objective optimal electric power planning in the power system using Gaussian bare-bones imperialist competitive algorithm. *Inform. Sci.* **2015**, *294*, 286–304. [CrossRef]

35. Mohammad, A.; Hamed, M. Multi-Objective Modified Imperialist Competitive Algorithm for Brushless DC Motor Optimization. *IETE J. Res.* **2019**, *65*, 96–103. [CrossRef]
36. Piroozfard, H.; Wong, K.Y.; Tiwari, M.K. Reduction of carbon emission and total late work criterion in job shop scheduling by applying a multi-objective imperialist competitive algorithm. *Int. J. Comput. Int. Syst.* **2018**, *11*, 805. [CrossRef]
37. Khanali, M.; Akram, A.; Behzadi, J. Multi-objective optimization of energy use and environmental emissions for walnut production using imperialist competitive algorithm. *Appl. Energy* **2021**, *284*, 116342. [CrossRef]
38. Li, M.; Lei, D. An imperialist competitive algorithm with feedback for energy-efficient flexible job shop scheduling with transportation and sequence-dependent setup times. *Eng. Appl. Artif. Intell.* **2021**, *103*, 104307. [CrossRef]
39. Kaveh, A.; Rahmani, P.; Eslamlou, D. An efficient hybrid approach based on Harris Hawks optimization and imperialist competitive algorithm for structural optimization. *Eng. Comput.* **2021**, 1–29. [CrossRef]
40. Li, M.; Su, B.; Lei, D. A novel imperialist competitive algorithm for fuzzy distributed assembly flow shop scheduling. *J. Intell. Fuzzy Syst.* **2021**, *40*, 4545–4561. [CrossRef]
41. Tao, X.-R.; Li, J.-Q.; Huang, T.-H.; Duan, P. Discrete imperialist competitive algorithm for the resource-constrained hybrid flowshop problem with energy consumption. *Complex Intell. Syst.* **2021**, *7*, 311–326. [CrossRef]
42. Luo, J.; Zhou, J.; Jiang, X. A modification of the imperialist competitive algorithm with hybrid methods for constrained optimization problems. *IEEE Access* **2021**, *9*, 161745–161760. [CrossRef]