Diversity Preference-based Many-Objective Particle Swarm Optimization Using Reference-Lines-based Framework

Deepak Sharma, Somnath Vats and Satyam Saurabh

Department of Mechanical Engineering, Indian Institute of Technology Guwahati, Assam-781039, India

Abstract

A simple yet effective diversity preference approach is developed and coupled with many-objective particle swarm optimization (PSO) for solving box-constraint optimization problems. Since the selection of global leaders in PSO is crucial for many-objective optimization, an approach is developed using the reference-lines-based framework to update these leaders. In the proposed approach, diversity is ensured first by making clusters of solutions for every reference line. Thereafter, only one solution from each cluster with minimum penalty-based boundary intersection (PBI) fitness is selected. In case any cluster of a line is empty, a solution with minimum PBI fitness with respect to the line is selected. This ensures the selection of isolated but sometimes dominated solutions. The proposed approach is then used for updating the archive of global leaders and for assigning them to particles in the swarm. The proposed algorithm, which is referred to as MaOPSO-DP, is tested on 3-, 5-, 8-, 10-, and 15-objective instances of DTLZ and WFG test problems. Results demonstrate the effectiveness of MaOPSO-DP over eight many-objective evolutionary and PSO algorithms from the literature.

Keywords: Diversity Preference, Many-Objective Optimization, PSO, Reference Lines, Global Leaders

1. Introduction

The field of many-objective optimization using evolutionary and swarm algorithms has attracted many researchers to develop an efficient and effective algorithm that can solve problems with a variety of complexities. The algorithms have been developed mainly using the Pareto-ranking-based [1], decomposition-based [2], or indicators-based [3] framework. Their main objective is to solve the optimization problem given in Equation (1) when the number of objectives is more than three, i.e., $M > 3$.

$$\begin{array}{ll} \text{Minimize:} & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})), \\ \text{subject to} & \mathbf{x} \in \Omega, \end{array} \tag{1}$$

where $\mathbf{f}(\mathbf{x})$ is the set of conflicting objectives, $f_i(\mathbf{x})$ is the i-th objective function, $\mathbf{x} \in \mathbb{R}^n$ is the set of decision variables, and Ω is the feasible search space. Since there are multiple

*Corresponding author: D. Sharma
Email address: `dsharma@iitg.ac.in` (Deepak Sharma, Somnath Vats and Satyam Saurabh)

optimal solutions to Equation (1), these solutions are compared using the concept of Pareto Dominance. It says that solution $\mathbf{x}$ *dominates* solution $\mathbf{y}$ (denoted as $\mathbf{x} \prec \mathbf{y}$) if and only if $\forall i \in (1, \ldots, M), f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ and $\exists j \in (1, \ldots, M), f_j(\mathbf{x}) < f_j(\mathbf{y})$. Any solution $\bar{\mathbf{x}}$ is said to be Pareto-optimal (PO) if $\nexists \mathbf{y} \in \mathbb{R}^n : \mathbf{y} \prec \bar{\mathbf{x}}$. A set of these Pareto-optimal solutions then constitutes the PO set.

It is worth mentioning that many real-world problems often consist of multiple objectives, as observed with parametric optimization of a bulldozer and its blade in soil cutting [4], topology optimization of continuum structures [5, 6, 7], water resource management [8], etc. to name a few. Therefore, an efficient algorithm is needed that can generate solutions close to or on the PO front.

The Pareto-ranking-based many-objective evolutionary and swarm algorithms solve the problem given in Equation (1) by first assigning Pareto-ranking to solutions and then applying diversity preserving operator. The major problem with such algorithms is the lack of selection pressure for many-objective optimization problem in which almost all solutions become non-dominated [9]. Therefore, the Pareto-ranking cannot differentiate solutions and selection, thus depends only on diversity preserving operator. It leads to a poor convergence and diversity of solutions on the PO front. Researchers target this problem by introducing relaxed Pareto-ranking procedures, such as fuzzy dominance [10], $L-$dominance [11], $\alpha-$dominance [12], etc. Still, these procedures cannot generate a well diverse set of solutions on the PO front. Researchers then focus on developing efficient environmental selection procedures. One of the efficient procedures is developed using the reference-lines-based framework in which solutions are selected through a set of reference lines or vectors. NSGA-III [1] uses this framework and develops the concept of niching via association of solutions with the reference lines. An efficient environmental selection procedure is developed by choosing solutions from less crowded regions. Using the same framework, algorithms like MOEA/DD [13], VaEA [14], many-objective PSO [15], etc. have been developed.

The decomposition-based many-objective algorithms decompose the problem given in Equation (1) into many scalar single-objective optimization sub-problems. All sub-problems are made using aggregate functions and are solved simultaneously. As per the survey [16], the first limitation is the generation of weight vectors that can help algorithms in generating a diverse set of solutions on the PO front. Another limitation is the selection of aggregate functions such as Weighted Sum (WS) method, Tchebycheff (TS) method, and Penalty-based boundary interaction (PBI) method. These methods always have shortcomings in approximating the widespread of the PO front. In order to address these limitations, MOEA/DD [13] uses both dominance- and decomposition-based approaches. REVA [17], on the other hand, uses angle-penalized distance with the decomposed objective space using reference lines and selects one solution from each subpopulation. I-DEBA [18] eliminates θ in the PBI method by giving precedence of a distance over others in the objective space. $\theta-$DEA [19] presents a new dominance relationship using the PBI method. These are a few decomposition-based algorithms for many-objective optimization that can generate a well diverse set of solutions on the PO front.

The indicator-based algorithms employ indicators to assign fitness to each solution in order to converge on the PO front and maintain diversity among solutions. Hypervolume is the most common indicator that has been used successfully for solving many-objective optimization problems. However, the main limitation is the high computation time that increases exponentially with the number of objectives. This limitation has been handled by

HypE [3] in which hypervolume is computed using Monte-Carlo simulation. Other indicators, such as R2 indicator [20], unary epsilon indicator [21], etc. have also been used. However, these algorithms fail to generate a well diverse set of solutions on the PO front.

The decomposition-based and indicator-based algorithms generally assign a composite fitness to every solution in the population or swarm for achieving convergence and diversity simultaneously. The Pareto-based algorithms, however, assign a rank to every solution for convergence followed by a diversity preserving operator. In spite of that, there is another class of algorithms that prefers diversity first followed by dominance. For example, $\theta-$DEA [19], MPSO/D [22], DoD [23] are few such algorithms that make clusters of solutions using the reference-lines-based framework and select one solution from each cluster using either Pareto-ranking or new dominance-relationship. Their performance on solving box-constrained many-objective optimization problems motives us to employ the diversity preference approach with particle swarm optimization (PSO).

We know that PSO is popular for its faster convergence, but maintaining diversity among particles of a swarm is always challenging for multi- and many-objective optimization. From the literature, it is evident that the leader selection is crucial for any multi-objective PSO since we need to emphasize non-dominated solutions and to maintain diversity among them. Therefore, the literature focuses on the selection of global leaders that can steer the search toward the PO front. The global leaders, which are generally non-dominated, are thus selected through various methods, such as at random [24], using $\sigma-$ method [25], using clustering [22], etc. In this paper, a diverse set of global leaders is selected through the proposed diversity preference approach. The following are the contributions of the paper.

- A diversity preference approach is developed using the reference-lines-based framework in which clusters of solutions for every reference line are made, and only one solution from each cluster is selected using the PBI method.

- The approach is further extended for those reference lines for which clusters have no solution. In that case, a solution, which is not selected yet, with minimum PBI fitness to the empty line is selected.

- The proposed approach is then used to update the archive of global leaders in every generation. Thereafter, the same approach is used for assigning a global leader to each particle in the swarm.

A comparative study of the proposed PSO algorithm is presented with eight evolutionary and swarm algorithms on various instances of DTLZ and WFG problems using two indicators and the Wilcoxon significance test.

The remainder of this paper is organized as follows. Section 2 presents details of PSO and the relevant studies of many-objective PSO algorithms. Section 3 presents the diversity preference approach, the proposed algorithm's framework, and other operators required for developing the algorithm. Section 4 presents results based on inverse generalized distance and hypervolume indicators. It also presents the average performance score of all algorithms. Section 5 concludes the paper with future work.

2. PSO and Relevant Studies for Many-Objective Optimization

2.1. PSO for Multi-Objective Optimization

The framework of PSO consists of a swarm that is initialized randomly. This swarm has N number of particles, essentially potential solutions for solving Equation (1). The framework also consists of two archives that are the archive of global leaders and the archive of local leaders. It is noted that the purpose of these archives is to store global and local leaders so that these leaders can be used for velocity update, etc. These leaders help PSO to explore the search space of the optimization problem. Initially, particles of the swarm are copied into the archive of local leaders and the non-dominated solutions from the swarm to the archive of global leaders. Inside a standard loop of generations, the global leader is assigned to each particle in the swarm. It is noted that there are many potential leaders in the archive, which are non-dominated. Thereafter, the velocity ($\mathbf{v}_i(t+1)$) of the particles in the generation (t) is updated as

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1 r_1 (LB_i(t) - \mathbf{x}_i(t)) + c_2 r_2 (GB_i(t) - \mathbf{x}_i(t)), \tag{2}$$

where w is the inertia weight of the particle, c_1 and c_2 are the coefficients for exploitation and exploration, r_1 and r_2 are the random numbers between $[0, 1]$, $LB_i(t)$ is the $i-$th local leader in the $LB(t)$ archive at t-th generation, and $GB_i(t)$ is the $i-$th global leader in the $GB(t)$ archive at t-th generation. The position of particle i is then updated as

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1). \tag{3}$$

The new position of each particle is evaluated, and the archives of the global and local leaders are updated. The counter for the generation (t) is then increased by one. PSO finally terminates when ($t > T$), where T is the maximum number of allowed generations.

There are plenty of PSO algorithms for multi-objective optimization in the literature. However, when PSO algorithms have to target many-objective optimization problems, mainly $M > 3$, the challenges like an update of global and local leaders' archives, assignment of global leader to each particle in the swarm, convergence, and diversity of solutions, non-domination of all solutions [1, 26], etc. need to be addressed. Therefore, we target those relevant PSO algorithms which are designed for many-objective optimization.

2.2. Studies on Many-Objective PSO Algorithms

Mostaghim and Schmeck [25] proposed a distance-based ranking method that computed a distance between two solutions in the objective space. The proposed method was then used for selecting global leaders. Britto and Pozo [27] used reference points that were generated on a hyperplane. Using a reference point, the farthest solution from this point was removed for updating an external archive. NWSum method was used for updating the leaders in the swarm. Dai et al. [22] proposed decomposition-based PSO, which made N clusters of solutions with respect to $N-$weight vectors by using the angle between the solution and vector. Along with the position update, the crossover was used to explore the feasible search space. The global leader to each particle was assigned from the neighboring particle, which made the smallest angle with the center vector of the cluster.

Figueiredo et al. [15] used the reference-lines-based framework of NSGA-III [1]. A density operator was used that projected solutions on the hyperplane. A convergence operator using achievement scalarizing function was used for assigning the fitness to each solution. An archive of non-dominated solutions was maintained through density and convergence operators. The global leaders were assigned from the external archive by using the extreme solutions and tournament selection. Hu et al. [28] proposed a two-stage strategy for dominance-based PSO in which $M-$extreme points were used to generate the rest of the population near to them. The external archive and selection of global leaders were performed using a parallel cell coordinate system. Zhu et al. [29] presented an immune-based evolutionary search for an external archive that involved cloning, evolutionary search by crossover and mutation operators, and crowding degree operator. The decomposition-based approach was adapted to update local leaders, and the global leader was selected randomly from the external archive. Pan et al. [30] used the decomposition-based approach in which the diversity was preserved through the angle-based association with the reference vectors. A solution from each association of the reference vector was selected using fitness similar to the PBI method. An external archive was maintained through angle-based diversity and PBI-based fitness. Clustering in the variable space was also performed using the k-means clustering method. Two-stage criteria were used to select global leaders that were through angle-based association and randomly.

Liu et al. [31] proposed a balanceable fitness estimation for preserving diversity in the swarm. The fitness consisted of diversity distance and convergence distance. The diversity distance was measured in the normalized objective space by comparing the objective values of two solutions. The convergence distance of a solution was calculated with respect to the reference point. An external archive was updated using the proposed fitness, and an evolutionary search was performed on the archive using crossover and mutation operators. Liu et al. [32] proposed a bottleneck objective learning for improving the convergence of multiple swarms in PSO. These swarms communicated through an external archive of non-dominated solutions. This archive was updated using the niching concept of NSGA-III. Qin et al. [33] developed a decomposition-based PSO using different ideal points on the reference vectors. The reference vectors were used to maintain diversity via angle-based association. The distance between the reference point on the reference vector and solution was used for convergence. An external archive was maintained through a parameter called CAD, which has angle value in the numerator and distance value in the denominator. The solution with the least CAD value was retained from each reference vector.

Apart from dominance-based PSO, researchers have focused on other algorithms as well. For example, Luo et al. [21] proposed an indicator-based PSO that used a unary epsilon indicator. It was used to update the local leader of a particle when the Pareto-ranking of both were the same. An external archive was maintained in which half of its size was filled using non-dominated solutions that were closed to the reference vectors. The rest of the size of the archive was filled using the indicator. The closest non-dominated solutions from the current swarm and the external archive to the reference vectors were selected for assigning a global leader to a particle. Yang et al. [34] used intuitionistic fuzzy dominance-based PSO for sorting particles in the swarm and new particles after position update. The new particles were generated through a double search strategy. A set of reference points was used for selecting local and global leaders using the PBI method. Li et al. [35] proposed a dominance difference approach that made a population size matrix. Each column of the matrix signified the difference among solutions when almost all of them were non-dominated. The same approach was used to select global and local leaders. The diversity among solutions was preserved through the L_p norm. An external archive was maintained and updated with

50% solutions using the approach. Xiang et al. [36] used historical solutions to guide the flight of each particle in the swarm using scalar projections. A diverse set of solutions was chosen using two fitness estimators that are L_2 norm with respect to the reference point and summation of objectives in the normalized objective space. The selection of the reference point was made by estimating the curvature of the PO front.

It can be observed from the literature that various efforts have been made to develop efficient many-objective PSO. These efforts focus mainly on selecting a diverse set of solutions for the archive so that the global leaders for many-objective PSO can be selected using various methods/approaches such as distance-based ranking method, decomposition-based approach, parallel cell coordinate system, balanceable fitness, etc. The selected leaders then steer the search of PSO toward the PO front. Moreover, many such algorithms have been developed using the decomposition-based approach because of the promising results. However, none of them have focused on retaining isolated solutions for preserving diversity. In the following section, the diversity preference approach is discussed, and details are given of coupling this approach for many-objective PSO.

3. Proposed Diversity Preference-based Many-Objective PSO: MaOPSO-DP

In this section, we first discuss the proposed diversity preference approach and thereafter, the reference-lines-based framework on which the proposed approach is employed.

3.1. Proposed Diversity Preference Approach

The proposed diversity preference approach is developed using the reference-lines-based framework, which has been successfully implemented for developing algorithms like NSGA-III [1], $\theta-$DEA [19], etc. The main task of this approach is to update global leaders and also to assign global leaders to particles in the swarm in every generation. Moreover, our earlier experiments and results of DoD [23] and LEAF [37] on the same framework motivate us to use their concepts of diversity preference with many-objective PSO (MaOPSO).

The diversity preference approach is explained using a case of 2-objective in which the objective space is decomposed by $N = 9$ reference lines (R1,. . . , R9), as shown in Fig. 1. These lines are drawn using the structured reference points generated using [38]. In the present case, there are ($2N = 18$) solutions in the swarm from which N solutions are to be selected. In the proposed approach, the clusters for each reference line are made as shown in Fig. 1 by calculating the Euclidean distance ($dist(\mathbf{s}, \mathbf{r})$) between the solution ($\mathbf{s}$) and the reference line ($\mathbf{r}$), which is given as

$$dist(\mathbf{s}, \mathbf{r}) = ||(\mathbf{s} - \mathbf{r}^T\mathbf{s}\mathbf{r}/||\mathbf{r}||^2)||. \tag{4}$$

A solution having minimum $dist(\mathbf{s}, \mathbf{r})$ value with respect to the line ($\mathbf{r}$) is included in the cluster of the same line. For example, the cluster for reference line 'R1' is made of only solution '1', the cluster for reference line 'R2' is made of solutions '11' and '12', and so on. Once these clusters are made, only one solution from each cluster is chosen using the PBI fitness, which is given as

$$F(\mathbf{s}) = d(\mathbf{s}, \mathbf{r}) + \theta \times dist(\mathbf{s}, \mathbf{r}), \tag{5}$$

where θ is a penalty parameter, and $d(\mathbf{s}, \mathbf{r})$ is the distance, which is calculated as

$$d(\mathbf{s}, \mathbf{r}) = \mathbf{s}^T\mathbf{r}/||\mathbf{r}||. \tag{6}$$

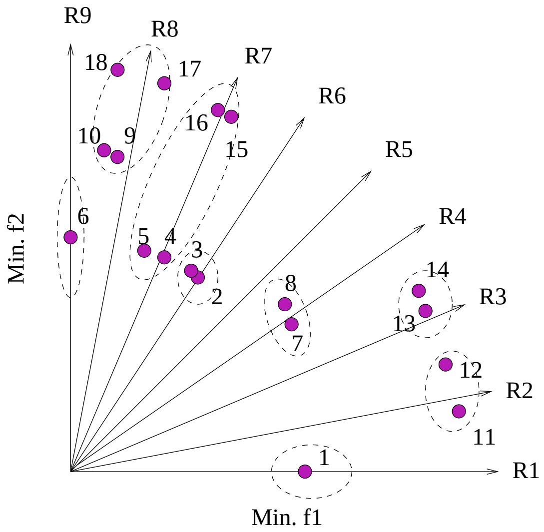

Figure 1: The diversity preference approach using the reference-lines-based framework. The clusters of solutions are shown using elliptical curves.

The solution, which shows a minimum $F(\mathbf{s})$ value, is selected from each cluster of the reference lines. For example, solution '1' is selected from the cluster of 'R1', solution '11' is selected from the cluster of 'R2', so on. Therefore, we can select eight solutions from the clusters that are '1', '11', '13', '7', '2', '4', '9', and '6'. We can see that there is no solution with the cluster of the reference line 'R5'. In this case, a solution, which is not selected yet and having minimum PBI fitness to the reference line 'R5', is selected. From the figure, solution '8' can be selected using the minimum PBI fitness.

The major steps of the diversity preference approach are presented in two stages using Algos. 1 and 2. In the stage-1, those clusters of reference lines are considered which are having at least one solution in Algo. 1. The PBI fitness of solutions in a cluster C_i is calculated with respect to the reference line i in Step 3. Solution ($\mathbf{s}$) with minimum PBI fitness is then chosen for the reference line i.

Algorithm 1 Stage-1 of the diversity preference approach

1: % Stage-1: For a cluster of reference line (i) which is having at least one solution
2: **for** every solution $\mathbf{x} \in C_i$, where $C_i \neq \emptyset$ **do**
3: Calculate PBI fitness of solution $\mathbf{x}$ with respect to the reference line i using Equation (5)
4: **end for**
5: Choose solution $\mathbf{s} \in C_i$ with minimum PBI fitness for the reference line i.

In stage-2, the clusters of reference lines, which have no solution, are considered in Algo. 2. In Step 3, the PBI fitness of solutions assigned to other clusters is calculated with respect to the reference line j. Solution ($\mathbf{s}$) with minimum PBI fitness is chosen for the reference line j.

