Noname manuscript No.
(will be inserted by the editor)

Reference-Lines Steered Guide Assignment and Update for Pareto-based Many-Objective Particle Swarm Optimization

Deepak Sharma · Devang Agarwal · Santosh Kumar

Received: date / Accepted: date

Abstract The reference-lines-framework has been successfully used for developing efficient many-objective evolutionary algorithms. In this paper, the concepts and methodologies of such evolutionary algorithms are adapted in the parlance of multi-objective particle swarm optimization (MOPSO) for addressing the challenges of assigning and updating the global and local guides. The proposed algorithm, which is referred to as RMaOPSO, is developed via five modules using the framework so that a diverse set of guides can be selected to steer the search of MOPSO toward the Pareto-optimal front. The modules include global guide assignment, local and global guide update, line assignment to the guides and swarm, and evolutionary search for global guides. The proposed algorithm is tested on DTLZ and WFG test instances of 3-, 5-, 8-, 10- and 15- objectives. Results obtained from RMaOPSO show its efficacy over six multi-objective evolutionary and MOPSO algorithms from the literature.

Keywords MOPSO · Guide Assignment · Guide Update · Reference Lines · Many objective optimization

1 Introduction

The real-world optimization problems such as car-cab design [21], topology optimization of continuum structures [46, 50, 51], water resource management [47], bulldozer-blade parametric optimization [6] to name a few, are modeled with multiple objectives. A generic multi-objective optimization problem (MOP) can be written as

$$\min \ \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_M(\mathbf{x}))^T, \tag{1}$$

subject to $\mathbf{x} \in \Omega$, where $\mathbf{f} \in \mathbb{R}^M$ is the vector of conflicting objectives, $\mathbf{x} \in \mathbb{R}^N$ is the vector of design variables, and Ω is the search space.

Evolutionary and swarm algorithms are mainly preferred for solving equation (1) because these algorithms can generate Pareto-optimal (PO) solutions in one run. Among them, particle swarm optimization (PSO) has been the choice for many researchers because it is simple in concept, easy to implement, and computationally efficient as compared to other meta-heuristic algorithms [26, 48].

PSO has been used for solving MOP, which is generally referred to as MOPSO. In the most commonly used framework of MOPSO, a swarm is initialized by assigning random values to $\mathbf{x}_i(t)$ for each particle $i \in \mathbb{R}^N$ at generation $t = 0$. The initial velocity ($\mathbf{v}_i(t)$) of each particle i is either kept zero or chosen randomly. The archives of global guides (G_t) and local guides (L_t) are initialized. At the beginning, the non-dominated solutions from the swarm are copied to the global guide archive and $\mathbf{x}_i$'s for all particles are copied to the local guide archive. In a typical loop of generation, the global guide is chosen for each particle in the swarm. The velocity of each particle is then calculated as

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

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 $[0, 1]$, L_{ti} is the personal best of $i-$th particle at $t-$th generation,

Department of Mechanical Engineering, Indian Institute of Technology, Guwahati, Assam-781039, India
E-mail: dsharma@iitg.ac.in

and G_{ti} is the global best of $i-$th particle at $t-$th generation. The position of each 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 guides 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.

It can be observed from the framework that a set of challenges needs to be addressed for developing MOPSO [15, 24, 26, 63, 64]. The first challenge is updating the archive of global guides. While solving MOP, the number of non-dominated solutions can be more than the size of the archive. In this situation, only a diverse set of non-dominated solutions needs to be selected for which an efficient selection operator is needed. Once the archive is updated, the another challenge is to select an appropriate global guide for a particle. It is crucial because the selection of guide can change the flight direction of a particle that can affect the convergence and diversity of MOPSO. Another challenge is diversity loss among the particles of a swarm due to the fast convergence characteristic of PSO.

The above challenges have been addressed by keeping an external archive of non-dominated solutions and the global guides are updated for every particle in a swarm. For example, MOPSOs are developed using an adaptive grid procedure [13, 14], $\epsilon-$dominance method [54], distance-based ranking method [38], non-dominated sorting and crowding distance [40, 63], parallel coordinate system method [26], global margin ranking method [32], multi-objective gradient method [24], and circular crowding distance measure [12] for pruning the size of the archive and for selecting global guides. Decomposition-based MOPSOs use Tchebycheff function [43], PBI function [5], and crowding distance [34] for the same purpose.