Algorithm 2 Stage-2 of the diversity preference approach

% Stage-2: For a cluster of reference line (j) which is having no solution
for every solution $\mathbf{x} \in C_i$, where $C_i \neq \emptyset$ **do**
  Calculate PBI fitness of solution $\mathbf{x}$ with respect to the reference line j using Equation (5)
end for
Choose solution $\mathbf{s} \in C_i$ with minimum PBI fitness for the reference line j

We can observe that a diverse set of solutions is selected in which preference is given to diversity first by selecting only a single solution from each cluster. This idea is motivated by the LEAF algorithm [37] in which the solutions are first ranked using the Pareto-ranking, and then clusters are made using association and re-association. The solution with the minimum rank from each cluster is then chosen. In the case of a tie, the solution with a minimum ($dist(\mathbf{s}, \mathbf{r})$) is chosen. However, in the proposed approach, the clusters are made first, and one solution with minimum PBI fitness from each cluster is selected.

Another observation is that a dominated solution based on the Pareto-ranking can be selected over a non-dominated solution in the proposed approach. In Fig. 1, solution '9' (dominated solution) is chosen over solution '5' (non-dominated solution). This idea is motivated by the DoD algorithm [23] in which the clusters are made, and then non-dominated ranking is performed independently in each cluster. However, the proposed approach selects a solution from each cluster using the PBI fitness.

In addition to selecting a diverse set of solutions, the proposed approach selects a solution for those lines which have an empty cluster. In this case, a solution that is not selected yet, with minimum PBI fitness with respect to the line is considered. However, DoD and LEAF algorithms select solution using ($dist(\mathbf{s}, \mathbf{r})$) only.

3.2. Proposed PSO using Reference-Lines-based Framework

The reference-lines-based framework for many-objective optimization is developed in the literature with the help of reference lines for decomposing the objective space and then selecting a diverse set of solutions using these reference lines. In this framework, a set of reference points is generated using [38] on a unit hyperplane, and the lines are drawn from the origin to these reference points. The number of reference points $|H|$ is calculated as

$$|H| = \binom{M+p-1}{p}, \tag{7}$$

where M is the number of objectives, and p is the number of equal division on each objective axis. Fig. 2 shows a 3-objective case in which 15 reference points are generated by using $p = 4$. The figure also shows a reference line. With the help of the Euclidean distance ($dist(\mathbf{s}, \mathbf{r})$), solutions in the population or swarm are associated or made clusters with the closest reference line. This distance measure helps in selecting a diverse set of solutions, as discussed in Sections 1 and 2.

The same reference-lines-based framework is used for the proposed MaOPSO, which is presented in Algo. 3. It starts with initializing random swarm ($P(t)$) of size N and keeping the archives of the global leaders ($GB(t)$) and the local leaders ($LB(t)$) empty. Thereafter,

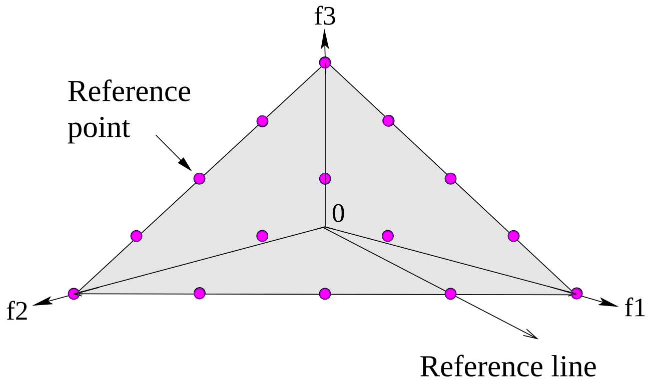

Figure 2: A set of reference points generated on a unit hyperplane is shown.

clusters of solutions of $P(t)$ are made for every reference line in Step 2. Since clusters are made in the normalized objective space, it requires normalization of $P(t)$. Initially, the same swarm of $P(t)$ is assigned to $GB(t)$ and $LB(t)$ in Step 3. Thereafter, an extreme point is calculated using the Nadir point from $P(t)$ in Step 4. This point is utilized later in normalization. In the standard loop of generations, the global leaders from the $GB(t)$ archive are assigned to each particle of the swarm in Step 6 using the diversity preference approach discussed in Section 3.1. The velocity and position of each particle in the swarm are updated in Steps 7 and 8. The evolutionary search is applied on the leaders of the $GB(t)$ archive using the SBX crossover operator and polynomial mutation operator [39]. Since we have the swarm and two archives of $GB(t)$ and $LB(t)$, these swarm and archives are combined together, and clusters are made in Step 10. The archives of $GB(t)$ and $LB(t)$ are updated in Steps 11 and 12 using the diversity preference approach. The generation counter (t) is then increased by one. The algorithm terminates when (t) reaches the maximum number of allowed generations. As an output, the non-dominated solutions from the $GB(t)$ archive are reported. In the following sections, details of the major steps of Algo. 3 are presented.

*3.3. Clustering of Solutions: **Clustering** in Steps 2 and 10 of Algo. 3*

As the name suggests, it is used to make clusters of solutions for the reference lines. Algo. 4 presents three major steps that are required to make clusters in the normalized objective space. In Step 1, the non-dominated sorting of NSGA-II [40] is used, which is required in normalization. Thereafter in Step 2, solutions of $R(t)$ are normalized by adopting the normalization scheme of LEAF [37].

The major steps of the normalization scheme are presented in Algo. 5. In this scheme, the extreme point in Step 4 of Algo. 3 is used to normalize solutions. Solutions of $R(t)$ are thus first translated by subtracting them with the ideal point, which is given as

$$\hat{\mathbf{f}}_i(\mathbf{x}) = \mathbf{f}_i(\mathbf{x}) - z_i^{min},\ \forall i \in M,\ \forall \mathbf{x} \in R(t), \tag{8}$$

where z_i^{min} is the i-th component of the ideal point ($\mathbf{z}^{min}$). This translation makes the new ideal point at the origin. Thereafter, the extreme solution along each objective axis is found

Algorithm 3 Reference-Lines-based Framework for the proposed MaOPSO

Input: $t = 1$, M : number of objectives, N : swarm size, H : reference points, T : maximum number of generations

Output: A set of non-dominated solutions from the $GB(t)$ archive

1: Initialize random swarm ($P(t)$) and the archives of $GB(t) = \emptyset$ and $LB(t) = \emptyset$;
2: Clustering of solutions: **Clustering**($P(t)$);
3: Assign particles to $GB(t) = P(t)$ and to $LB(t) = P(t)$;
4: Compute extreme point: $\mathbf{e} = (e_1, e_2, \dots, e_M)^T$ such that $e_j = \max_{\mathbf{x} \in P(t)} f_j(\mathbf{x})$ and $\mathbf{x}$ is a non-dominated solution;
5: **while** $t \leq T$ **do**
6: Assign global leader from the $GB(t)$ archive to each particle in the swarm: **Assign_GB**;
7: Update velocity of particles: **Update_Velocity**;
8: Update position of particles: **Update_Position**;
9: Perform evolutionary search on the leaders of the $GB(t)$ archive: **Evolutionary_Search**($GB(t)$);
10: Clustering of solutions: **Clustering**($M_t = P(t) \cup GB(t) \cup LB(t)$);
11: Update the archive of local leaders: $LB(t) =$ **Update_LB**($P(t), GB(t)$);
12: Update the archive of global leaders: $GB(t) =$ **Update_GB**(M_t);
13: $t = t + 1$
14: **end while**

using the augmented scalarizing function, which is given as

$$\mathbf{z}_j^e = \hat{\mathbf{f}}(\mathbf{x}), \mathbf{x} : \min_{\mathbf{x} \in R(t)} \left(\max_{i=1}^{M} \hat{f}_i(\mathbf{x}) / \mu_i \right), \tag{9}$$

where $\mathbf{z}_j^e$ is the extreme solution corresponding to $j-$th objective, $\hat{f}_i(\mathbf{x})$ is the $i-$th component of $\hat{\mathbf{f}}(\mathbf{x})$, and μ_i's are the weights for each objective. In order to calculate $j-$th component, i.e., $\mathbf{z}_j^e$, $\mu_j = 1$, and the rest of the weights are kept 10^{-6}.

The extreme solutions from every objective axis (Z) are then used to make the hyperplane. This hyperplane intersects the objective axes to find the intercepts on them. Since we solve a system of linear equations for determining the intercepts, the degenerate cases, such as duplicate solutions in (Z) or negative intercept, can occur. It can be seen that the Nadir point is determined when any degenerate case is encountered, as shown in Steps 4 and 8. For these cases, each component of the extreme point ($\mathbf{e}$) is compared in Step 14 and gets updated in Step 15. If there is no degenerate case, the computed intercept in Step 6 is used to update

Algorithm 4 Clustering($R(t)$)

1: $(F_1, F_2, \dots) =$ Non-dominated sorting ($R(t)$);
2: $\bar{R}_t =$ Normalize($R(t)$); %Using Algo. 5
3: MakeClusters($\bar{R}_t$); %Using Algo. 6

the extreme point ($\mathbf{e}$) in Step 10. At last, solutions are normalized using the extreme point ($\mathbf{e}$) in Step 18.

Algorithm 5 Normalization Scheme of LEAF [37]

1: Translate the solutions of $R(t)$ using Equation (8);
2: Compute extreme solutions (Z) using Equation (9);
3: **if** Duplicate solutions exist in Z **then**
4:     Compute the nadir point $(\mathbf{z}^N)$ from the current non-dominated front;
5: **else**
6:     Compute intercept $\mathbf{a} = (a_1, a_2, \ldots, a_M)^T$ from (Z);
7:     **if** Any intercept is negative **then**
8:         Compute the nadir point $(\mathbf{z}^N)$ from the current non-dominated front;
9:     **else**
10:         Update the extreme point ($\mathbf{e}$) using the intercept, i.e., $e_j = a_j,\ \forall j \in \{1, \ldots, M\}$;
11:     **end if**
12: **end if**
13: **if** Duplicate solutions exist or negative intercept is found **then**
14:     **if** $z_j^N < e_j$, where $j \in \{1, \ldots, M\},\ z_j^N \in \mathbf{z}^N$ **then**
15:         $e_j = z_j^N$;
16:     **end if**
17: **end if**
18: $\bar{f}_j(\mathbf{x}) = \hat{f}_j(\mathbf{x})/e_j, \forall \mathbf{x} \in R(t),\ \forall j \in \{1, \ldots, M\}$;

After normalization, the clustering of solutions is performed in the normalized objective space in Step 3 of Algo. 4. The major steps of clustering are presented in Algo. 6 in which the reference lines are computed in Step 1 using the reference points and the origin, as discussed earlier in Section 3.1. Thereafter, the distance ($dist(\mathbf{x}, \mathbf{w})$) is calculated for all solutions in $R(t)$ with every reference line. Every solution $\mathbf{x}$ having minimum $dist(\mathbf{x}, \mathbf{w})$ value with a reference line $\mathbf{w}$ is included in the cluster of the same line.

Algorithm 6 MakeClusters($\bar{R}_t$)

1: Compute every reference line $\mathbf{w}$ using the reference points generated through [38] and the origin;
2: Compute the distance ($dist(\mathbf{x}, \mathbf{w})$) for all solutions $\mathbf{x} \in R(t)$ to every reference line $\mathbf{w}$ using Equation (4) in the normalized objective space;
3: Include solution $\mathbf{x}$ to the cluster of line $\mathbf{w}$ which has the minimum $dist(\mathbf{x}, \mathbf{w})$ value;

3.4. Assign Global Leader: **Assign_GB** *in Step 6 of Algo. 3*

In the standard loop of generations, the first step of the proposed framework is to assign a global leader from the archive of $GB(t)$ to each particle in the swarm. The major steps of **Assign_GB** are presented in Algo. 7. Since the diversity preference approach is used for assigning the global leader, the task is divided into two stages. In the stage-1, the clusters of the reference lines that are having at least one solution are considered in Step 2. In Step 3, a solution ($\mathbf{s}$) is selected using Algo. 1. The same solution is assigned as the global leader to the particles in the swarm that belong to the cluster of the reference line i. In order to avoid the selection of the same solution, it is removed from C_i in Step 5 for further consideration. In the stage-2, the reference line (j) having no solution in its cluster is chosen in Step 8. In Step 9, a solution ($\mathbf{s}$) is chosen using Algo. 2. The same solution is then chosen as the global leader for the particles which belong to the cluster of the reference line j. It is then removed from the cluster in Step 11.

Algorithm 7 Assign_GB

1: % Stage-1 of the diversity preference approach: Clusters of reference lines having at least one solution
2: **for** cluster ($C_i \neq \emptyset$) of every reference line i **do**
3: Select solution ($\mathbf{s}$) using Algo. 1;
4: Assign solution ($\mathbf{s}$) as the global leader to all particles that belong to the cluster of the reference line i;
5: Remove solution ($\mathbf{s}$) from C_i, i.e., $C_i \setminus \mathbf{s}$;
6: **end for**
7: % Stage-2 of the diversity preference approach: Clusters of reference lines having no solution
8: **for** cluster ($C_j = \emptyset$) of every reference line j **do**
9: Select solution ($\mathbf{s}$) using Algo. 2;
10: Assign solution ($\mathbf{s}$) as the global leader to all particles that belong to the cluster of sthe reference line j;
11: Remove solution ($\mathbf{s}$) from C_i, i.e., $C_i \setminus \mathbf{s}$;
12: **end for**

3.5. Velocity and Position Update of particles in Steps 7 and 8 of Algo. 3

Equation (2) is mostly used for updating the velocity of each particle. For the proposed algorithm, some modifications are made with the parameters. For example, the weight (w) changes with generations as $w = w_o(1 - t/T) + 0.05$. Similarly, the coefficients c_1 and c_2 also change with generation as $c_1, c_2 = c_o(1 - t/T) + 0.1$. An adaptive parameter δ is kept which also changes with generations as $\delta = 0.5(1 - t/T) + 0.1$. When a random number between $[0, 1]$ is generated and found less than δ, the velocity is updated as given in Equation (2). Otherwise, the velocity of a particle (i) is updated as

$$\mathbf{v}_i(t+1) = (GB_i(t) - \mathbf{x}_i(t)). \tag{10}$$

Thereafter, the position of each particle is updated as given in Equation (3). In case any variable is jumped over its bounds, it is brought back to the nearest bound.

*3.6. Evolutionary Search on $GB(t)$: **Evolutionary_Search**($GB(t)$) in Step 9 of Algo. 3*

The evolutionary search is performed so that leaders in the archive of $GB(t)$ should not be stuck at the local optima, and their improved positions steer the search of swarm toward the PO front. The major steps of **Evolutionary_Search** are presented in Algo. 8 in which crossover and mutation are performed on the archive of $GB(t)$ in Step 1. Since this evolutionary search generates another set of leaders, the $GB(t)$ archive is updated by combining the new and existing leaders in Step 2. The steps for updating the archive of $GB(t)$ are presented in Section 3.8.

Algorithm 8 Evolutionary_Search($GB(t)$)

1: $GB(t)_{new}$ = Perform crossover followed by mutation on $GB(t)$;%by using SBX crossover operator and Polynomial mutation operator
2: Update the archive of $GB(t)$: **Update_GB**($GB(t)_{new} \cup GB(t)$);

3.7. Update $LB(t)$ Archive: **Update_LB** *in Step 11 of Algorithm 3*

The archive of $LB(t)$ is updated using the PBI fitness, which is already calculated in Step 10 of Algo. 3. If the PBI fitness of the particle in the swarm ($P(t)$) is smaller than its local leader, the local leader is updated for the particle.

3.8. Update $GB(t)$ Archive: **Update_GB** *in Step 12 of Algorithm 3*

An update of the $GB(t)$ archive is developed using the diversity preference approach discussed in Section 3.1. The major steps are shown in Algo. 9. In the beginning, the archive of $GB(t)$ is kept empty. In Step 3, solutions are selected using the stage-1 of the diversity preference approach. A solution ($\mathbf{s}$) with a minimum PBI fitness from each cluster ($C_i \neq \emptyset$) is chosen and copied it in the archive of $GB(t)$ in Step 5. This ensures the selection of a diverse set of solutions in the archive of $GB(t)$ through the set of the reference lines. The chosen solution ($\mathbf{s}$) is then removed from its respective cluster in Step 6. Since some clusters ($C_j = \emptyset$) of the reference lines can be empty, the stage-2 of the diversity preference approach is used in Step 8. In this case, the PBI fitness is calculated among the remaining solutions of the non-empty clusters ($C_i \neq \emptyset$) and the reference line j. Solution ($\mathbf{s}$) with minimum PBI fitness with respect to the reference line j is chosen and copied it to the archive of $GB(t)$ in Step 10. Finally, the solution ($\mathbf{s}$) is removed from its respective cluster for further consideration in Step 11.

The above procedure ensures that we can copy $|H|$ number of solutions to the archive of $GB(t)$. Since the archive size can be $N > |H|$, we need to select a few more solutions for filling the archive of $GB(t)$. Therefore, a non-empty cluster ($C_i \neq \emptyset$) is chosen randomly, and solution ($\mathbf{s}$) with a minimum PBI fitness is selected in Step 16. The same solution is then copied to the $GB(t)$ archive in Step 17 and gets removed from its respective cluster in Step 18. This procedure is repeated until the archive of $GB(t)$ is completely filled.

3.9. Computational Complexity

The major computation is involved either with non-dominated sorting at Step 1 of Algo. 4, or calculating distance for making clusters in Step 2 of Algo. 6. The non-dominated sorting of $2N$ population size and M objectives is $O(N^2 log^{M-2} N)$ [41]. The clustering operation with respect to H number of reference points for $2N$ population size with M objective requires $O(NMH)$ computation. In the present implementation, we consider $H \approx N$ or $N > H$ that suggests the worst complexity of clustering operation as $O(N^2M)$. Therefore, the worst order complexity of the proposed MaOPSO is either $O(N^2 log^{M-2} N)$, or $O(N^2M)$ in one generation.

Algorithm 9 Update_GB$(R(t))$

% In Step 10 of Algo. 3, solutions of $R(t)$ are ranked and normalized. Clusters of solutions are also made.
Initialize $GB(t) = \emptyset$;
% Stage-1 of the diversity preference approach
for every cluster $C_i \neq \emptyset$ **do**
  Select one solution ($\mathbf{s}$) using Algo. 1 and copy it in $GB(t)$, i.e., $GB(t) = GB(t) \cup \mathbf{s}$;
  Remove solution ($\mathbf{s}$) from the cluster, i.e., $C_i = C_i \setminus \mathbf{s}$;
end for
% Stage-2 of the diversity preference approach
for every cluster $C_j = \emptyset$ **do**
  Select one solution ($\mathbf{s}$) using Algo. 2 and copy it in $GB(t)$, i.e., $GB(t) = GB(t) \cup \mathbf{s}$;
  Remove solution ($\mathbf{s}$) from the cluster, i.e., $C_i = C_i \setminus \mathbf{s}$;
end for
% When $GB(t)$ is not filled completely.
if $|H| < N$ **then**
  while $|GB(t)| < N$ **do**
    Choose a cluster ($C_k : C_k \neq \emptyset$) randomly and select the minimum PBI fitness solution ($\mathbf{s}$) from the remaining solutions in C_k;
    Copy solution ($\mathbf{s}$) into $GB(t)$, i.e., $GB(t) = GB(t) \cup \mathbf{s}$;
    Remove solution ($\mathbf{s}$) from the cluster, i.e., $C_k = C_k \setminus \mathbf{s}$;
  end while
end if

4. Results and Discussion

The proposed MaOPSO, which is developed using the diversity preference approach, is now referred to as MaOPSO-DP. It is tested using DTLZ [42] and WFG [43] problems, and its outcome is compared with the existing many-objective evolutionary and PSO algorithms. First, the details of the simulation are given, followed by the comparison of results.

4.1. Simulation Details

Test Instances. Various test instances of 3-, 5-, 8-, 10-, and 15-objective of DTLZ and WFG problems are chosen for generating and comparing the outcome of MaOPSO-DP. The number of decision variables for DTLZ problems is given as $n = M + k - 1$. Here, $k = 5$ for DTLZ1, and $k = 10$ for DTLZ2-4 problems. The number of decision variables for WFG1-9 problems is set to $n = k + l$. Here, k is the position-related variable, which is kept as $k = 2 \times (M - 1)$, and l is the distance-related variable, which is kept constant as $l = 20$. It can be seen from the literature that these problems have different characteristics that make them difficult to solve by various algorithms [37].

Algorithms. A set of algorithms is chosen for comparison. The list includes eight many-objective evolutionary and PSO algorithms, such as NSGA-III [1], LEAF [37], $\theta-$DEA [19], MEAD/D-URAW [44], ISDE+ [45], MaPSO [36], NMPSO [31], and MPSO/D [1] [22]. Algorithms like NSGA-III, LEAF, $\theta-$DEA, and MaPOS use the similar reference-lines-based framework. Algorithm $\theta-$DEA uses an approach similar to the diversity preference approach in which the diversity is preserved first through the reference lines, and dominance is targeted later through a new dominance rule called $\theta-$sorting. MaPSO, NMPSO, and MPSO/D are recently published and efficient many-objective PSO algorithms.

Statistical Performance Assessment. Since all algorithms are meta-heuristic, we run them 20 times with different initial population. Their performance is assessed through the inverse generalized distance (IGD) and hypervolume (HV) indicators. IGD indicator is calculated as,

$$IGD(\mathbf{Q}, \mathbf{P}^*) = \frac{\sum_{i=1}^{|P^*|} \min_{j=1}^{|Q|} d(p_i, q_j)}{|P^*|}, \tag{11}$$

where P^* is the set of PO solutions, Q is the set of obtained non-dominated solutions, $|P^*|$ is the cardinality of P^*, $|Q|$ is the cardinality of Q, and $d(p_i, q_j) = ||p_i - q_j||^2$. Since a set of the PO solution (P^*) is required for the IGD indicator, the set is found using the reference lines generated in Section 3.2. For example, we know the PO front of the DTLZ1 problem is a hyperplane, that is, $\sum_{i=1}^{M} f_i = 0.5, \forall f_i \geq 0$. The points of intersection between the reference lines and the PO front generate a set of PO solutions. Similarly, we find the set of the PO solutions for other DTLZ problems, that is, the point of intersection between the

[1]The source codes of LEAF, θ-DEA, ISDE+, and MaPSO are provided by the authors with a link in their papers. PlatEMO [46] is used for MEAD/D-URAW, NMPSO, and MPSO/D algorithms. The source code of NNSGA-III provided by the authors of LEAF is used for NSGA-III results. The source code of MaOPSO-DP is available at https://www.iitg.ac.in/dsharma/index.html.

reference lines and the PO front, $\sum_{i=1}^{M} f_i^2 = 1, \forall f_i \geq 0$. We find the PO solutions for WFG4-9 problems similar to DTLZ2-4 problems. Moreover, the PO solutions of WFG1-3 problems are also found by following the same procedure.

HV indicator measures the size of the objective space dominated by the solutions in Q and bounded by $\mathbf{z}^r$, which is calculated as

$$HV(Q) = VOL\left(\bigcup_{\mathbf{x}\in\Omega} [f_1((x), z_i^r] \times \ldots [f_M((x), z_M^r]\right), \tag{12}$$

where $VOL(.)$ indicates the Lebesgue measure, and $\mathbf{z}^r = (z_1^r, \ldots, z_M^r)^T$ is the reference point in the objective space that is dominated by all Pareto-optimal solutions. The large is the HV value, the better is the quality of Q for approximating the PO front [47]. For DTLZ1, $\mathbf{z}^r = (1, \ldots, 1)^T$ is chosen. For other DTLZ and WFG problems, $\mathbf{z}^r = (2, \ldots, 2)^T$ is considered[2]. HV values are normalized between $[0, 1]$ by dividing $z = \prod_{i=1}^{M} z_i^r$. Both the indicators are determined by normalizing Q, except for DTLZ1.

The Wilcoxon signed-rank test at 5% significance level is also used to determine the difference for statistical significance between MaOPSO-DP and the benchmark algorithm.

Parameter Setting. We set the population size based on the reference points generated using Equation (7) [1]. The details of no. of division for each objective instance, number of reference points, and population size for algorithms are given in Table 1. It is noted that the two-layered approach defined by [1] is used for a higher number of objectives.

Table 1: Population sizes for all algorithms.

No. of obj. (M)	divisions p or (p_1, p_2)	No. of ref. points (H)	Population (N)
3	12	91	92
5	6	210	210
8	$(3, 2)$	156	156
10	$(3, 2)$	275	276
15	$(2, 1)$	135	136

All algorithms terminate using the maximum number of allowed generations. Table 2 presents the common termination conditions for all algorithms.

The algorithm-specific parameters are listed in Table 3. These parameters are chosen based on the general parameter settings mentioned in the original studies of algorithms for solving DTLZ and WFG problems. NSGA-III, LEAF, $\theta-$DEA, ISDE+, and MaOPSO-DP use SBX crossover operator [39] for which the probability is set to 1.0 and crossover index (η_c) is set to 30. NSGA-III, LEAF, $\theta-$DEA, ISDE+, NMPSO, MPSO/D, and MaOPSO-DP use polynomial mutation operator [39] for which the probability is set to $(1/n)$ and mutation

[2]The source of WFG group is used for HV computation (https://github.com/lbradstreet/WFG-hypervolume).

Table 2: Maximum number of generations for terminating all algorithms.

No. of objectives	DTLZ1	DTLZ2	DTLZ3	DTLZ4	WFG (all)
3	400	250	1000	600	1000
5	600	350	1000	1000	1250
8	750	500	1000	1250	1500
10	1000	750	1500	2000	2000
15	1500	1000	2000	3000	3000

index (η_m) is set to 20. As suggested in the original study, the parameter θ is set to 5 for $\theta-$DEA [19]. Similarly, the number of closest vectors $T = N/10$, and other parameters such as $\delta = 0.9,\ nr = 2$ are set for MOEA/D-URAW [44]. According to [36], the parameters $K = 3$, $\theta_{max} = 0.5$, $w = 0.1$, and $C = 2.0$ are set for MaPSO. For NMPSO [31], the factor w is chosen randomly between $[0.1, 0.5]$, and the coefficients (c_1, c_2, c_3) lie between $[1.5, 2.5]$. According to MPSO/D [22], the factor w is chosen randomly between $[0.1, 0.9]$, and the coefficients (c_1, c_2) are set to 2. For MaOPSO-DP, the initial values of $w_o = 0.9$ and $c_o = 2.5$ are set that reduce with the number of generations.

Table 3: Parameter settings of all algorithms.

Algorithms	Parameters
NSGA-III	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$ $\eta_m = 20$
LEAF	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$ $\eta_m = 20$
θ-DEA	$\theta = 5.0$ for PBI method
MEAD/D-URAW	$T = N/10,\ \delta = 0.9,\ nr = 2$
ISDE+	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$ $\eta_m = 20$
MaPSO	$K = 3$, $\theta_{max} = 0.5$, $w = 0.1$, $C = 2.0$
NMPSO	$w \in [0.1, 0.5]$, $c_1, c_2, c_3 \in [1.5, 2.5]$, $p_m = 1/n$, $\eta_m = 20$
MPSO/D	$c_1, c_2 = 2$, $w \in [0.1, 0.9]$, $p_m = 1/n$, $\eta_m = 20$, no. of weight vectors in the neighborhood is 20
MaOPSO-DP	$w_o = 0.9$, $c_o = 2.5$, $p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$ $\eta_m = 20$

4.2. Performance Assessment on DTLZ problems

Table 4 presents the best, median, and worst IGD values obtained from the results of algorithms on DTLZ problem instances. The gray shaded cells represent the best (smaller value) IGD value in the corresponding row of the problem instances. DTLZ1 problem is a multi-modal problem with a linear PO front that is intersecting the objective axes at 0.5 coordinate. While solving this problem, MaOPSO-DP shows the best IGD value for all instances, except for the 15-objective instance. DTLZ2 problem is one of the simplest problems that has a convex PO front. MaOPSO-DP shows the best IGD values for all instances of objectives. DTLZ3 problem has the same PO front as DTLZ2 problem, but it is a multi-modal problem. Although DTLZ3 problem is a relatively difficult problem, MaOPSO-DP again shows the best IGD values for all objective instances. The last problem from DTLZ problem set considered for performance comparison is DTLZ4 problem that has a concave PO font similar to DTLZ2 problem, but it is a biased problem. MaOPSO-DP shows the best IGD value for all instances.

It can be seen from Table 4 that MaOPSO-DP shows the best IGD values in 54 rows out of 60. The Wilcoxon test results are shown in the same table with symbols $(+, =, -)$. The symbol '+' suggests a significantly better performance of MaOPSO-DP over the corresponding benchmark algorithm. Other symbols '$-$' and '=' suggest significantly bad performance and equivalent performance of MaOPSO-DP over the corresponding algorithm, respectively. For DTLZ1 problem, MaOPSO-DP shows significantly better performance over eight algorithms for 3-, 5-, 8- and 10-objective instances. NSGA-III, LEAF, ISDE+, and MaPSO show significantly better performance than MaOPSO-DP on 15-objective instance. MOEA/D-URAW shows equivalent performance on the same objective instance against MaOPSO-DP. For DTLZ2 problem, MaOPSO-DP shows significantly better performance over eight algorithms on all objective instances. For DTLZ3 problem, MaOPSO-DP shows significantly better performance over eight algorithms for 3-, 5-, 8- and 10-objective instances. For 15-objective instance, MaOPSO-DP shows significantly better performance over MOEA/D-URAW, ISDE+, MaPSO, and NMPSO, and equivalent performance with NSGA-III, LEAF, $\theta-$DEA, and MPSO/D. For DTLZ4 problem, MaOPSO-DP shows again a significantly better performance over eight algorithms over eight algorithms. At the bottom of the table, a collective outcome of the Wilcoxon test for every algorithm with respect to MaOPSO-DP is given. It can be seen that MaOPSO-DP outperforms the chosen set of algorithms on DTLZ problem instances.

Table 5 presents the best, median, and worst HV values obtained from the results of algorithms on DTLZ problem instances. Again, the gray cells represent the best (larger value) HV value for the corresponding row. For DTLZ1 problem, MaOPSO-DP shows the best HV values in all objective instances than the chosen set of eight algorithms. For DTLZ2 problem, MaOPSO-DP shows the best HV values for 3- and 5-objective instances. ISDE+ shows the best HV values for the remaining instances of DTLZ2 problem. For DTLZ3 problem, MaOPSO-DP shows the best HV values for all instances over other algorithms. For DTLZ4 problem, $\theta-$DEA, MOEA/D-URAW, and NMPSO show the best HV values for 3-objective instance. NMPSO and MPSO/D show the best HV values for the remaining objective instance. As a cumulative performance, MaOPSO-DP shows the best HV value in 27 rows out of 48, which is the highest number among other algorithms.

The outcome of the Wilcoxon test is also presented at the bottom of Table 5. MaOPSO-DP shows significantly better performance in 12 instances, equivalent performance in 3 instances, and significantly worst performance in 1 instance than NSGA-III and $\theta-$DEA for all DTLZ problem instances. Similarly, MaOPSO-DP shows significantly better performance in 13 instances, equivalent performance in 2 instances and significantly worst performance in 1 instance than LEAF and ISDE+. MaOPSO-DP seems to be quite better than MOEA/D-URAW and MaPSO while showing significantly better performance in almost all instances. MaOPSO-DP shows significantly better performance in 11 and 13 instances than NMPSO and MPSO/D, respectively. It can be seen that MaOPSO-DP outperforms the benchmark algorithms on DTLZ problem instances using HV indicator values. The results based on the IGD and HV indicators suggest that the proposed diversity preference approach enhances the performance of many-objective PSO by selecting isolated leaders for the GB archive.

The obtained non-dominated solutions after the termination of algorithms are presented in Fig. 3 for 3-objective DTLZ3 problem, which is a concave and multi-modal multi-objective optimization problem. The solutions are drawn with respect to the run of the median IGD value of each algorithm. It can be seen that all algorithms converge to the PO front. Except for ISDE+, MaPSO, and NMPSO, all algorithms generate a diverse set of solutions on the PO front. Fig. 4 shows the value path obtained from the non-dominated solutions of algorithms for 10-objective DTLZ3 problem. Again, all algorithms generate the non-dominated solutions in the range of all objectives, except for MaPSO and NMPSO.

4.3. Performance Assessment on WFG Problems

Table 6 presents the best, median, and worst IGD values obtained from the algorithms for various instances of WFG problems. The gray cells represent the best IGD values for each row. WFG1 problem is a mixed and biased problem, and MOEA/D-URAW shows the best IGD values for all instances. MOEA/D-URAW shows one decimal place of improvement in IGD values for 3- and 5-objective instances over MaOPSO-DP. WFG2 is a convex, disconnected, multi-modal, and non-separable problem on which LEAF shows the best IGD values for 3- and 5-objective instances. For 8-objective WFG2 instance, LEAF shows the minimum IGD value, MaOPSO-DP shows the best median IGD value and MOEA/D-URAW shows the improved worst IGD value. MOEA/D-URAW and MaPSO show the best IGD values for 10- and 15-objective instances, respectively. WFG3 problem is a linear, degenerate, and non-separable problem, and NMPSO shows the best IGD values for 3-, 8-, 10- and 15-objective instances. LEAF shows the best IGD values for 5-objective instances. WFG4 problem is a concave and multi-modal problem, and $\theta-$DEA shows the best IGD values for 3- and 5-objective instances. MaOPSO-DP shows the best IGD values for 8-objective instance. LEAF and NMPSO show the best IGD values for 10- and 15-objective instances, respectively. WFG5 problem is a concave and deceptive problem, and MaOPSO-DP shows the best IGD values for 3-, 8- and 10-objective instances. LEAF shows the best IGD values for 5- and 15-objective instances. WFG6 problem is a concave and non-separable problem, and the best IGD values for 3-objective instance are shown by MaOPSO-DP. LEAF shows the best IGD values for 5-, 10- and 15-objective instances. $\theta-$DEA shows the best IGD values for 8-objective instance. WFG7 is a concave and biased problem, and the best IGD values for all instances are shown by MaOPSO-DP. WFG8 is a concave, biased, and non-separable problem, and the best IGD values for 3-, 5-, 8- and 10-objective instances are shown by MaOPSO-DP. NMPSO shows the best IGD values for 15-objective instance. WFG9 is a concave, biased, multi-modal, deceptive, and non-separable problem, and the minimum IGD value for 3-objective instance is shown by MaOPSO. LEAF shows the best IGD values for other objective instances. Overall, MaOPSO-DP shows the best IGD values in 43 rows out of 135, which is the highest number, followed by LEAF in 37 rows. MaOPSO-DP is found the best in solving 8- and 15-objective WFG1, 3-, 8- and 10-objective WFG5, 3- objective WFG6, 3-, 5-, 8- and 10-objective WFG7, 3-, 5-, 8- and 10-objective WFG8, 3- and 5-objective WFG9 instances.

The outcome of the Wilcoxon test can be seen at the bottom of Table 6. MaOPSO-DP shows significantly better performance on many instances of WFG problems set over NSGA-III, ISDE+, and MPSO/D. MaOPSO-DP shows a significantly better performance on 32 and 35 instances over MaPSO and NMPSO, respectively. Similarly, MaOPSO-DP shows a significantly better performance on 25 and 27 instances over $\theta-$DEA and MOEA/D-URAW, respectively. MaOPSO-DP and LEAF seem to have an equivalent performance using the Wilcoxon test because both the algorithms show significantly better performance in 16 instances and equivalent performance in 13 instances.

Table 7 presents the best, median, and worst HV values obtained from the algorithms. The gray cells signify the better HV values with respect to the other algorithms. MOEA/D-URAW seem to be the dominating algorithm that shows better HV values for all objective instances of WFG1 and WFG4 problems, 5-objective WFG2 and WFG3 problems, 8- and 10-objective instances of WFG5 problem, 5-, 8- and 10-objective instances of WFG6 and WFG8 problems, 3-objective instance of WFG7 problem and 3- and 5-objective instance of WFG9 problem. MaPSO shows the best HV values for 8- and 10-objective instances of WFG2 and WFG3 problems, and 5-, 8- and 10-objective instances of WFG7 problem. MaOPSO-DP shows the best HV values for 3-objective WFG2 and WFG6 problems only. MOEA/D-URAW and MaPSO outperform MaOPSO-DP based on the statistical HV values.

The outcome of the Wilcoxon test is presented at the bottom of Table 7. MaOPSO-DP shows significantly better performance in almost all instances of WFG problems over ISDE+ and MPSO/D. MaOPSO-DP shows significantly better performance in 20, 17, 19, and 18 instances over NSGA-III, LEAF, $\theta-$DEA and MaPSO, respectively. MOEA/D-URAW, and NMPSO show significantly better performance in 29 and 22 instances over MaOPSO-DP.

Fig. 5 shows the obtained non-dominated solutions corresponding to the run of the median IGD value of each algorithm for 3-objective WFG7 problem. All algorithms are converged to the PO front, except for ISDE+. Among the converged algorithms, NMPSO and MPSO/D do not show a well-distributed set of solutions on the PO front. Fig. 6 shows value path plots for 10-objective WFG7 problem. Algorithms like MaOPSO-DP, NSGA-III, LEAF, and $\theta-$DEA show a similar distribution of solutions in the range of objectives. MOEA/D-URAW, ISDE+, MaPSO, and NMPSO show dense lines in the same figure. MPSO/D is failed to distribute solutions in the range of objectives.

The performance of MaOPSO-DP is also tested using IGD$^+$ indicator [48] on DTLZ and WFG problems. The results and discussion can be found in the supplementary sheet.

4.4. Average Performance Score

It can be observed from Sections 4.2 and 4.3, none of the algorithms is the clear winner on all instances of DTLZ and WFG problems. Therefore, an average performance score of each algorithm is calculated using IGD values, which is similar to the study presented by [19]. Fig. 7 shows an average performance score of every algorithm for different objective instances of DTLZ and WFG problems. It can be seen that MaOPSO-DP is the best for 3-, 5-, 8- and 10-objective instances, and the second-best for 15-objective instance. Still, MaOPSO-DP needs to improve for a higher number of objectives, that is, 15-objective. Fig. 8 shows an average performance score of every algorithm on each of DTLZ and WFG problems. MaOPSO-DP is found the best in solving DTLZ1-4, WFG5 and WFG7 problems. It is the second best in solving WFG1 and WFG2 problems. However, MaOPSO-DP needs to improve solving WFG3, WFG4, WFG6 and WFG9 problems. Fig. 9 shows an average rank of each algorithm over all instances of DTLZ and WFG problems. It can be seen that MaOPSO-DP is ranked the best among the chosen set of algorithms for solving box-constrained many-objective optimization problems.