In addition to the above challenges, MOPSOs encounter another big challenge when solving many-objective optimization problems (MaOPs), when $M \geq 3$ in equation (1). It is because many of the above MOPSOs can fail due to the reduction of selection pressure when almost all particles become non-dominated [44] along with the global guides in the archive. In this situation, the Pareto-ranking cannot differentiate particles/guides and selection procedure depends only on diversity preserving operator.

Efforts have been made in the literature for developing MOPSO for MaOPs. For example, MOPSOs are developed using the gradual Pareto-dominance [31], the weighted average ranking and distance-based ranking [39], Tchebycheff function and augmented scalarizing function (ASF) [55], ideal point and NWSUM method [7], reference points with $k-$means clustering [11], Tchebycheff function and crowding distance measure [25], non-dominated sorting and minimum angle approach [15]. The idea of association and niching of structured reference lines of NSGA-III [18] is also explored and the global guides are selected [23]. In another attempt, association and PBI distance are used for storing non-dominated solutions in an external archive [42]. Parallel cell coordinate system [27], scalar projection approach [59], balanceable fitness estimator [35], cooperative hybrid strategy [61] are few recent attempts of developing MOPSOs for MaOPs.

In this paper, the challenges described earlier for developing MOPSO are addressed using the reference-lines-based framework of NSGA-III in order to select a diverse set of solutions using the reference lines. The proposed algorithm, which is referred to as RMaOPSO, is developed by adapting the concepts and methodologies of NSGA-III in the parlance of MOPSO for improving its performance. Therefore, RMaOPSO is developed using the five modules with the following contributions.

- The first contribution of RMaOPSO is the development `Global_Guide_Assignment` module in which a global guide is assigned to each particle through an evenly distributed reference lines. These lines are drawn through the origin and the reference points generated on a unit hyperplane using Das and Dennis approach [16]. The first challenge is addressed by assigning the nearest non-dominated solution from the archive of global guides (G_t) to each reference line. The same solution becomes the global guide for all particles in a swarm that are associated with the same reference line.
- Another contribution is the update of global guides using the `Global_Guide_Update` module, which is developed using the concept of niching of NSGA-III. At this point, other challenges are addressed in which guides are assigned and updated using the structured reference lines that can help in maintaining diversity among the guides and can steer the search toward the PO front along these reference lines.
- Since the reference-lines-based framework is used, the local guide for each particle is updated using the `Local_Guide_Update` module. In this module, the update of local guide is performed by comparing the rank followed by the distance between the particle and local guide with their respective reference lines.
- Since the guides and particles are updated in each generation, they are ranked and associated together

with the reference lines using the `Line_Assignment` module.
- `Evolutionary_Search` is also coupled with the global guides of RMaOPSO so that these guides do not stuck in the local optima and can further improve the search of the algorithm. It is performed by using the simulated binary crossover and polynomial mutation operators [17].

RMaOPSO is tested using DTLZ and WFG test problems. Since both the sets of test problems are scalable, the objective instances of $M = \{3, 5, 8, 10, 15\}$ are used and the results are compared using the inverse generalized distance (IGD) and hypervolume (HV) indicators, and using the Wilcoxon test. The outcome of RMaOPSO is compared with six multi-objective evolutionary and MOPSO algorithms from the literature.

The paper is organized into five sections. Section 2 presents the approaches of Pareto-based MOPSO for MaOP. Section 3 presents the proposed RMaOPSO algorithm in which the framework and all modules are described. Section 4 presents the results obtained by RMaOPSO on DTLZ and WFG test problems, and the outcome of RMaOPSO is compared with six exiting algorithms. Section 5 concludes the paper with a future work.

2 Overview of Pareto-based Many-Objective PSO

A many-objective optimization problem is referred to as MaOP defined in equation (1), when $M > 3$ objectives. In the parlance of PSO, various many objective PSO algorithms have been developed. For example, MOPSO using gradual Pareto-dominance [31] for MaOPs is proposed in which ranking is given to each solution by calculating the degree of being dominated. The global guides for particles are selected through the fuzzy Pareto-dominance concept. MOPSOs are also developed using weighted average ranking and distance-based ranking [39] in which the global guides are selected using fitness proportionate selection and tournament selection. In another attempt, a distance-metric using Tchebycheff function and augmented scalarizing function [55] are used with MOPSO. The archive is maintained using these functions, and both global and local guides are updated.