5. Conclusions

The diversity preference approach was developed using the reference-lines-based framework, which was coupled with many-objective PSO. The main objective was to select isolated solutions to maintain diversity among them. In this process, some dominated solutions can also be selected. The proposed approach was used to update leaders of the $GB(t)$ archive and to assign a leader from the same archive to each particle of the swarm. MaOPSO-DP was tested on DTLZ and WFG problem instances, and its outcome was compared with eight many-objective evolutionary and PSO algorithms. On DTLZ problems, MaOPSO-DP outperformed all algorithms using both the indicators. It is because of the diversity preference approach. On WFG problems, MaOPSO-DP was the best among the chosen set of algorithms using the IGD indicator. However, two benchmark algorithms were found better than MaOPSO-DP using the HV indicator. An interesting outcome was observed when comparing the obtained non-dominated solutions of MaOPSO-DP, NSGA-III, LEAF, and θ-DEA that the distribution of solutions on the PO front was very similar to instances of DTLZ3 and WFG7 problems. Nevertheless, MaOPSO-DP needs attention for solving higher-objective problems. The main reason for not so good performance on those problem instances is frequent jumping of particles out of the variable bounds. In future work, MaOPSO-DP can be improved further by adapting an effective velocity update so that frequent jumping of particles can be reduced. Moreover, the strategy to generate structure reference points on the unit hyperplane needs attention so that MaOPSO-DP can solve problems effectively that are having inverted or baldy scaled PO fronts, such as with MaF problems set [49]. MaOPSO-DP can be further extended for constraint many-objective optimization problems and real-world problems.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints, IEEE Transactions on Evolutionary Computation 18 (4) (2014) 577–601. doi:10.1109/TEVC.2013.2281535.

[2] Q. Zhang, H. Li, Moea/d: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731. doi:10.1109/TEVC.2007.892759.

[3] J. Bader, E. Zitzler, Hype: An algorithm for fast hypervolume-based many-objective optimization, Evolutionary Computation 19 (1) (2011) 45–76. doi:10.1162/EVCO_a_00009.

[4] N. Barakat, D. Sharma, Evolutionary multi-objective optimization for bulldozer and its blade in soil cutting, International Journal of Management Science and Engineering Management 14 (2) (2019) 102–112. doi:10.1080/17509653.2018.1500953.

[5] D. Sharma, K. Deb, N. N. Kishore, Domain-specific initial population strategy for compliant mechanisms using customized genetic algorithm, Structural and Multidisciplinary Optimization 43 (4) (2011) 541–554.

[6] D. Sharma, K. Deb, N. N. Kishore, Customized evolutionary optimization procedure for generating minimum weight compliant mechanisms, Engineering Optimization 46 (1) (2014) 39–60.

[7] L. Ram, D. Sharma, Evolutionary and gpu computing for topology optimization of structures, Swarm and Evolutionary Computation 35 (2017) 1 – 13. doi:https://doi.org/10.1016/j.swevo.2016.08.004.

[8] T. Ray, K. Tai, K. C. Seow, Multiobejctive design optimization by an evolutionary algorithm, Engineering Optimization 33 (4) (2001) 399–424.

[9] R. C. Purshouse, P. J. Fleming, On the evolutionary optimization of many conflicting objectives, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 770–784. doi:10.1109/TEVC.2007.910138.

[10] Z. He, G. G. Yen, J. Zhang, Fuzzy-based pareto optimality for many-objective evolutionary algorithms, IEEE Transactions on Evolutionary Computation 18 (2) (2014) 269–285. doi:10.1109/TEVC.2013.2258025.

[11] X. Zou, Y. Chen, M. Liu, L. Kang, A new evolutionary algorithm for solving many-objective optimization problems, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 38 (5) (2008) 1402–1412. doi:10.1109/TSMCB.2008.926329.

[12] C. Dai, Y. Wang, L. Hu, An improved α-dominance strategy for many-objective optimization problems, Soft Computing 20 (3) (2016) 1105–1111. doi:10.1007/s00500-014-1570-8.

[13] K. Li, K. Deb, Q. Zhang, S. Kwong, An evolutionary many-objective optimization algorithm based on dominance and decomposition, IEEE Transactions on Evolutionary Computation 19 (5) (2015) 694–716. doi:10.1109/TEVC.2014.2373386.

[14] Y. Xiang, Y. Zhou, M. Li, Z. Chen, A vector angle-based evolutionary algorithm for unconstrained many-objective optimization, IEEE Transactions on Evolutionary Computation 21 (1) (2017) 131–152. doi:10.1109/TEVC.2016.2587808.

[15] E. Figueiredo, T. Ludermir, C. Bastos-Filho, Many objective particle swarm optimization, Information Sciences 374 (2016) 115 – 134. doi:https://doi.org/10.1016/j.ins.2016.09.026.

[16] A. Trivedi, D. Srinivasan, K. Sanyal, A. Ghosh, A survey of multiobjective evolutionary algorithms based on decomposition, IEEE Transactions on Evolutionary Computation 21 (3) (2017) 440–462. doi:10.1109/TEVC.2016.2608507.

[17] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, A reference vector guided evolutionary algorithm for many-objective optimization, IEEE Transactions on Evolutionary Computation 20 (5) (2016) 773–791.

[18] M. Asafuddoula, T. Ray, R. Sarker, A decomposition-based evolutionary algorithm for many objective optimization, IEEE Transactions on Evolutionary Computation 19 (3) (2015) 445–460. doi:10.1109/TEVC.2014.2339823.

[19] Y. Yuan, H. Xu, B. Wang, X. Yao, A new dominance relation-based evolutionary algorithm for many-objective optimization, IEEE Transactions on Evolutionary Computation 20 (1) (2016) 16–37. doi:10.1109/TEVC.2015.2420112.

[20] R. Hernández Gómez, C. A. Coello Coello, Improved metaheuristic based on the r2 indicator for many-objective optimization, in: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, GECCO '15, ACM, New York, NY, USA, 2015, pp. 679–686. doi:10.1145/2739480.2754776.

[21] J. Luo, X. Huang, Y. Yang, X. Li, Z. Wang, J. Feng, A many-objective particle swarm optimizer based on indicator and direction vectors for many-objective optimization, Information Sciences 514 (2020) 166 – 202. doi:https://doi.org/10.1016/j.ins.2019.11.047.

[22] C. Dai, Y. Wang, M. Ye, A new multi-objective particle swarm optimization algorithm based on decomposition, Information Sciences 325 (2015) 541 – 557. doi:https://doi.org/10.1016/j.ins.2015.07.018.

[23] D. Sharma, S. Z. Basha, S. A. Kumar, Diversity over dominance approach for many-objective optimization on reference-points-based framework, in: K. Deb, E. Goodman, C. A. Coello Coello, K. Klamroth, K. Miettinen, S. Mostaghim, P. Reed (Eds.), Evolutionary Multi-Criterion Optimization, Springer International Publishing, Cham, 2019, pp. 278–290.

[24] C. A. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 256–279. doi:10.1109/TEVC.2004.826067.

[25] S. Mostaghim, H. Schmeck, Distance Based Ranking in Many-Objective Particle Swarm Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 753–762. doi:10.1007/978-3-540-87700-4-75.

[26] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, E. Alba, Smpso: A new pso-based metaheuristic for multi-objective optimization, in: 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making(MCDM), 2009, pp. 66–73. doi:10.1109/MCDM.2009.4938830.

[27] A. Britto, A. Pozo, Using reference points to update the archive of mopso algorithms in many-objective optimization, Neurocomputing 127 (2014) 78 – 87, advances in Intelligent Systems. doi:https://doi.org/10.1016/j.neucom.2013.05.049.

[28] W. Hu, G. G. Yen, G. Luo, Many-objective particle swarm optimization using two-stage strategy and parallel cell coordinate system, IEEE Transactions on Cybernetics 47 (6) (2017) 1446–1459. doi:10.1109/TCYB.2016.2548239.

[29] Q. Zhu, Q. Lin, W. Chen, K. Wong, C. A. Coello Coello, J. Li, J. Chen, J. Zhang, An external archive-guided multiobjective particle swarm optimization algorithm, IEEE Transactions on Cybernetics 47 (9) (2017) 2794–2808. doi:10.1109/TCYB.2017.2710133.

[30] A. Pan, L. Wang, W. Guo, Q. Wu, A diversity enhanced multiobjective particle swarm optimization, Information Sciences 436-437 (2018) 441 – 465. doi:https://doi.org/10.1016/j.ins.2018.01.038.

[31] Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, C. A. C. Coello, K. Wong, J. Zhang, Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems, IEEE Transactions on Evolutionary Computation 22 (1) (2018) 32–46. doi:10.1109/TEVC.2016.2631279.

[32] X. Liu, Z. Zhan, Y. Gao, J. Zhang, S. Kwong, J. Zhang, Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization, IEEE Transactions on Evolutionary Computation 23 (4) (2019) 587–602.

[33] S. Qin, C. Sun, G. Zhang, X. He, Y. Tan, A modified particle swarm optimization based on decomposition with different ideal points for many-objective optimization problems, Complex & Intelligent Systems (in press) (2020) 1 – 12doi:https://doi.org/10.1007/s40747-020-00134-7.

[34] W. Yang, L. Chen, Y. Wang, M. Zhang, A reference points and intuitionistic fuzzy dominance based particle swarm algorithm for multi/many-objective optimization, Applied Intelligence 50 (2020) 1133 – 1154. doi:https://doi.org/10.1007/s10489-019-01569-3.

[35] L. Li, L. Chang, T. Gu, W. Sheng, W. Wang, On the norm of dominant difference for many-objective particle swarm optimization, IEEE Transactions on Cybernetics (in press) (2020) 1–13.

[36] Y. Xiang, Y. Zhou, Z. Chen, J. Zhang, A many-objective particle swarm optimizer with leaders selected from historical solutions by using scalar projections, IEEE Transactions on Cybernetics 50 (5) (2020) 2209–2222.

[37] D. Sharma, P. K. Shukla, Line-prioritized environmental selection and normalization scheme for many-objective optimization using reference-lines-based framework, Swarm and Evolutionary Computation 51 (2019) 100592. doi:https://doi.org/10.1016/j.swevo.2019.100592.

[38] I. Das, J. E. Dennis, Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems, SIAM Journal on Optimization 8 (3) (1998) 631–657. doi:10.1137/S1052623496307510.

[39] K. Deb, R. B. Agrawal, Simulated binary crossover for continuous search space, Complex Systems 9 (2) (1995) 115–148.

[40] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, Evolutionary Computation, IEEE Transactions on 6 (2) (2002) 182 –197.

[41] H. T. Kung, F. Luccio, F. P. Preparata, On finding the maxima of a set of vectors, Journal of the Association for Computing Machinery 22 (4).

[42] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: A. Abraham, L. Jain, R. Goldberg (Eds.), Evolutionary multiobjective optimization: theoritical advances and applications, Springer-Verlag, London, 2005, pp. 105–145.

[43] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Transactions on Evolutionary Computation 10 (5) (2006) 477–506. doi:10.1109/TEVC.2005.861417.

[44] L. R. C. Farias, A. F. R. Arajol, Many-objective evolutionary algorithm based on decomposition with random and adaptive weights, in: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), 2019, pp. 3746–3751.

[45] T. Pamulapati, R. Mallipeddi, P. N. Suganthan, i_{sde} +an indicator for multi and many-objective optimization, IEEE Transactions on Evolutionary Computation 23 (2) (2019) 346–352.

[46] Y. Tian, R. Cheng, X. Zhang, Y. Jin, PlatEMO: A MATLAB platform for evolutionary multi-objective optimization, IEEE Computational Intelligence Magazine 12 (4) (2017) 73–87.

[47] L. While, L. Bradstreet, L. Barone, A fast way of calculating exact hypervolumes, IEEE Transactions on Evolutionary Computation 16 (1) (2012) 86–95. doi:10.1109/TEVC.2010.2077298.

[48] H. Ishibuchi, H. Masuda, Y. Tanigaki, Y. Nojima, Modified distance calculation in generational distance and inverted generational distance, in: A. Gaspar-Cunha, C. Henggeler Antunes, C. C. Coello (Eds.), Evolutionary Multi-Criterion Optimization, Springer International Publishing, Cham, 2015, pp. 110–125.

[49] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, X. Yao, A benchmark test suite for evolutionary many-objective optimization, Complex & Intelligent Systems 3 (2017) 6781.

Table 4: Best, median and worst IGD values obtained by MaOPSO-DP and other algorithms on DTLZ instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III [1]	LEAF [37]	$\theta-$DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
DTLZ1	3	3.510E-04	5.286E-04	3.854E-04	1.756E-02	2.042E-02	2.135E-02	2.083E-02	2.763E-01	**5.531E-05**
		1.536E-03 +	1.249E-03+	1.178E-03 +	2.092E-02 +	2.251E-02+	2.331E-02+	2.381E-02 +	9.869E-01 +	**6.826E-05**
		5.787E-03	2.867E-03	4.201E-03	2.364E-02	2.396E-02	2.494E-02	2.789E-02	1.873E+00	**8.439E-05**
	5	4.962E-04	4.148E-04	3.931E-04	5.061E-02	5.101E-02	6.422E-02	5.962E-02	1.561E-01	**1.249E-04**
		7.431E-04 +	6.975E-04+	4.982E-04 +	5.401E-02 +	5.550E-02+	7.465E-02+	6.363E-02 +	2.731E-01 +	**1.549E-04**
		1.246E-03	1.170E-03	9.666E-04	5.657E-02	5.781E-02	1.069E-01	6.728E-02	4.682E-01	**1.870E-04**
	8	2.175E-03	2.030E-03	1.925E-03	1.047E-01	1.021E-01	1.178E-01	1.194E-01	3.014E-01	**1.021E-03**
		3.582E-03 +	2.638E-03+	2.260E-03 +	1.177E-01 +	1.054E-01+	1.370E-01+	1.776E-01 +	4.414E-01 +	**1.156E-03**
		6.645E-02	4.431E-03	1.519E-02	1.277E-01	1.109E-01	1.925E-01	2.133E-01	5.749E-01	**1.532E-03**
	10	2.279E-03	1.784E-03	2.089E-03	1.177E-01	1.118E-01	1.249E-01	1.608E-01	1.192E-01	**1.196E-03**
		2.583E-03 +	2.280E-03+	2.380E-03 +	1.262E-01 +	1.150E-01+	1.564E-01+	1.850E-01 +	1.997E-01 +	**1.356E-03**
		9.297E-02	4.756E-03	2.590E-03	1.403E-01	1.188E-01	1.969E-01	2.086E-01	2.617E-01	**1.582E-03**
	15	**1.922E-03**	2.015E-03	2.143E-01	2.201E-01	1.797E-01	1.736E-01	2.986E-01	1.879E-01	2.953E-03
		2.853E-03−	3.019E-03−	3.073E-01 +	2.418E-01 =	1.960E-01−	1.959E-01−	8.578E-01 +	2.822E-01 +	2.447E-01
		4.324E-03	4.356E-03	4.145E-01	2.622E-01	2.376E-01	2.163E-01	4.632E+01	3.527E-01	2.982E-01
DTLZ2	3	1.045E-03	1.029E-03	9.181E-04	5.069E-02	6.970E-02	4.903E-02	6.753E-02	7.596E-03	**4.890E-04**
		1.270E-03 +	1.308E-03+	1.058E-03 +	5.453E-02 +	7.507E-02+	5.543E-02+	7.492E-02 +	8.322E-03 +	**6.402E-04**
		2.870E-03	1.912E-03	1.263E-03	5.690E-02	8.053E-02	5.736E-02	8.036E-02	9.450E-03	**7.438E-04**
	5	3.058E-03	3.265E-03	2.072E-03	1.490E-01	1.634E-01	1.589E-01	1.634E-01	1.650E-02	**1.169E-03**
		4.481E-03 +	4.359E-03+	2.381E-03 +	1.548E-01 +	1.695E-01+	1.675E-01+	1.682E-01 +	1.780E-02 +	**1.440E-03**
		1.128E-02	6.989E-03	2.682E-03	1.626E-01	1.716E-01	1.891E-01	1.755E-01	1.926E-02	**1.707E-03**
	8	1.152E-02	1.065E-02	6.635E-03	2.960E-01	2.068E-01	3.167E-01	2.288E-01	2.653E-02	**4.100E-03**
		1.293E-02 +	1.326E-02+	7.407E-03 +	3.142E-01 +	2.273E-01+	3.352E-01+	2.402E-01 +	2.919E-02 +	**4.754E-03**
		1.691E-02	1.474E-02	9.067E-03	3.290E-01	2.410E-01	3.547E-01	2.652E-01	3.409E-02	**5.582E-03**
	10	1.142E-02	1.030E-02	6.683E-03	3.435E-01	2.452E-01	3.652E-01	2.806E-01	2.417E-02	**4.094E-03**
		1.279E-02 +	1.195E-02+	7.350E-03 +	3.595E-01 +	2.784E-01+	3.833E-01+	2.943E-01 +	2.602E-02 +	**4.778E-03**
		1.486E-02	2.006E-02	8.375E-03	3.702E-01	2.946E-01	3.945E-01	3.204E-01	2.748E-02	**5.415E-03**
	15	1.052E-02	1.055E-02	9.163E-03	4.907E-01	1.723E-01	4.247E-01	3.823E-01	3.115E-02	**4.367E-03**
		1.428E-02 +	1.207E-02+	1.042E-02 +	5.357E-01 +	2.276E-01+	4.592E-01+	6.317E-01 +	3.520E-02 +	**5.528E-03**
		1.758E-02	1.731E-02	2.257E-02	6.146E-01	3.025E-01	4.797E-01	8.084E-01	3.999E-02	**7.294E-03**
DTLZ3	3	8.723E-04	1.600E-03	9.103E-04	4.990E-02	6.901E-02	4.860E-02	6.978E-02	7.308E-03	**2.383E-04**
		3.991E-03 +	3.049E-03+	3.217E-03 +	5.470E-02 +	7.500E-02+	5.646E-02+	7.473E-02 +	1.277E-02 +	**2.891E-04**
		9.847E-03	9.425E-03	4.047E-02	5.743E-02	7.991E-02	2.002E+00	8.213E-02	1.428E-01	**3.479E-04**
	5	2.174E-03	1.447E-03	1.590E-03	1.532E-01	1.620E-01	1.718E-01	1.599E-01	2.282E-02	**3.720E-04**
		3.675E-03 +	3.637E-03+	2.500E-03 +	1.608E-01 +	1.724E-01+	2.153E-01+	1.684E-01 +	3.265E-02 +	**4.195E-04**
		1.014E-02	7.537E-03	7.298E-03	1.689E-01	1.781E-01	3.813E-01	1.759E-01	4.821E-02	**5.082E-04**
	8	1.256E-02	1.405E-02	7.475E-03	3.182E-01	2.401E-01	3.762E-01	2.325E-01	1.216E-01	**2.078E-03**
		2.444E-02 +	2.236E-02+	1.407E-02 +	3.423E-01 +	2.679E-01+	4.392E-01+	3.753E-01 +	3.942E-01 +	**2.479E-03**
		5.287E-02	5.302E-02	1.848E-01	4.268E-01	3.264E-01	5.171E-01	9.081E-01	9.314E-01	**3.345E-03**
	10	8.236E-03	7.087E-03	5.746E-03	3.625E-01	2.802E-01	4.699E-01	2.983E-01	4.242E-02	**2.261E-03**
		1.069E-02 +	9.583E-03+	6.543E-03 +	3.880E-01 +	3.216E-01+	5.020E-01+	5.438E-01 +	5.420E-02 +	**2.430E-03**
		1.929E-02	1.601E-02	8.574E-03	4.373E-01	3.447E-01	5.985E-01	1.265E+00	6.299E-02	**2.999E-03**
	15	1.121E-02	9.958E-03	7.383E-03	6.121E-01	2.285E-01	5.234E-01	7.200E-01	7.685E-02	**2.688E-03**
		1.766E-02 =	1.387E-02=	1.155E-02 =	7.083E-01 +	3.161E-01+	6.156E-01+	1.323E+00+	1.287E-01 =	**3.540E-03**
		3.671E-02	**2.184E-02**	6.418E-02	8.082E-01	6.105E-01	7.536E-01	1.326E+00	2.863E-01	6.633E-01
DTLZ4	3	3.113E-04	**2.668E-04**	3.079E-04	4.736E-02	7.061E-02	5.124E-02	6.782E-02	4.166E-03	2.988E-04
		3.918E-04 +	3.907E-04+	3.639E-04 =	5.176E-02 +	7.670E-02+	5.555E-02+	7.626E-02 +	4.744E-03 +	**3.576E-04**
		5.314E-01	5.306E-01	4.493E-04	5.313E-01	8.172E-02	5.766E-02	5.325E-01	5.192E-03	**4.394E-04**
	5	3.641E-04	3.641E-04	3.397E-04	1.348E-01	1.666E-01	1.463E-01	1.607E-01	8.703E-03	**3.304E-04**
		4.334E-04 +	4.391E-04+	4.093E-04 =	1.414E-01 +	1.726E-01+	1.533E-01+	1.688E-01 +	9.675E-03 +	**3.924E-04**
		5.072E-04	6.257E-04	4.909E-04	1.484E-01	1.798E-01	1.578E-01	1.750E-01	1.047E-02	**4.322E-04**
	8	2.541E-03	2.963E-03	2.262E-03	2.785E-01	2.467E-01	2.532E-01	2.130E-01	4.299E-02	**1.873E-03**
		3.442E-03 +	3.434E-03+	3.061E-03 +	2.945E-01 +	2.645E-01+	2.728E-01+	2.409E-01 +	4.820E-02 +	**2.678E-03**
		5.319E-03	4.451E-03	6.968E-01	3.099E-01	3.688E-01	2.862E-01	6.180E-01	5.154E-02	**3.113E-03**
	10	3.578E-03	3.333E-03	2.844E-03	3.267E-01	2.863E-01	2.813E-01	2.496E-01	3.844E-02	**2.546E-03**
		4.228E-03 +	3.951E-03+	3.355E-03 +	3.362E-01 +	3.062E-01+	2.990E-01+	2.707E-01 +	4.095E-02 +	**2.885E-03**
		5.174E-03	4.637E-03	4.182E-03	3.491E-01	3.364E-01	3.047E-01	3.111E-01	4.336E-02	**3.676E-03**
	15	5.257E-03	5.613E-03	5.704E-03	3.950E-01	1.918E-01	2.512E-01	2.183E-01	1.687E-02	**4.322E-03**
		7.298E-03 =	8.156E-03+	4.619E-01 +	4.628E-01 +	2.729E-01+	2.773E-01+	7.359E-01 +	1.998E-02 +	**6.497E-03**
		9.578E-03	1.319E-02	6.925E-01	5.310E-01	3.441E-01	3.023E-01	9.442E-01	2.240E-02	9.600E-03
(+/=/-)		17/2/1	18/1/1	17/3/0	19/1/0	19/0/1	19/0/1	20/0/0	19/1/0	

Table 5: Best, median and worst HV values obtained by MaOPSO-DP and other algorithms on DTLZ instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III [1]	LEAF [37]	$\theta-$DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
DTLZ1	3	9.73656E-01 9.73423E-01 + 9.72470E-01	9.73583E-01 9.73434E-01 + 9.72982E-01	9.73593E-01 9.73395E-01 + 9.72737E-01	9.73046E-01 9.72280E-01 + 9.70319E-01	9.72445E-01 9.69382E-01 + 9.57022E-01	9.72251E-01 9.71936E-01 + 9.71078E-01	9.71801E-01 9.70241E-01 + 9.66553E-01	1.10624E-01 1.73248E-01 + 2.63190E-01	**9.73687E-01** **9.73669E-01** **9.73663E-01**
	5	9.98982E-01 9.98977E-01 + 9.98963E-01	9.98982E-01 9.98975E-01 + 9.98963E-01	9.98984E-01 9.98972E-01 + 9.98961E-01	9.98829E-01 9.98491E-01 + 9.98189E-01	9.98152E-01 9.97497E-01 + 9.96326E-01	9.98277E-01 9.97662E-01 + 9.94801E-01	9.98335E-01 9.97537E-01 + 9.96799E-01	9.66164E-01 8.64720E-01 + 5.97339E-01	**9.98988E-01** **9.98986E-01** **9.98984E-01**
	8	9.99974E-01 9.99971E-01 + 9.99896E-01	9.99977E-01 9.99972E-01 + 9.99966E-01	9.99967E-01 9.99954E-01 + 9.99927E-01	9.99946E-01 9.99812E-01 + 9.99528E-01	9.99707E-01 9.99361E-01 + 9.98801E-01	9.99884E-01 9.99807E-01 + 9.99051E-01	9.99432E-01 9.97911E-01 + 9.95176E-01	9.36488E-01 6.90423E-01 + 5.40445E-01	**9.99980E-01** **9.99979E-01** **9.99978E-01**
	10	9.99998E-01 9.99997E-01 + 9.99988E-01	9.99998E-01 9.99997E-01 + 9.99996E-01	9.99995E-01 9.99991E-01 + 9.99985E-01	9.99995E-01 9.99984E-01 + 9.99965E-01	9.99946E-01 9.99849E-01 + 9.99748E-01	9.99994E-01 9.99972E-01 + 9.99870E-01	9.99808E-01 9.99563E-01 + 9.99105E-01	9.99390E-01 9.93112E-01 + 9.69177E-01	**9.99999E-01** **9.99999E-01** **9.99999E-01**
DTLZ2	3	9.26692E-01 9.26640E-01 + 9.26584E-01	9.26662E-01 9.26636E-01 + 9.26577E-01	9.26700E-01 9.26662E-01 + 9.26607E-01	**9.26959E-01** 9.24944E-01 + 9.21424E-01	9.26836E-01 9.26582E-01 + 9.26294E-01	9.26472E-01 9.26058E-01 + 9.25503E-01	9.26848E-01 9.26663E-01 + 9.26305E-01	9.26643E-01 9.26598E-01 + 9.26545E-01	9.26794E-01 **9.26725E-01** **9.26704E-01**
	5	9.90505E-01 9.90474E-01 + 9.90418E-01	9.90492E-01 9.90462E-01 + 9.90443E-01	9.90533E-01 9.90520E-01 + 9.90496E-01	9.90178E-01 9.89242E-01 + 9.87013E-01	9.90561E-01 9.90446E-01 + 9.90275E-01	9.89463E-01 9.89135E-01 + 9.87376E-01	9.90545E-01 9.90456E-01 + 9.90313E-01	**9.90586E-01** 9.90549E-01 + 9.90535E-01	9.90569E-01 **9.90559E-01** **9.90551E-01**
	8	9.99337E-01 9.99328E-01 + 9.99315E-01	9.99339E-01 9.99329E-01 + 9.99316E-01	9.99349E-01 9.99340E-01 + 9.99326E-01	9.99140E-01 9.98795E-01 + 9.97946E-01	**9.99425E-01** **9.99409E-01** − **9.99382E-01**	9.99055E-01 9.98856E-01 + 9.98726E-01	9.99409E-01 9.99396E-01 − 9.99367E-01	9.99352E-01 9.99343E-01 + 9.99326E-01	9.99361E-01 9.99360E-01 9.99357E-01
	10	9.99919E-01 9.99917E-01 + 9.99916E-01	9.99920E-01 9.99919E-01 + 9.99915E-01	9.99920E-01 9.99919E-01 + 9.99916E-01	9.99872E-01 9.99843E-01 + 9.99699E-01	**9.99927E-01** **9.99924E-01** = 9.99915E-01	9.99825E-01 9.99801E-01 + 9.99783E-01	9.99925E-01 9.99918E-01 + 9.99910E-01	9.99921E-01 9.99920E-01 + 9.99920E-01	9.99923E-01 9.99923E-01 **9.99923E-01**
DTLZ3	3	9.26593E-01 9.25882E-01 + 9.24558E-01	9.26466E-01 9.26084E-01 + 9.24700E-01	9.26602E-01 9.26050E-01 + 9.21048E-01	9.26760E-01 9.26291E-01 + 9.22296E-01	9.26726E-01 9.26162E-01 + 9.24835E-01	9.26684E-01 9.26131E-01 + 4.40325E-03	**9.27081E-01** 9.26728E-01 = 9.26192E-01	9.26633E-01 9.26310E-01 + 8.60481E-01	9.26783E-01 **9.26731E-01** **9.26730E-01**
	5	9.90525E-01 9.90469E-01 + 9.90135E-01	9.90536E-01 9.90425E-01 + 9.90252E-01	9.90531E-01 9.90480E-01 + 9.90372E-01	9.89525E-01 9.89034E-01 + 9.86805E-01	9.90489E-01 9.90352E-01 + 9.90031E-01	9.88524E-01 9.86276E-01 + 9.78306E-01	9.90580E-01 9.90498E-01 + 9.90364E-01	9.90476E-01 9.90281E-01 + 9.89929E-01	**9.90588E-01** **9.90575E-01** **9.90571E-01**
	8	9.99334E-01 9.99264E-01 + 9.99166E-01	9.99323E-01 9.99281E-01 + 9.99162E-01	9.99342E-01 9.99313E-01 + 9.98320E-01	9.98655E-01 9.97544E-01 + 9.95905E-01	9.99380E-01 9.99342E-01 + 9.99194E-01	9.98633E-01 9.98135E-01 + 9.97371E-01	**9.99399E-01** 9.98167E-01 + 9.25750E-01	9.99043E-01 9.90911E-01 + 7.70214E-01	9.99364E-01 **9.99364E-01** **9.99363E-01**
	10	9.99922E-01 9.99918E-01 + 9.99909E-01	9.99922E-01 9.99919E-01 + 9.99914E-01	9.99922E-01 9.99921E-01 + 9.99917E-01	9.99843E-01 9.99652E-01 + 9.98452E-01	9.99915E-01 9.99907E-01 + 9.99896E-01	9.99661E-01 9.99590E-01 + 9.99363E-01	9.99917E-01 9.98656E-01 + 4.99770E-01	9.99915E-01 9.99910E-01 + 9.99904E-01	**9.99924E-01** **9.99923E-01** **9.99923E-01**
DTLZ4	3	9.26777E-01 9.26733E-01 = 7.98533E-01	9.26730E-01 9.26729E-01 + 8.01492E-01	9.26774E-01 9.26731E-01 = **9.26728E-01**	9.26980E-01 **9.26785E-01** = 8.01917E-01	9.26826E-01 9.26507E-01 + 9.25961E-01	9.26525E-01 9.26163E-01 + 9.25349E-01	**9.27044E-01** 9.26665E-01 + 8.01590E-01	9.26721E-01 9.26695E-01 + 9.26672E-01	9.26748E-01 9.26730E-01 9.26728E-01
	5	9.90593E-01 9.90578E-01 − 9.90571E-01	9.90571E-01 9.90570E-01 = 9.90568E-01	9.90590E-01 9.90575E-01 − 9.90571E-01	9.90528E-01 9.90449E-01 + 9.90038E-01	9.90510E-01 9.90284E-01 + 9.90181E-01	9.90341E-01 9.90211E-01 + 9.90066E-01	**9.90706E-01** 9.90524E-01 = 9.90422E-01	9.90601E-01 **9.90593E-01** − **9.90586E-01**	9.90574E-01 9.90571E-01 9.90570E-01
	8	9.99365E-01 9.99364E-01 = 9.99363E-01	9.99365E-01 9.99364E-01 = 9.99363E-01	9.99364E-01 9.99364E-01 = 9.71325E-01	9.99313E-01 9.99099E-01 + 9.98723E-01	9.99395E-01 9.99359E-01 = 9.98380E-01	9.99390E-01 9.99339E-01 + 9.99309E-01	**9.99433E-01** **9.99405E-01** = 9.89031E-01	9.99391E-01 9.99385E-01 − **9.99378E-01**	9.99364E-01 9.99364E-01 9.99364E-01
	10	9.99924E-01 9.99923E-01 = 9.99923E-01	9.99923E-01 9.99923E-01 − 9.99923E-01	9.99924E-01 9.99924E-01 = 9.99923E-01	9.99899E-01 9.99884E-01 + 9.99839E-01	9.99921E-01 9.99916E-01 + 9.99910E-01	9.99924E-01 9.99917E-01 + 9.99913E-01	**9.99930E-01** **9.99927E-01** − 9.99921E-01	9.99927E-01 9.99927E-01 − **9.99926E-01**	9.99923E-01 9.99923E-01 9.99923E-01
(+/=/-)		12/3/1	13/2/1	12/3/1	15/1/0	13/2/1	16/0/0	11/3/2	13/0/3	

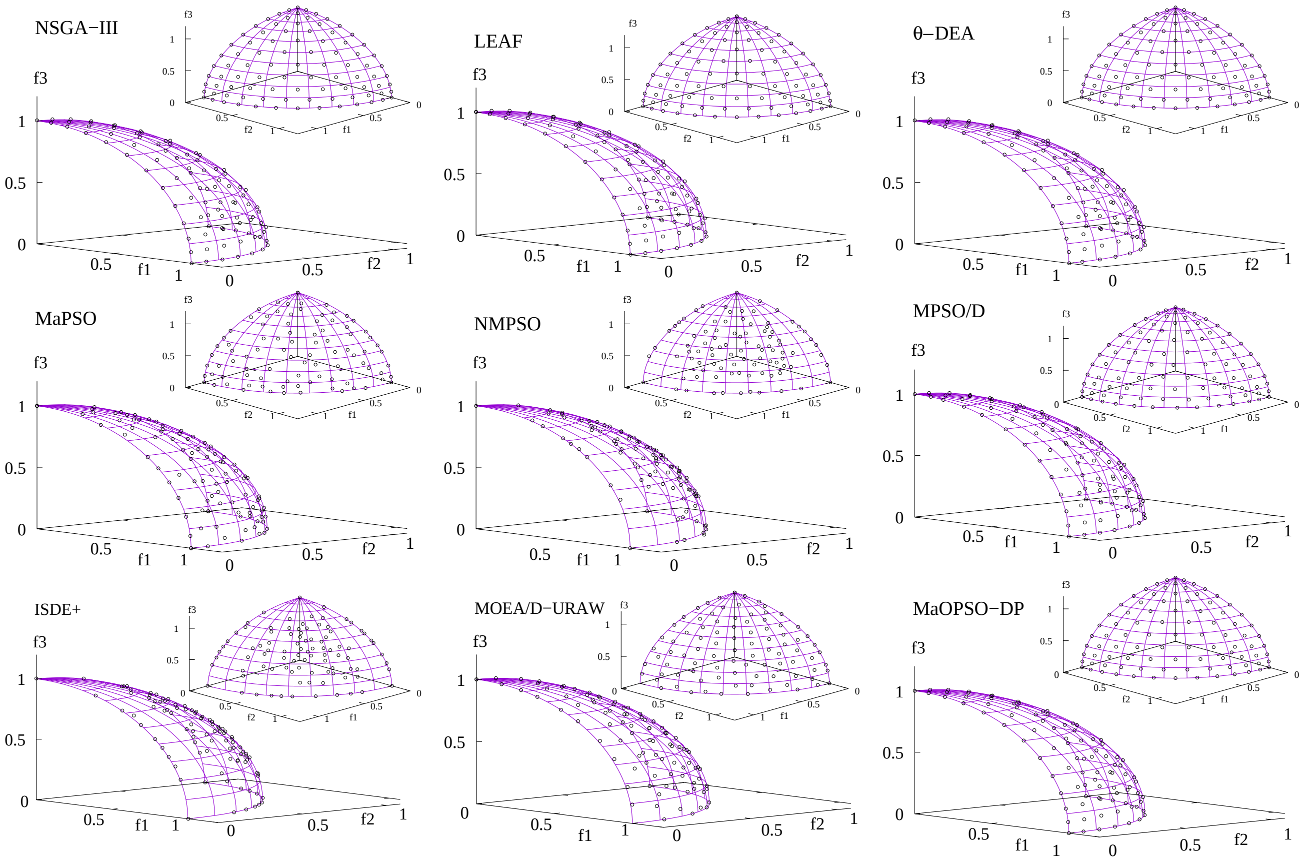

Figure 3: The value path of the obtained non-dominated solutions by the algorithms for 3-objective DTLZ3 problem.

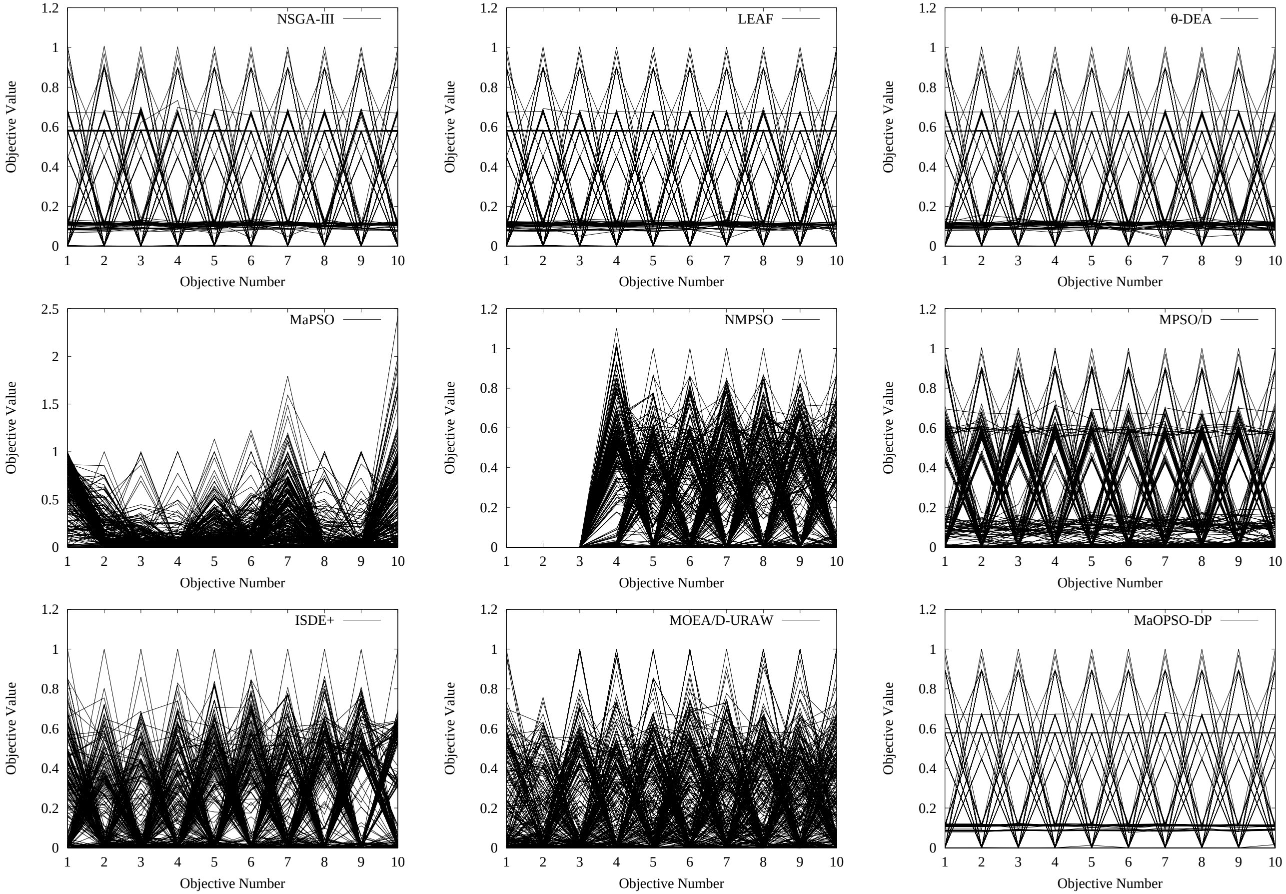

Figure 4: The value path of the obtained non-dominated solutions by the algorithms for 10-objective DTLZ3 problem.