Speed-constrained Multi-objective PSO (SMPSO) has also been extended for MaOP. An ideal point is used for pruning the size of the archive in which the non-dominated solution farthest from the ideal point is removed [7]. The global guide for each particle is selected through NWSum method [41]. Later, a set of reference points is used for pruning the size of the archive [9]. A multi-grid archiver approach is also coupled with SMPSO [11], which stores a diverse set of non-dominated solutions. The clusters are then made using $k-$means algorithm in the variable space of non-dominated solutions for generating new solutions.

An objective space decomposition approach is attempted that uses two-step search with MOPSO [25]. First, a swarm is divided into $M + 1$ groups, where M is the number of objective functions. The best solution from each group is selected as a global guide using Tchebycheff function. In step-2, these global guides are used for diversity. The archive size is controlled using crowding distance measure. In another approach, the objective space is decomposed using weight vectors and the clusters are made for every weight vector. The non-dominated solution, which makes the smallest angle to each weight vector, is selected for the archive [15]. In the recent attempt, MOPSO is modified using the decomposition-based approach for different ideal points for MaOP [45].

A reference-lines-based framework similar to NSGA-III [18] is used for MOPSO in which the archive size is controlled through the niche count of each reference line [23]. The non-dominated solutions closer to the reference lines are selected as the global guides for the swarm. On a similar framework, an external archive is maintained through the association of non-dominated solutions using PBI distance [42]. The non-dominated solution, which makes the maximum cosine angle, becomes the global guide for the particle. A bottleneck learning strategy for convergence and multiple swarm strategy for diversity is coupled with NSGA-III framework for updating the archive [36]. An evolutionary state estimation is used [57] for selecting two types of global guides for convergence and diversity. The reference-lines-based framework is used for updating the archive. The unary epsilon indicator for selecting the local guides, and the reference-vector framework for global guides and the archive are proposed by [37]. Using the reference-lines-based framework, a diversity preference approach is developed [53] for MOPSO in which diversity is preserved first by making clusters of solutions and then one solution from each cluster is selected using PBI method. The approach is used for updating the archives of global and local guides.

Some other recent approaches include an immune-based evolutionary strategy [64] for an archive update and pruning of the same is done using crowding distance measure. The global guide for a particle is selected randomly from the archive. Parallel cell coordinate system [27] is also used for MaOP in which two archives of global guides are kept separately for convergence and diversity. A scalar projection approach [59] is used for selecting the global guides from the swarm which is updated through fitness summation and $L2$ norm. A balanceable fitness estimator [35] consisting of convergence and diversity distances is proposed for updating the archive. The convergence distance is calculated with respect to the ideal point and the diversity distance is found using shift-based density estimation. Intuitionistic fuzzy dominance [60] is used with double search strategy for updating particle's position. The archive is updated using reference points with PBI distance metric. A concept of dominant different [33] is used for comparing solutions, selection of global and local guides, and for updating the archive.

Meanwhile, some comparative studies are also performed for ranking the solutions, guides assignment, and archive update. The study [30] presents comparison among the guide selection methods, such as random, crowding distance, WSum, NWSum, Sigma-method, and opposite method, which are coupled with SMPSO. It is found that NWSum and Sigma method evolve better results. Another study [8] presents comparison of various archiving methods, such as adaptive grid, crowding distance, dominating archive, adaptive $\epsilon-$approx archiving, adaptive $\epsilon-$Pareto archiving, multi-level grid archiving, random archiver, and unbounded archive. The unbounded archiver and $\epsilon-$Pareto archiving are found to be the best. The study [56] presents comparison of the methods that can differentiate non-dominated solutions for MaOP. Methods like favour relation, $k-$optimality, CDAS, crowding distance and average ranking, and sum ratios are considered. It is found that CDAS-based archiving method [10] is the most efficient among others.

In the literature, there is a considerable effort toward developing MOPSO for solving MaOP. Still, the performance of MOPSOs is not comparable with other multi-objective evolutionary algorithms for solving MaOPs. In this paper, an efficient MOPSO is developed using the reference-lines-framework through modules for better convergence and diversity. The proposed algorithm is described in the following section.