Table 6: Best, median and worst IGD values obtained by MaOPSO-DP and other algorithms on WFG instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III [1]	LEAF [37]	$\theta-$DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
WFG1	3	2.368E-01 2.401E-01 − 2.425E-01	3.385E-01 3.674E-01 + 3.753E-01	3.226E-01 3.624E-01 + 3.758E-01	**4.198E-02** **4.779E-02** − **5.658E-02**	5.298E-01 5.303E-01 + 5.319E-01	1.815E-01 2.441E-01 − 3.000E-01	1.074E-01 1.426E-01 − 2.678E-01	5.047E-01 5.486E-01 + 5.626E-01	2.900E-01 3.234E-01 3.416E-01
	5	2.403E-01 2.431E-01 − 2.469E-01	3.729E-01 3.992E-01 + 4.053E-01	3.033E-01 3.434E-01 − 3.683E-01	**7.920E-02** **8.167E-02** − **8.894E-02**	5.466E-01 5.474E-01 + 5.482E-01	2.557E-01 3.263E-01 − 3.981E-01	1.716E-01 2.083E-01 − 3.233E-01	5.365E-01 5.532E-01 + 5.589E-01	3.229E-01 3.786E-01 4.274E-01
	8	3.005E-01 3.045E-01 + 3.225E-01	1.940E-01 2.203E-01 + 3.710E-01	2.321E-01 2.578E-01 + 2.856E-01	**1.357E-01** **1.459E-01** − **1.492E-01**	5.780E-01 5.815E-01 + 5.869E-01	2.464E-01 3.254E-01 + 4.288E-01	2.626E-01 3.137E-01 + 4.499E-01	5.799E-01 5.875E-01 + 5.927E-01	1.691E-01 2.087E-01 2.910E-01
	10	3.008E-01 3.122E-01 + 3.203E-01	1.647E-01 1.747E-01 − 8.374E-01	2.161E-01 2.270E-01 + 2.411E-01	**1.482E-01** **1.628E-01** − **1.713E-01**	5.924E-01 5.939E-01 + 5.963E-01	1.998E-01 2.257E-01 + 3.988E-01	2.661E-01 3.948E-01 + 4.880E-01	5.934E-01 5.972E-01 + 6.012E-01	1.802E-01 1.917E-01 2.034E-01
	15	4.556E-01 4.584E-01 + 4.626E-01	3.211E-01 3.390E-01 + 3.601E-01	4.116E-01 4.423E-01 + 5.219E-01	**2.564E-01** **2.800E-01** − **2.933E-01**	6.571E-01 6.632E-01 + 6.707E-01	3.260E-01 3.443E-01 + 3.629E-01	4.164E-01 5.007E-01 + 6.072E-01	6.259E-01 6.368E-01 + 6.479E-01	2.734E-01 3.153E-01 3.660E-01
WFG2	3	3.332E-01 3.346E-01 + 3.660E-01	**1.602E-02** **2.036E-02** = 9.824E-02	3.363E-02 4.112E-02 + 1.146E-01	3.555E-02 3.868E-02 + 4.352E-02	6.131E-01 6.143E-01 + 7.141E-01	4.398E-02 4.785E-02 + 5.373E-02	1.078E-01 1.286E-01 + 1.584E-01	6.744E-02 7.278E-02 + 7.588E-02	2.652E-02 2.898E-02 **3.021E-02**
	5	2.263E-01 2.271E-01 + 2.479E-01	**5.673E-02** **5.928E-02** − 1.599E-01	6.936E-02 7.950E-02 = 1.940E-01	6.748E-02 7.019E-02 − **7.399E-02**	7.263E-01 7.269E-01 + 8.214E-01	7.156E-02 7.499E-02 − 7.919E-02	1.570E-01 1.737E-01 + 2.279E-01	1.061E-01 1.284E-01 + 1.386E-01	8.477E-02 9.236E-02 1.021E-01
	8	2.258E-01 2.451E-01 + 2.664E-01	**8.843E-02** 1.976E-01 + 2.397E-01	1.603E-01 2.544E-01 + 2.620E-01	1.044E-01 1.148E-01 − **1.249E-01**	7.637E-01 7.644E-01 + 8.537E-01	1.155E-01 1.216E-01 = 1.312E-01	1.891E-01 2.375E-01 + 2.449E-01	1.670E-01 1.777E-01 + 1.832E-01	1.108E-01 **1.138E-01** 2.817E-01
	10	3.306E-01 3.571E-01 + 3.719E-01	1.731E-01 2.619E-01 = 3.165E-01	2.192E-01 3.363E-01 + 3.579E-01	**1.675E-01** **1.772E-01** − **1.972E-01**	8.383E-01 8.387E-01 + 9.312E-01	1.875E-01 2.066E-01 − 2.227E-01	2.941E-01 3.188E-01 + 3.657E-01	2.188E-01 2.404E-01 = 2.569E-01	2.071E-01 2.261E-01 3.118E-01
	15	9.633E-01 9.795E-01 + 9.862E-01	5.929E-01 6.426E-01 + 8.046E-01	8.548E-01 8.886E-01 + 9.463E-01	1.046E-01 3.344E-01 − 5.160E-01	1.142E+00 1.142E+00 + 1.207E+00	**1.187E-02** **2.191E-02** − **4.289E-02**	6.612E-01 7.901E-01 + 9.540E-01	7.537E-01 8.243E-01 + 8.552E-01	5.729E-02 5.615E-01 9.448E-01
WFG3	3	4.275E-01 4.316E-01 + 4.360E-01	1.567E-02 1.901E-02 + 2.736E-02	1.754E-02 2.370E-02 + 5.632E-02	1.412E-02 1.901E-02 + 3.688E-02	4.543E-01 4.585E-01 + 4.633E-01	2.465E-02 2.923E-02 + 3.721E-02	**9.162E-03** **1.192E-02** − **1.585E-02**	6.960E-02 8.092E-02 + 9.453E-02	1.376E-02 1.609E-02 2.879E-02
	5	4.317E-01 4.366E-01 + 4.416E-01	**3.503E-02** **4.667E-02** − **6.037E-02**	3.815E-02 5.406E-02 = 7.908E-02	3.669E-02 4.716E-02 − 7.953E-02	5.244E-01 5.255E-01 + 5.291E-01	4.339E-02 5.463E-02 − 6.321E-02	3.961E-02 5.471E-02 = 6.874E-02	1.360E-01 1.631E-01 + 2.061E-01	4.500E-02 6.211E-02 8.371E-02
	8	4.754E-01 4.767E-01 + 4.793E-01	**5.955E-02** 3.004E-01 = 5.617E-01	1.275E-01 3.900E-01 = 5.832E-01	9.734E-02 1.399E-01 − 2.582E-01	5.626E-01 5.638E-01 + 5.655E-01	1.035E-01 1.645E-01 − 2.385E-01	6.499E-02 **8.636E-02** − **1.315E-01**	2.025E-01 2.516E-01 − 3.103E-01	2.066E-01 2.723E-01 4.772E-01
	10	4.812E-01 4.819E-01 + 4.829E-01	4.436E-02 8.091E-02 − 2.346E-01	3.902E-02 2.134E-01 = 6.376E-01	1.035E-01 1.537E-01 − 2.052E-01	5.730E-01 5.736E-01 + 5.755E-01	1.206E-01 1.762E-01 − 2.183E-01	**3.068E-02** **4.761E-02** − **8.002E-02**	2.410E-01 2.685E-01 = 2.961E-01	1.339E-01 2.674E-01 4.348E-01
	15	4.913E-01 4.920E-01 = 4.929E-01	8.268E-02 3.211E-01 = 5.716E-01	6.080E-02 6.337E-01 + 7.091E-01	2.123E-01 2.973E-01 − 4.525E-01	5.837E-01 5.857E-01 + 5.893E-01	1.383E-01 2.069E-01 − 2.956E-01	**3.706E-02** **7.914E-02** − **1.659E-01**	2.770E-01 3.366E-01 − 3.652E-01	1.300E-01 4.179E-01 8.026E-01
WFG4	3	6.983E-01 6.986E-01 + 6.989E-01	4.662E-03 6.102E-03 − **7.890E-03**	**4.495E-03** **5.214E-03** − 1.251E-02	5.405E-02 5.511E-02 + 5.857E-02	4.169E-01 4.205E-01 + 4.248E-01	5.588E-02 6.121E-02 + 6.711E-02	6.982E-02 7.549E-02 + 8.296E-02	7.830E-02 8.157E-02 + 8.399E-02	4.657E-03 7.411E-03 9.680E-03
	5	7.735E-01 7.746E-01 + 7.753E-01	1.697E-02 1.981E-02 − 2.327E-02	**1.634E-02** **1.945E-02** − **2.190E-02**	1.550E-01 1.584E-01 + 1.623E-01	6.299E-01 6.402E-01 + 6.455E-01	1.702E-01 1.765E-01 + 1.840E-01	1.666E-01 1.729E-01 + 1.791E-01	2.817E-01 2.965E-01 + 3.020E-01	1.753E-02 2.197E-02 2.497E-02
	8	8.319E-01 8.325E-01 + 8.331E-01	2.925E-02 3.602E-02 + 4.398E-02	**2.863E-02** 3.502E-02 = 3.968E-02	2.966E-01 3.076E-01 + 3.194E-01	8.848E-01 8.901E-01 + 9.027E-01	2.983E-01 3.163E-01 + 3.293E-01	2.136E-01 2.382E-01 + 2.652E-01	6.441E-01 6.882E-01 + 6.953E-01	2.882E-02 **3.400E-02** **3.797E-02**
	10	8.544E-01 8.547E-01 + 8.553E-01	3.345E-02 **3.833E-02** − **4.921E-02**	4.052E-02 8.208E-02 + 5.009E-01	3.408E-01 3.507E-01 + 3.555E-01	9.651E-01 9.786E-01 + 9.930E-01	3.301E-01 3.533E-01 + 3.655E-01	3.178E-01 3.325E-01 + 3.437E-01	7.808E-01 7.909E-01 + 7.963E-01	**3.271E-02** 4.077E-02 9.718E-02
	15	9.097E-01 9.416E-01 + 9.549E-01	5.195E-01 5.701E-01 = 6.860E-01	7.117E-01 7.746E-01 + 8.398E-01	4.717E-01 5.048E-01 − 6.468E-01	1.201E+00 1.220E+00 + 1.230E+00	3.686E-01 4.035E-01 − **4.336E-01**	**1.683E-01** **2.354E-01** − 4.855E-01	8.738E-01 9.729E-01 + 9.905E-01	5.447E-01 6.003E-01 7.587E-01
WFG5	3	6.927E-01 6.931E-01 + 6.932E-01	2.956E-02 **3.158E-02** = 3.563E-02	2.989E-02 3.235E-02 = 3.703E-02	6.128E-02 6.483E-02 + 6.794E-02	2.677E-01 2.736E-01 + 2.866E-01	6.614E-02 7.025E-02 + 7.468E-02	7.304E-02 7.619E-02 + 8.018E-02	7.880E-02 8.056E-02 + 8.129E-02	**2.905E-02** 3.198E-02 **3.370E-02**
	5	7.721E-01 7.723E-01 + 7.727E-01	**3.501E-02** 3.990E-02 = **4.426E-02**	3.665E-02 4.169E-02 + 4.831E-02	1.604E-01 1.627E-01 + 1.672E-01	6.755E-01 6.805E-01 + 6.875E-01	1.879E-01 1.946E-01 + 2.081E-01	1.703E-01 1.759E-01 + 1.803E-01	2.939E-01 3.035E-01 + 3.054E-01	3.562E-02 **3.966E-02** 4.578E-02

	M	NSGA-III [1]	LEAF [37]	$\theta-$DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
	8	8.317E-01 8.320E-01 + 8.321E-01	4.612E-02 5.127E-02 = **5.356E-02**	4.902E-02 5.068E-02 = 5.886E-02	2.989E-01 3.074E-01 + 3.190E-01	9.400E-01 9.648E-01 + 9.729E-01	3.410E-01 3.560E-01 + 3.628E-01	2.563E-01 2.801E-01 + 3.038E-01	5.513E-01 6.259E-01 + 6.710E-01	**4.453E-02** **5.013E-02** 5.631E-02
	10	8.541E-01 8.543E-01 + 8.545E-01	4.386E-02 4.948E-02 = 5.475E-02	4.520E-02 5.117E-02 + 5.804E-02	3.396E-01 3.458E-01 + 3.552E-01	1.026E+00 1.028E+00 + 1.050E+00	3.941E-01 4.037E-01 + 4.128E-01	3.358E-01 3.456E-01 + 3.550E-01	6.304E-01 6.867E-01 + 7.555E-01	**4.215E-02** **4.737E-02** **5.226E-02**
	15	8.889E-01 8.890E-01 + 9.110E-01	**3.685E-02** **3.872E-02** − **4.269E-02**	4.431E-02 1.730E-01 + 5.129E-01	4.411E-01 4.857E-01 + 5.149E-01	1.186E+00 1.187E+00 + 1.188E+00	4.691E-01 4.930E-01 + 5.144E-01	2.660E-01 4.853E-01 + 7.098E-01	7.120E-01 7.397E-01 + 8.259E-01	3.726E-02 5.312E-02 8.015E-02
WFG6	3	6.918E-01 6.934E-01 + 6.943E-01	2.090E-02 2.810E-02 + 3.307E-02	2.311E-02 2.933E-02 + 3.853E-02	5.766E-02 6.253E-02 + 6.765E-02	5.539E-01 5.607E-01 + 5.723E-01	5.568E-02 7.527E-02 + 8.048E-02	8.269E-02 8.737E-02 + 9.310E-02	8.386E-02 8.852E-02 + 9.369E-02	**1.282E-02** **1.308E-02** **1.361E-02**
	5	7.713E-01 7.724E-01 + 7.733E-01	**2.719E-02** 3.419E-02 − **4.051E-02**	2.835E-02 **3.371E-02** − 4.327E-02	1.558E-01 1.605E-01 + 1.652E-01	8.832E-01 8.848E-01 + 8.867E-01	1.667E-01 1.726E-01 + 1.768E-01	1.736E-01 1.796E-01 + 1.827E-01	2.506E-01 2.879E-01 + 3.107E-01	5.566E-02 6.022E-02 6.321E-02
	8	8.314E-01 8.319E-01 + 8.323E-01	3.474E-02 3.936E-02 − 4.964E-02	**2.971E-02** **3.645E-02** − **4.812E-02**	2.973E-01 3.055E-01 + 3.137E-01	1.075E+00 1.076E+00 + 1.079E+00	3.032E-01 3.264E-01 + 3.351E-01	2.178E-01 2.402E-01 + 2.589E-01	5.889E-01 6.812E-01 + 7.068E-01	5.740E-02 6.148E-02 6.676E-02
	10	8.539E-01 8.545E-01 + 8.553E-01	**3.043E-02** 3.719E-02 − **4.184E-02**	3.120E-02 **3.639E-02** − 4.364E-02	3.356E-01 3.441E-01 + 3.541E-01	1.139E+00 1.140E+00 + 1.140E+00	3.523E-01 3.641E-01 + 3.742E-01	2.477E-01 2.764E-01 + 2.988E-01	6.825E-01 7.600E-01 + 8.026E-01	5.703E-02 6.210E-02 6.810E-02
	15	8.886E-01 8.892E-01 + 8.894E-01	**3.043E-02** 3.719E-02 − **4.184E-02**	3.103E-02 **3.573E-02** − 4.440E-02	4.615E-01 4.830E-01 + 5.734E-01	1.248E+00 1.248E+00 + 1.249E+00	4.146E-01 4.442E-01 + 4.742E-01	1.329E-01 2.380E-01 + 5.003E-01	7.721E-01 8.910E-01 + 9.531E-01	3.923E-02 6.210E-02 6.498E-02
WFG7	3	6.992E-01 6.993E-01 + 6.995E-01	1.679E-03 2.606E-03 + 3.288E-03	1.891E-03 2.331E-03 + 2.728E-03	5.211E-02 5.537E-02 + 5.938E-02	4.545E-01 4.554E-01 + 4.560E-01	5.180E-02 5.561E-02 + 5.824E-02	6.676E-02 7.462E-02 + 7.924E-02	7.219E-02 7.614E-02 + 7.841E-02	**1.423E-03** **1.841E-03** **2.172E-03**
	5	7.761E-01 7.764E-01 + 7.766E-01	6.518E-03 7.325E-03 + 1.424E-02	5.630E-03 6.636E-03 = 1.295E-02	1.506E-01 1.576E-01 + 1.612E-01	5.368E-01 5.397E-01 + 5.417E-01	1.586E-01 1.638E-01 + 1.715E-01	1.616E-01 1.699E-01 + 1.763E-01	2.753E-01 2.928E-01 + 2.997E-01	**5.413E-03** **6.451E-03** **7.520E-03**
	8	8.334E-01 8.338E-01 + 8.340E-01	1.721E-02 1.953E-02 + 2.211E-02	1.656E-02 1.857E-02 + 2.104E-02	2.931E-01 3.071E-01 + 3.229E-01	6.161E-01 6.224E-01 + 6.275E-01	3.037E-01 3.108E-01 + 3.255E-01	2.223E-01 2.419E-01 + 2.600E-01	5.686E-01 6.626E-01 + 6.972E-01	**1.389E-02** **1.559E-02** **1.823E-02**
	10	8.558E-01 8.559E-01 + 8.561E-01	1.944E-02 2.065E-02 + 2.310E-02	2.017E-02 2.248E-02 + 2.455E-02	3.409E-01 3.465E-01 + 3.583E-01	6.539E-01 6.566E-01 + 6.637E-01	3.275E-01 3.424E-01 + 3.562E-01	2.630E-01 2.923E-01 + 3.088E-01	6.554E-01 7.262E-01 + 7.949E-01	**1.336E-02** **1.446E-02** **1.769E-02**
	15	8.895E-01 8.945E-01 + 9.311E-01	3.739E-02 7.754E-02 = **4.482E-01**	5.381E-01 5.978E-01 + 6.633E-01	4.711E-01 4.884E-01 + 5.455E-01	7.352E-01 7.446E-01 + 7.694E-01	3.683E-01 4.085E-01 + 4.599E-01	1.879E-01 4.119E-01 + 5.800E-01	7.293E-01 8.120E-01 + 8.603E-01	**7.991E-03** **1.769E-02** 5.958E-01
WFG8	3	6.831E-01 6.844E-01 + 6.860E-01	6.510E-02 7.235E-02 + 7.515E-02	7.442E-02 7.598E-02 + 7.820E-02	**6.414E-02** 6.843E-02 = 7.393E-02	6.665E-01 6.773E-01 + 6.823E-01	8.379E-02 8.751E-02 + 9.144E-02	8.241E-02 8.783E-02 + 9.183E-02	9.211E-02 9.652E-02 + 9.950E-02	6.544E-02 **6.837E-02** **7.166E-02**
	5	7.580E-01 7.591E-01 + 7.606E-01	1.203E-01 1.302E-01 + 1.370E-01	1.183E-01 1.330E-01 + 1.435E-01	1.661E-01 1.713E-01 + 1.773E-01	9.469E-01 9.667E-01 + 9.712E-01	2.092E-01 2.170E-01 + 2.227E-01	1.746E-01 1.841E-01 + 1.901E-01	2.393E-01 2.538E-01 + 2.696E-01	**1.107E-01** **1.277E-01** **1.369E-01**
	8	8.123E-01 8.129E-01 + 8.140E-01	2.049E-01 2.211E-01 = 2.413E-01	2.396E-01 2.578E-01 + 2.664E-01	3.108E-01 3.198E-01 + 3.432E-01	1.124E+00 1.133E+00 + 1.135E+00	3.715E-01 3.925E-01 + 4.047E-01	2.468E-01 2.605E-01 + 2.843E-01	4.416E-01 4.806E-01 + 5.618E-01	**1.320E-01** **2.205E-01** **2.316E-01**
	10	8.371E-01 8.374E-01 + 8.392E-01	1.174E-01 2.025E-01 = **2.660E-01**	2.807E-01 3.111E-01 + 3.274E-01	3.561E-01 3.656E-01 + 3.958E-01	1.175E+00 1.180E+00 + 1.185E+00	4.332E-01 4.442E-01 + 4.511E-01	2.921E-01 3.034E-01 + 3.268E-01	5.153E-01 5.329E-01 + 5.915E-01	**6.663E-02** **1.892E-01** 4.417E-01
	15	9.389E-01 9.458E-01 + 9.504E-01	5.429E-01 5.917E-01 − 6.231E-01	6.154E-01 6.565E-01 − 6.832E-01	5.172E-01 5.737E-01 − 6.065E-01	1.268E+00 1.276E+00 + 1.279E+00	5.401E-01 5.563E-01 − **5.715E-01**	**2.385E-01** **5.206E-01** − 7.315E-01	5.997E-01 6.650E-01 − 7.443E-01	6.604E-01 6.987E-01 7.335E-01
WFG9	3	6.869E-01 6.906E-01 + 6.962E-01	2.867E-02 6.484E-02 + 6.542E-02	3.060E-02 **5.021E-02** = 6.679E-02	5.708E-02 5.948E-02 = **6.287E-02**	9.316E-02 9.909E-02 + 1.211E-01	6.145E-02 8.471E-02 + 8.889E-02	8.040E-02 8.412E-02 + 8.958E-02	7.730E-02 9.068E-02 + 9.180E-02	**2.060E-02** 6.408E-02 6.530E-02
	5	7.671E-01 7.681E-01 + 7.736E-01	5.670E-02 6.101E-02 = **9.265E-02**	5.407E-02 9.304E-02 + 9.787E-02	1.604E-01 1.672E-01 + 1.712E-01	1.973E-01 2.218E-01 + 2.409E-01	1.940E-01 2.052E-01 + 2.095E-01	1.801E-01 1.861E-01 + 1.904E-01	2.803E-01 2.928E-01 + 3.051E-01	**4.789E-02** **5.478E-02** 1.007E-01
	8	8.278E-01 8.286E-01 + 8.299E-01	**9.049E-02** **1.001E-01** − **1.372E-01**	9.219E-02 1.266E-01 = 1.644E-01	3.120E-01 3.213E-01 + 3.302E-01	3.966E-01 7.237E-01 + 8.421E-01	3.504E-01 3.625E-01 + 3.760E-01	2.480E-01 2.805E-01 + 3.013E-01	5.372E-01 6.037E-01 + 6.771E-01	1.108E-01 1.249E-01 1.425E-01
	10	8.514E-01 8.522E-01 + 8.543E-01	**9.885E-02** **1.071E-01** − **1.358E-01**	1.030E-01 1.190E-01 = 2.237E-01	3.507E-01 3.635E-01 + 3.741E-01	6.421E-01 7.388E-01 + 9.268E-01	3.983E-01 4.063E-01 + 4.165E-01	3.213E-01 3.491E-01 + 3.553E-01	5.999E-01 6.731E-01 + 7.078E-01	1.241E-01 1.424E-01 1.888E-01
	15	8.880E-01 8.901E-01 + 9.010E-01	**1.272E-01** **1.385E-01** − **3.000E-01**	1.865E-01 4.595E-01 − 6.168E-01	4.880E-01 5.311E-01 − 5.687E-01	9.503E-01 1.047E+00 + 1.131E+00	5.051E-01 5.221E-01 − 5.607E-01	1.924E-01 4.262E-01 − 6.313E-01	6.277E-01 7.248E-01 + 7.918E-01	2.533E-01 5.814E-01 7.423E-01
(+/=/-)		42/1/2	16/13/16	25/11/9	27/2/16	45/0/0	32/1/12	35/1/9	40/2/3	

Table 7: Best, median and worst HV values obtained by MaOPSO-DP and other algorithms on WFG instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III [1]	LEAF [37]	θ−DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
WFG1	3	7.12840E-01	7.30150E-01	7.25130E-01	**9.89810E-01**	5.88180E-01	8.45320E-01	9.53340E-01	6.27010E-01	7.57960E-01
		7.03900E-01+	7.05510E-01+	7.02160E-01+	**9.86410E-01** −	5.85360E-01+	7.94780E-01 −	9.15990E-01 −	6.03010E-01+	7.31700E-01
		6.95820E-01	6.97660E-01	6.93220E-01	**9.79280E-01**	5.81060E-01	7.54300E-01	8.75650E-01	5.97620E-01	7.19640E-01
	5	6.50050E-01	6.69450E-01	7.17250E-01	**9.99670E-01**	5.32820E-01	7.64530E-01	9.94030E-01	5.73540E-01	6.99500E-01
		6.44320E-01+	6.49440E-01=	6.82270E-01−	**9.99050E-01** −	5.31880E-01+	7.00990E-01 −	9.60600E-01 −	5.55170E-01+	6.57980E-01
		6.40330E-01	6.45800E-01	6.63790E-01	**9.89020E-01**	5.30630E-01	6.47850E-01	9.23130E-01	5.51300E-01	6.26690E-01
	8	6.61610E-01	9.05230E-01	8.67910E-01	**9.99970E-01**	4.75920E-01	8.05640E-01	9.94250E-01	5.04730E-01	9.83650E-01
		6.00340E-01+	8.29650E-01+	8.20300E-01+	**9.99760E-01** −	4.74400E-01+	6.95770E-01 +	9.75280E-01 −	4.97520E-01+	9.54440E-01
		5.86340E-01	6.55680E-01	7.55540E-01	**9.84900E-01**	4.72460E-01	6.00520E-01	9.05040E-01	4.94130E-01	8.38710E-01
	10	6.93100E-01	9.84430E-01	9.60560E-01	**1.00000E+00**	4.51400E-01	9.61450E-01	9.94210E-01	4.87320E-01	9.85280E-01
		6.51950E-01+	9.59740E-01+	9.20020E-01+	**9.99990E-01** −	4.50770E-01+	9.17990E-01 +	9.49740E-01 +	4.71590E-01+	9.85270E-01
		5.66840E-01	5.78980E-01	8.71940E-01	**9.99890E-01**	4.49430E-01	6.20230E-01	8.74000E-01	4.69490E-01	9.62100E-01
WFG2	3	9.87120E-01	9.87550E-01	9.84370E-01	9.88060E-01	4.77780E-01	9.82720E-01	9.66840E-01	9.62090E-01	**9.88310E-01**
		9.84290E-01+	9.85110E-01+	9.82220E-01+	**9.87140E-01** =	4.77630E-01+	9.80360E-01 +	9.48740E-01 +	9.56080E-01+	9.87050E-01
		8.93220E-01	8.92750E-01	8.90330E-01	9.75840E-01	4.34650E-01	9.70600E-01	9.30740E-01	9.48320E-01	**9.84310E-01**
	5	9.96560E-01	9.97470E-01	9.89570E-01	**9.99230E-01**	4.45770E-01	9.98280E-01	9.94670E-01	9.61910E-01	9.83600E-01
		9.95630E-01−	9.96330E-01−	9.87110E-01=	**9.98820E-01** −	4.45710E-01+	9.97770E-01 −	9.88390E-01 −	9.33280E-01+	9.79130E-01
		8.97790E-01	8.97050E-01	8.89210E-01	**9.98250E-01**	4.06910E-01	9.95190E-01	9.72660E-01	9.14810E-01	9.75050E-01
	8	9.97490E-01	9.96200E-01	9.76870E-01	9.98820E-01	4.14660E-01	**9.99280E-01**	9.98660E-01	9.53110E-01	9.99270E-01
		9.93110E-01+	8.97450E-01+	8.90790E-01+	9.97860E-01 =	4.14600E-01+	**9.98750E-01**−	9.97050E-01 =	9.41160E-01+	9.97920E-01
		8.94710E-01	8.92770E-01	8.85610E-01	9.95890E-01	3.79180E-01	**9.97110E-01**	9.88950E-01	9.32120E-01	8.94430E-01
	10	9.98030E-01	9.96150E-01	9.92700E-01	9.99310E-01	3.99980E-01	9.99730E-01	**9.99780E-01**	9.63230E-01	9.99360E-01
		9.94850E-01+	9.44820E-01+	8.93340E-01+	9.98270E-01 =	3.99960E-01+	**9.99500E-01**−	9.98810E-01 =	9.55410E-01+	9.98010E-01
		8.95970E-01	8.93820E-01	8.87790E-01	9.97000E-01	3.66030E-01	**9.99040E-01**	9.94910E-01	9.46990E-01	9.76190E-01
WFG3	3	8.75020E-01	8.80340E-01	8.73270E-01	8.80820E-01	4.88960E-01	8.74740E-01	**8.84510E-01**	8.17660E-01	8.82090E-01
		8.71570E-01+	8.76760E-01=	8.70180E-01+	**8.77420E-01** =	4.87440E-01+	8.65220E-01 +	8.76400E-01 =	7.98010E-01+	8.76950E-01
		8.66020E-01	**8.71800E-01**	8.57720E-01	8.68350E-01	4.84630E-01	8.59630E-01	8.69570E-01	7.90830E-01	8.64890E-01
	5	8.75090E-01	8.88090E-01	8.82240E-01	**8.96540E-01**	4.56700E-01	8.86780E-01	8.73580E-01	7.61850E-01	8.72270E-01
		8.66150E-01−	8.79870E-01−	8.75860E-01−	**8.91580E-01** −	4.56480E-01+	8.72820E-01 −	8.65190E-01 =	7.42160E-01+	8.62730E-01
		8.60270E-01	8.71050E-01	8.70690E-01	**8.72600E-01**	4.55290E-01	8.63440E-01	8.53260E-01	7.24770E-01	8.48990E-01
	8	**8.73700E-01**	8.56560E-01	8.02400E-01	8.70700E-01	4.24920E-01	8.73460E-01	8.64690E-01	7.28840E-01	7.49910E-01
		8.55450E-01−	7.12880E-01=	6.70270E-01=	8.52160E-01 −	4.24730E-01+	**8.62380E-01**−	8.47210E-01 −	7.09460E-01=	7.20210E-01
		7.91750E-01	6.13980E-01	6.02070E-01	8.28700E-01	4.24140E-01	**8.39310E-01**	8.21280E-01	6.92880E-01	6.44390E-01
	10	8.67930E-01	8.67630E-01	8.27920E-01	8.71680E-01	4.10010E-01	8.75170E-01	**8.79960E-01**	7.26310E-01	8.08300E-01
		8.62790E-01−	8.11510E-01−	7.42870E-01=	8.53850E-01 −	4.09860E-01+	**8.63960E-01**−	8.58910E-01 −	7.11270E-01+	7.43110E-01
		8.40580E-01	7.43330E-01	5.81550E-01	8.28130E-01	4.09680E-01	**8.48880E-01**	8.31880E-01	6.84340E-01	6.58400E-01
WFG4	3	9.23800E-01	9.23760E-01	9.23660E-01	**9.25510E-01**	5.96710E-01	9.18590E-01	9.24600E-01	8.99350E-01	9.23410E-01
		9.22670E-01−	9.22510E-01−	9.23080E-01−	**9.24370E-01** −	5.90590E-01+	9.16450E-01 +	9.24010E-01 −	8.83780E-01+	9.21680E-01
		9.21520E-01	9.21010E-01	9.21050E-01	9.21970E-01	5.88430E-01	9.11490E-01	**9.22530E-01**	8.70640E-01	9.19330E-01
	5	9.82040E-01	9.81810E-01	9.81960E-01	**9.88070E-01**	5.63550E-01	9.82000E-01	9.85240E-01	9.07880E-01	9.79760E-01
		9.78910E-01−	9.80020E-01−	9.79810E-01−	**9.86450E-01** −	5.57720E-01+	9.78390E-01 −	9.83590E-01 −	8.93840E-01+	9.76520E-01
		9.74300E-01	9.78700E-01	9.78510E-01	**9.84370E-01**	5.53400E-01	9.73290E-01	9.80810E-01	8.86670E-01	9.74140E-01
	8	9.87140E-01	9.86880E-01	9.86810E-01	**9.97450E-01**	5.03920E-01	9.87910E-01	9.94170E-01	8.48350E-01	9.84500E-01
		9.82120E-01=	9.82270E-01=	9.83240E-01−	**9.96530E-01** −	4.95740E-01+	9.82390E-01 =	9.90470E-01 −	8.24930E-01+	9.81330E-01
		9.77580E-01	9.77270E-01	9.79280E-01	**9.94870E-01**	4.88940E-01	9.73770E-01	9.84960E-01	8.13520E-01	9.75640E-01
	10	9.86710E-01	9.89800E-01	9.87140E-01	**9.99040E-01**	4.78680E-01	9.89000E-01	9.87020E-01	8.32950E-01	9.88520E-01
		9.83350E-01=	9.86470E-01−	9.79080E-01+	**9.98400E-01** −	4.72700E-01+	9.83050E-01 =	9.83120E-01 =	8.16590E-01+	9.83580E-01
		9.77770E-01	9.81940E-01	9.74510E-01	**9.97610E-01**	4.64610E-01	9.73810E-01	9.76080E-01	8.04820E-01	9.74780E-01
WFG5	3	9.03370E-01	9.03510E-01	9.03190E-01	9.03690E-01	7.35380E-01	8.92250E-01	**9.04310E-01**	8.84360E-01	9.03450E-01
		8.99610E-01=	9.02710E-01=	9.01910E-01=	9.01460E-01 =	7.33640E-01+	8.87070E-01 +	**9.03850E-01**−	8.67720E-01+	9.00180E-01
		8.95120E-01	8.97040E-01	8.95220E-01	8.93780E-01	7.30760E-01	8.82730E-01	**9.03310E-01**	8.66070E-01	8.96520E-01
	5	9.59720E-01	9.60070E-01	9.59910E-01	9.61810E-01	7.33620E-01	9.51630E-01	**9.61880E-01**	8.97310E-01	9.60620E-01
		9.58590E-01+	9.58880E-01=	9.57890E-01+	9.60960E-01 −	7.32680E-01+	9.40420E-01 +	**9.61540E-01**−	8.78510E-01+	9.59050E-01
		9.57080E-01	9.57390E-01	9.56180E-01	9.58480E-01	7.31570E-01	9.32700E-01	**9.61000E-01**	8.74390E-01	9.55590E-01
	8	9.62480E-01	9.61710E-01	9.62090E-01	**9.65630E-01**	7.29190E-01	9.45370E-01	9.65440E-01	8.76790E-01	9.64270E-01
		9.60390E-01+	9.60780E-01+	9.60880E-01+	**9.65260E-01** −	7.28080E-01+	9.38590E-01 +	9.65200E-01 −	8.36320E-01+	9.62270E-01
		9.56410E-01	9.58600E-01	9.59480E-01	9.64130E-01	7.25950E-01	9.26950E-01	**9.64770E-01**	8.17800E-01	9.60070E-01
	10	9.61040E-01	9.61450E-01	9.61420E-01	**9.63780E-01**	7.28650E-01	9.44740E-01	9.63400E-01	9.00700E-01	9.62950E-01
		9.60030E-01+	9.60010E-01+	9.60410E-01+	**9.63730E-01** −	7.28100E-01+	9.40820E-01 +	9.63010E-01 −	8.52280E-01+	9.61690E-01
		9.57960E-01	9.57330E-01	9.58330E-01	**9.63460E-01**	7.27320E-01	9.33540E-01	9.62450E-01	8.25720E-01	9.60400E-01