3 RMaOPSO: Proposed Many-objective MOPSO

The reference-lines-based framework is used to develop MOPSO, which is presented in Algo. 1. It is referred to as RMaOPSO that begins by initializing random swarm (P_t) of size N. At the same time, the archives to store local guides (L_t) and global guides (G_t) are kept empty. In order to assign L_t and G_t in subsequent generations, RMaOPSO adopts various features of reference lines-based-framework similar to NSGA-III [18] in which swarm P_t is evaluated, ranked, normalized and associated with a set of structured reference lines in Step 2 using `Line_Assignment` module of Algo. 1, which is discussed later. Initially, L_t and G_t archives are filled with the same swarm of P_t. An extreme vector ($\mathbf{e} \in \mathbb{R}^M$) [49, 52] is also initialized with the Nadir point, which is evaluated from the set of the non-dominated solutions from P_t. The vector $\mathbf{e}$ will be used later for normalization.

RMaOPSO enters into the standard loop of generation in Step 5. Since PSO is used for multi-objective optimization, there are always multiple global guides to steer the search of particles in the swarm. Generally, the non-dominated solutions are considered as global guides for which various algorithms have been adopted in the literature as discussed in Section 2. RMaOPSO proposes `Global_Guide_Assignment` module for assigning the global guides to particles in the swarm using the reference-lines-based framework. Once the guides are assigned, particle's velocity and position are updated in Steps 7 and 8. Thereafter, the `Line_Assignment` module is used on the combined population of the current swarm at $t-$th generation and the archives of global and local guides ($M_t = P_t \cup G_t \cup L_t$) to rank, normalize and associate them together with the set of structured reference lines. In Step 10, the `Local_Guide_Update` module is developed to update the local guides using the rank and association obtained in Step 9. Thereafter, the `Global_Guide_Update` module is developed to update the global guides in Step 11 from the combined popualtion (M_t). The niching concept of NSGA-III is adopted to develop this module. At the last, the `Evolutionary_Search` module is applied to the archive of the global guides in Step 12 in which crossover and mutation operators are used to update G_t. The modules under the generation-loop are repeated until t reaches to the maximum allowed generations (T). The archive of the global guides is then reported as the set of non-dominated solutions for the given optimization problem. In the following subsections, the reference-lines-based framework and the modules of Algo. 1 are discussed in detail.

3.1 Reference-Lines-based Framework

In this framework, a set of structured reference points is generated on a unit hyperplane using [16] approach in the objective space. In this approach, each objective axis is divided into p equal divisions that create

Algorithm 1 Reference-Lines-based Framework for MOPSO (RMaOPSO)

Input: $t = 1$, M : objectives, N : swarm size, H : reference points
Output: A set of non-dominated solutions (G_t)

1: Initialize random swarm (P_t), and initialize the archive of global guides ($G_t = \emptyset$) and local guides ($L_t = \emptyset$);
2: `Line_Assignment`(P_t);
3: Assign global guides ($G_t = P_t$) and local guides ($L_t = P_t$);
4: Compute extreme point: $\mathbf{e} = (e_1, e_2, \ldots, e_M)^T$ 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: `Global_Guide_Assignment`(G_t);
7: $W_t =$ Velocity_Update(P_t);
8: $P_t =$ Position_Update(W_t);
9: `Line_Assignment`($M_t = P_t \cup G_t \cup L_t$);
10: $L_t =$ `Local_Guide_Update`(P_t, L_t);
11: $G_t =$ `Global_Guide_Update`(M_t);
12: `Evolutionary_Search`(G_t);
13: $t = t + 1$
14: **end while**

$|H|$ number of reference points on a unit hyperplane as given in (4).

$$|H| = \binom{M + p - 1}{p}. \quad (4)$$

The set of structured reference points for $M = 3$ objective case is shown in Fig. 1. The figure on the top shows 21 reference points when $p = 5$ is taken. The two-layered approach [18] is shown at the bottom of the figure in which $p = 2$ for the outer layer and $p = 1$ for the inner layer are used. Although the two-layered approach is shown for 3-objective case, but it is mainly used for the instances of more than five objectives. It is because the number of reference points becomes very high for higher number of objectives as given by equation (4). The reference lines are then drawn, which pass from these reference points and the origin. These reference lines are used to associate each particle in the swarm. For example, a particle ($\mathbf{s}$) is associated to that reference line ($\mathbf{w}$) which is closest to it. The Euclidean distance, referred to as $dist(\mathbf{s}, \mathbf{w})$, between ($\mathbf{s}$) and ($\mathbf{w}$) is calculated using (5).