	M	NSGA-III [1]	LEAF [37]	$\theta-$DEA [19]	MOEA/D-URAW [44]	ISDE+ [45]	MaPSO [36]	NMPSO [31]	MPSO/D [22]	MaOPSO-DP
WFG6	3	9.07810E-01 9.04990E-01+ 8.99480E-01	9.10700E-01 9.04950E-01+ 9.01000E-01	9.08680E-01 9.03880E-01+ 8.97020E-01	9.12500E-01 9.06320E-01 + 8.96600E-01	5.82870E-01 5.82530E-01+ 5.82170E-01	9.16270E-01 8.82970E-01 + 8.82230E-01	8.87860E-01 8.87420E-01 + 8.87000E-01	8.98870E-01 8.86690E-01+ 8.69090E-01	**9.16700E-01** **9.16510E-01** **9.16070E-01**
	5	9.65910E-01 9.61300E-01− 9.54490E-01	9.67870E-01 9.61550E-01− 9.56620E-01	9.66090E-01 9.61840E-01− 9.54190E-01	**9.75550E-01** **9.68090E-01** − **9.60750E-01**	5.78340E-01 5.78140E-01+ 5.77830E-01	9.36850E-01 9.36550E-01 + 9.36340E-01	9.45720E-01 9.45600E-01 − 9.45300E-01	9.18220E-01 9.02320E-01+ 8.81100E-01	9.58710E-01 9.37140E-01 9.36830E-01
	8	9.71930E-01 9.65630E-01− 9.60550E-01	9.71610E-01 9.64650E-01− 9.56080E-01	9.73540E-01 9.67020E-01− 9.59750E-01	**9.83830E-01** **9.75890E-01** − **9.67710E-01**	5.72920E-01 5.72610E-01+ 5.72330E-01	9.36130E-01 9.36070E-01 + 9.36000E-01	9.51810E-01 9.51780E-01 − 9.51750E-01	9.17120E-01 8.52540E-01+ 8.24000E-01	9.61220E-01 9.36280E-01 9.36250E-01
	10	9.76850E-01 9.65770E-01− 9.55710E-01	9.75080E-01 9.65710E-01− 9.60820E-01	9.71600E-01 9.65650E-01− 9.58720E-01	**9.85230E-01** **9.77350E-01** − **9.74160E-01**	5.70350E-01 5.70090E-01+ 5.69890E-01	9.32110E-01 9.32100E-01 + 9.32090E-01	9.52080E-01 9.52070E-01 = 9.52060E-01	9.33090E-01 8.55800E-01+ 8.25210E-01	9.77320E-01 9.57170E-01 9.32140E-01
WFG7	3	9.25530E-01 9.25160E-01+ 9.24530E-01	9.25790E-01 9.25170E-01+ 9.24760E-01	9.25590E-01 9.25280E-01+ 9.24940E-01	**9.26550E-01** **9.26250E-01** − 9.20990E-01	5.34880E-01 5.30990E-01+ 5.21480E-01	9.26440E-01 9.26070E-01 − **9.25590E-01**	9.26150E-01 9.25740E-01 = 9.25370E-01	9.16840E-01 9.06190E-01+ 8.99430E-01	9.25910E-01 9.25790E-01 9.25570E-01
	5	9.87490E-01 9.86590E-01+ 9.85090E-01	9.87850E-01 9.87120E-01+ 9.86030E-01	9.88020E-01 9.87380E-01+ 9.85360E-01	**9.89850E-01** 9.89590E-01 − 9.88900E-01	5.15450E-01 5.12230E-01+ 5.09920E-01	9.89750E-01 **9.89640E-01**− **9.89450E-01**	9.88300E-01 9.87900E-01 = 9.87020E-01	9.32120E-01 9.09730E-01+ 8.96720E-01	9.88590E-01 9.87840E-01 9.86960E-01
	8	9.95210E-01 9.93990E-01+ 9.91930E-01	9.95600E-01 9.94250E-01+ 9.93550E-01	9.95730E-01 9.94500E-01+ 9.93300E-01	9.98720E-01 9.98510E-01 − 9.97870E-01	4.65360E-01 4.63050E-01+ 4.61390E-01	**9.99130E-01** **9.99050E-01**− **9.98940E-01**	9.97750E-01 9.97340E-01 − 9.97110E-01	9.13060E-01 8.65470E-01+ 8.09950E-01	9.96070E-01 9.94960E-01 9.94090E-01
	10	9.96800E-01 9.96000E-01+ 9.94820E-01	9.97630E-01 9.96650E-01+ 9.96060E-01	9.97400E-01 9.96240E-01+ 9.95270E-01	9.99660E-01 9.99410E-01 − 9.98990E-01	4.40560E-01 4.39220E-01+ 4.37880E-01	**9.99840E-01** **9.99810E-01**− **9.99730E-01**	9.97510E-01 9.97130E-01 = 9.96080E-01	9.09110E-01 8.67850E-01+ 8.02420E-01	9.97930E-01 9.97230E-01 9.96620E-01
WFG8	3	9.07130E-01 9.04450E-01+ 9.03040E-01	9.08500E-01 9.06290E-01+ 9.04610E-01	9.06520E-01 9.05010E-01+ 9.03650E-01	**9.11310E-01** 9.08240E-01 = 8.96090E-01	5.53790E-01 5.51200E-01+ 5.46910E-01	9.01710E-01 8.99280E-01 + 8.97750E-01	9.10600E-01 **9.09820E-01**− **9.08790E-01**	8.88910E-01 8.75180E-01+ 8.68040E-01	9.09690E-01 9.07610E-01 9.05350E-01
	5	9.68530E-01 9.65260E-01= 9.60950E-01	9.70340E-01 9.66480E-01= 9.63020E-01	9.68300E-01 9.65400E-01= 9.61560E-01	**9.78460E-01** **9.74940E-01** − **9.71520E-01**	5.64850E-01 5.63880E-01+ 5.61130E-01	9.70330E-01 9.69380E-01 − 9.64090E-01	9.72600E-01 9.70300E-01 − 9.67590E-01	9.40270E-01 9.28370E-01+ 9.15060E-01	9.68270E-01 9.66820E-01 9.63430E-01
	8	9.75980E-01 9.67820E-01= 9.55860E-01	9.65800E-01 9.59860E-01+ 9.48000E-01	9.77110E-01 9.69750E-01= 9.59250E-01	**9.92570E-01** **9.89870E-01** − **9.86180E-01**	5.66960E-01 5.66400E-01+ 5.65780E-01	9.87860E-01 9.83540E-01 − 9.77240E-01	9.80350E-01 9.76630E-01 − 9.74130E-01	9.23970E-01 9.02870E-01+ 8.57380E-01	9.74550E-01 9.66380E-01 9.58660E-01
	10	9.78740E-01 9.73090E-01+ 9.64010E-01	9.79960E-01 9.66620E-01+ 9.56930E-01	9.79780E-01 9.74440E-01+ 9.69200E-01	**9.96060E-01** **9.92380E-01** − **9.90410E-01**	5.67610E-01 5.66950E-01+ 5.65380E-01	9.92790E-01 9.91020E-01 − 9.81690E-01	9.81440E-01 9.77530E-01 + 9.71800E-01	9.30130E-01 9.04170E-01+ 8.63140E-01	9.92390E-01 9.87400E-01 9.73340E-01
WFG9	3	8.93710E-01 8.74290E-01= 8.59390E-01	8.96170E-01 8.61500E-01+ 8.60710E-01	8.94990E-01 8.75520E-01= 8.59720E-01	**9.14930E-01** **9.09390E-01** − **8.95460E-01**	8.60670E-01 8.54850E-01+ 8.16010E-01	8.92260E-01 8.53540E-01 + 8.49270E-01	8.87220E-01 8.86710E-01 − 8.83250E-01	8.82360E-01 8.54950E-01+ 8.50330E-01	9.04400E-01 8.62100E-01 8.60490E-01
	5	9.45610E-01 9.07190E-01+ 9.03020E-01	9.48550E-01 9.43320E-01= 9.05420E-01	9.49050E-01 9.08770E-01+ 9.04500E-01	**9.69740E-01** **9.64580E-01** − **9.58390E-01**	9.00180E-01 8.86240E-01+ 8.69100E-01	9.39710E-01 8.98630E-01 + 8.93420E-01	9.44720E-01 9.42780E-01 = 9.41470E-01	9.02490E-01 8.81690E-01+ 8.70060E-01	9.53340E-01 9.45240E-01 9.04930E-01
	8	9.03250E-01 8.98530E-01+ 8.85810E-01	9.42990E-01 9.33660E-01− 9.03180E-01	9.36850E-01 9.01330E-01= 8.94970E-01	9.62190E-01 **9.51750E-01** − 9.33120E-01	8.71610E-01 8.57720E-01+ 8.17870E-01	8.98110E-01 8.91610E-01 + 8.87440E-01	**9.71170E-01** 9.48720E-01 − **9.47340E-01**	8.61300E-01 8.08870E-01+ 7.75140E-01	9.07570E-01 9.01650E-01 8.92450E-01
	10	9.43600E-01 9.01710E-01− 8.91960E-01	9.47210E-01 9.40410E-01− 9.00730E-01	9.49630E-01 9.40680E-01− 8.98450E-01	9.66150E-01 **9.56540E-01** − 9.40210E-01	8.78310E-01 8.59330E-01+ 8.12880E-01	8.96880E-01 8.91780E-01 + 8.85640E-01	**9.68490E-01** 9.49600E-01 − **9.47760E-01**	8.41020E-01 8.18990E-01+ 7.74320E-01	9.04800E-01 8.99260E-01 8.85690E-01
(+/=/-)		20/6/10	17/8/11	19/8/9	1/6/29	36/0/0	18/2/16	4/10/22	35/1/0	

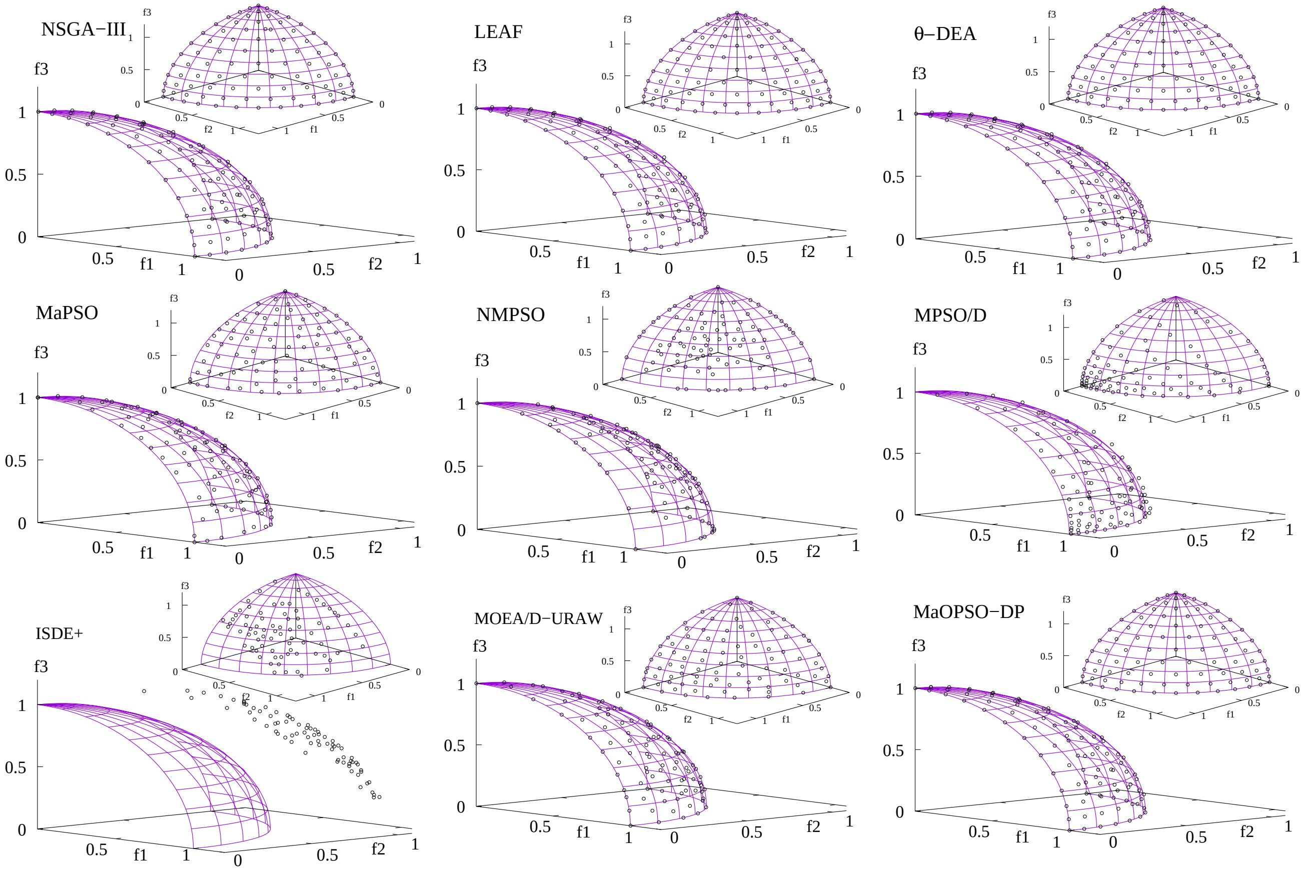

Figure 5: The value path of the obtained non-dominated solutions by the algorithms for 3-objective WFG7 problem.

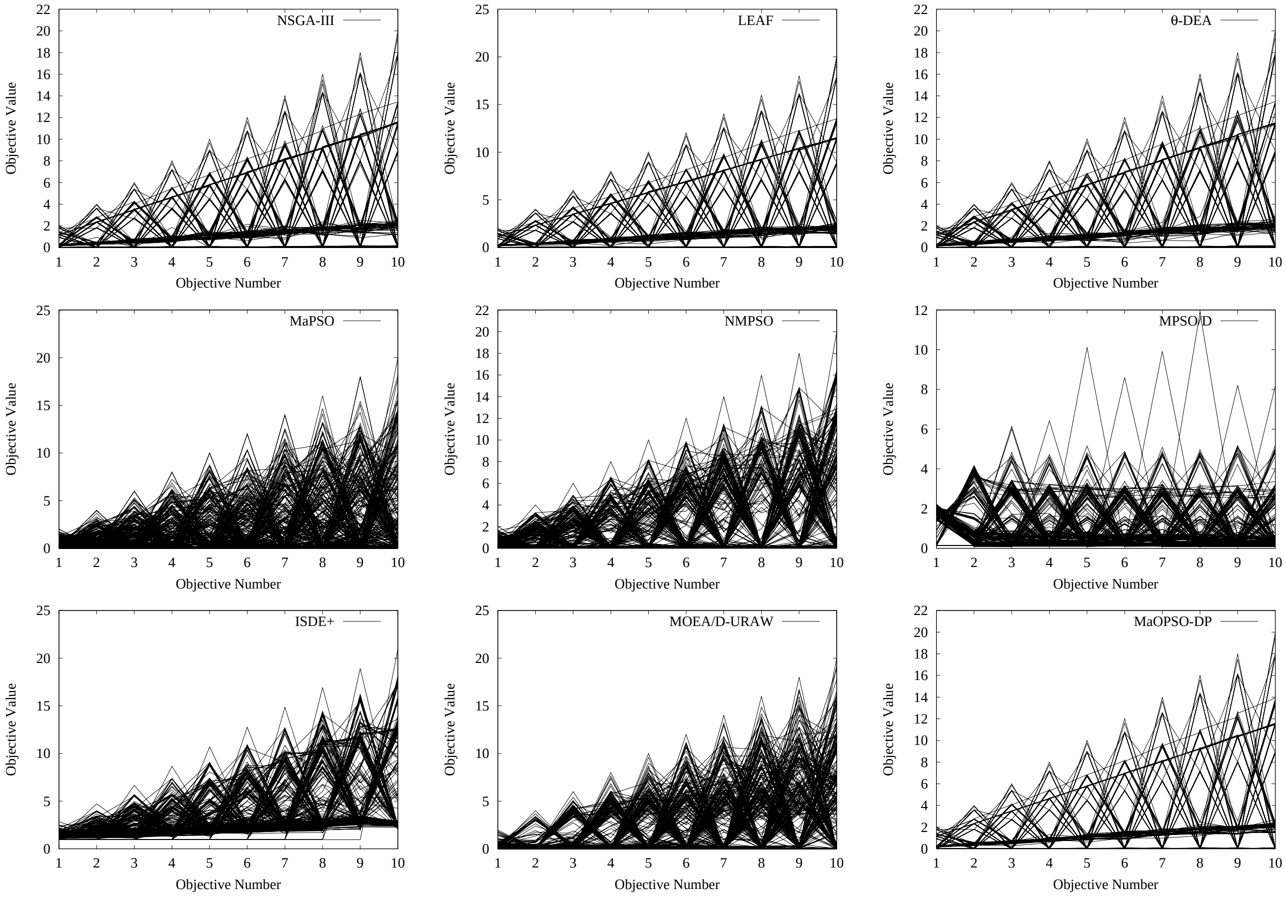

Figure 6: The value path of the obtained non-dominated solutions by the algorithms for 10-objective WFG7 problem.

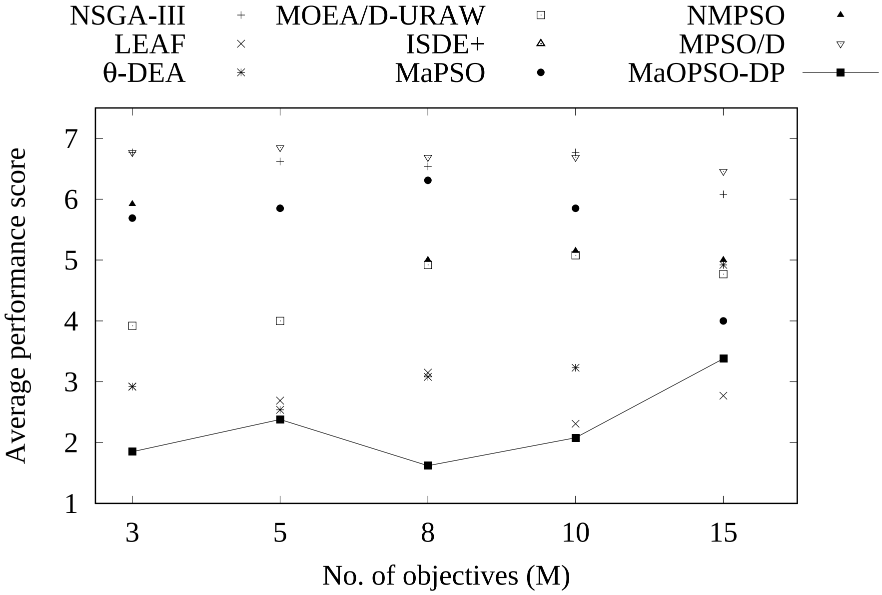

Figure 7: Average performance score based on the median IGD values over all objectives for different DTLZ and WFG problem instances. The solid line represents the performance of MaOPSO-DP.

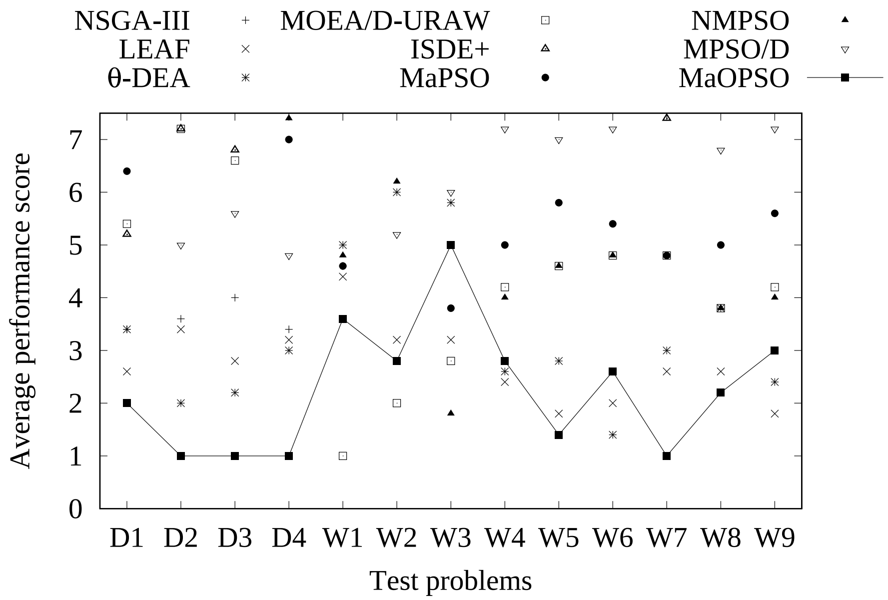

Figure 8: Average performance score based on the median IGD values over all DTLZ (Dx) and WFG (Wx) problem instances. The solid line represents the performance of MaOPSO-DP.

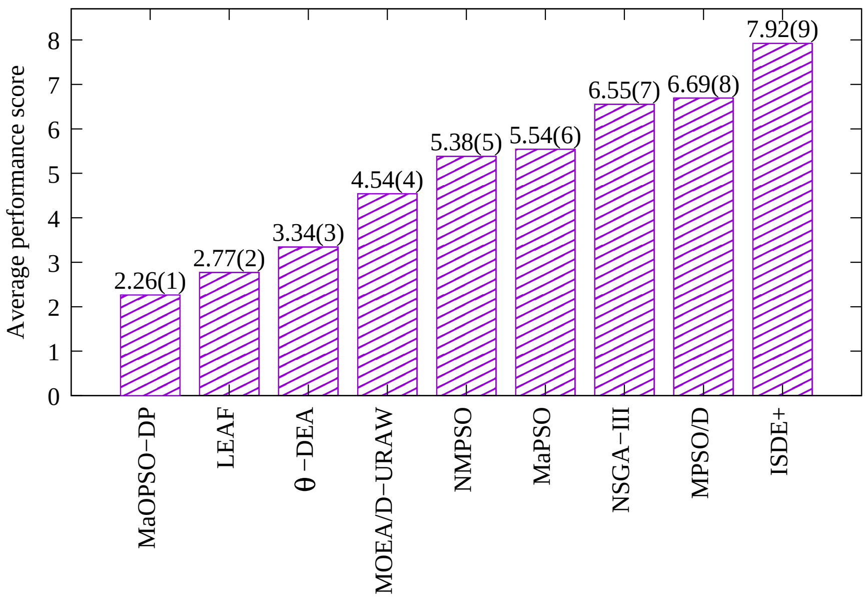

Figure 9: Ranking of algorithms over all DTLZ and WFG test instances. The smaller rank represents better performance.