$$dist(\mathbf{s}, \mathbf{w}) = ||(\mathbf{s} - \mathbf{w}^T \mathbf{s} \mathbf{w} / ||\mathbf{w}||^2)||. \quad (5)$$

In the reference-lines-based framework, the convergence of a swarm is maintained through the non-dominated sorting [19] in which particles are sorted in different fronts based on their ranks. The diversity is maintained by associating particles with the reference lines. In the ideal condition when the global guides are assigned to each reference line and particles are associated with their closest reference lines, particles are expected to converge along those reference lines onto the PO front. Thus, the framework can evolve a well converge and diverse set of non-dominated solutions, which is maintained through the distribution of those structured reference lines.

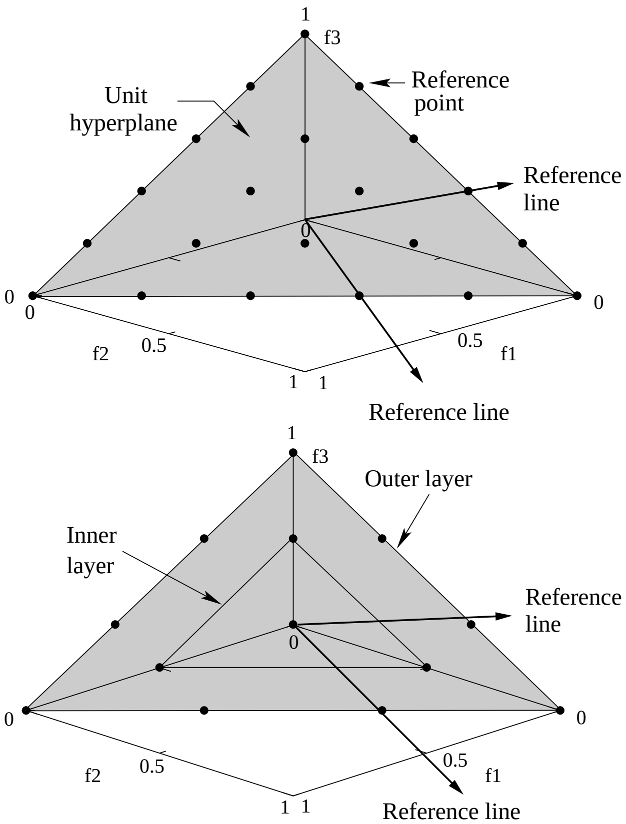

Fig. 1 A set of structured reference lines are drawn using Das and Dennis approach [16].

3.2 `Line_Assignment` Module: Steps 2 and 9 of Algo. 1

`Line_assignment` module is developed to quantify the solution based on its rank and association. Algo. 2 for this module presents three major steps in which solutions of R_t are ranked, normalized and associated with their nearest reference lines. For ranking, solutions of R_t are sorted in different fronts based on their ranks using the non-dominated sorting [19]. Thereafter in Step 2, R_t is normalized using Algo. 3 in which the ideal point ($\mathbf{z^I}$) of R_t is calculated in Step 1 using (6).

$$\mathbf{z^I} = (z_1^I, z_2^I, \ldots, z_M^I)^T : z_j^I = \min_{\mathbf{s} \in R_t} f_j(\mathbf{s}) \quad (6)$$

Algorithm 2 `Line_Assignment`(R_t)

1: $(F_1, F_2, \ldots) =$ Non-dominated sorting (R_t);
2: $\bar{R}_t =$ Normalize(R_t); %Using Algo. 3
3: Associate($\bar{R}_t$); %Using Algo. 4

In this case, the minimum of each objective is stored to translate R_t in Step 2 using (7).

$$\mathbf{f}^{'}(\mathbf{s}) = (f_1^{'}(\mathbf{s}), f_2^{'}(\mathbf{s}), \ldots, f_M^{'}(\mathbf{s}))^T : f_j^{'}(\mathbf{s}) = f_j(\mathbf{s}) - z_j^I, \ \forall \mathbf{s} \in R_t. \quad (7)$$

The ideal point of R_t is now translated to the origin. For normalizing R_t, the extreme vectors (Z) in each objective are calculated in Step 3 of Algo. 3 by minimizing the achievement scalarizing function given in (8).

$$Z = (\mathbf{z_1^e}, \mathbf{z_2^e}, \ldots, \mathbf{z_M^e}) : \mathbf{z}_j^e = \mathbf{f}^{'}(\mathbf{s}), \mathbf{s} : \min_{\mathbf{s} \in R_t} \left(\max_{i=1}^{M} f_i^{'}(\mathbf{s})/w_i \right). \quad (8)$$

Here, w_i is set to 1.0 when $i = j$ for $\mathbf{z}_j^e$ and 10^{-6} for rest of the objectives. These vectors in Z construct the $M-$dimensional hyperplane. This plane intersects each objective axis at a_j that is used for normalizing R_t as given in (9).

$$\bar{f}_j(\mathbf{s}) = \frac{f_j^{'}(\mathbf{s})}{a_j}, \ \forall j \in \{1, \ldots, M\}. \quad (9)$$

However, it has been found that the extreme vectors in Z can have duplicates that results in a degenerate case. In Step 6, the Nadir point ($\mathbf{Z^N}$) is computed from the set of non-dominated solutions of R_t if there is any duplicate in Z. Otherwise, the intercept $\mathbf{a} = (a_1, a_2, \ldots, a_M)^T$ on each objective axis is found in Step 8. At this stage, any intercept can become negative after solving a system of linear equations. In order to deal with this degenerate case, the Nadir point ($\mathbf{Z^N}$) is again computed in Step 10. If negative intercept is not found, the extreme vector $\mathbf{e}$ is updated in Step 13. Otherwise, the component of the extreme vector is compared with its corresponding Nadir point component and is updated accordingly in Step 18 [49]. The normalization of R_t is performed with the updated $\mathbf{e}$ vector in Step 21.

The last step in Algo. 2 is association, which is presented in Algo. 4. First, the reference lines ($\mathbf{w}$) are created using the structured reference points generated on the hyperplane, which pass through these points and the origin in Step 2 of Algo. 4. Thereafter, an Euclidean distance ($dist(\mathbf{s}, \mathbf{w})$) is calculated for each solution ($\mathbf{s} \in R_t$) to all reference lines ($\mathbf{w}$) using (5). The solution is then associated with the nearest reference line in Step 8 and its distance is stored in $d(\mathbf{s})$. This association will help RMaOPSO in assignment and updating the global guides.

Algorithm 3 Normalize(R_t) [49]

1: Compute ideal point using (6);
2: Translate objectives using (7);
3: Compute extreme points using (8);
4: Compute number of duplicate points (D) in Z;
5: **if** $D > 0$ **then**
6:    Compute Nadir point, $\mathbf{z^N} = (z_1^N, z_2^N, \ldots, z_M^N)^T$ from the set of the non-dominated solutions of R_t;
7: **else**
8:    Compute intercept $\mathbf{a} = (a_1, a_2, \ldots, a_M)^T$ from (Z) and assign flag= 0;
9:    **if** $a_i < 0$ **then**
10:       Compute Nadir point, $\mathbf{z^N} = (z_1^N, z_2^N, \ldots, z_M^N)^T$ from the set of the non-dominated solutions of R_t;
11:       flag = 1;
12:    **else**
13:       update $\mathbf{e} = \mathbf{a}$;
14:    **end if**
15: **end if**
16: **if** $D > 0$ or flag= 1 **then**
17:    **if** $z_j^N < e_j$, where $j \in \{1, \ldots, M\}$ **then**
18:       $e_j = z_j^N$
19:    **end if**
20: **end if**
21: $\bar{f}_j(\mathbf{s}) = f_j^{'}(\mathbf{s})/e_j, \forall \mathbf{s} \in R_t, \ \forall j \in \{1, \ldots, M\}$

Algorithm 4 Associate ($\bar{R}_t$) [18]

1: **for all** $\mathbf{r} \in H$ **do**
2:    Compute reference line $\mathbf{w}$ and $W = W \cup \mathbf{w}$; %Note that $|W| = H$.
3: **end for**
4: **for all** $\mathbf{s} \in \bar{R}_t$ **do**
5:    **for all** $\mathbf{w} \in W$ **do**
6:       Compute $dist(\mathbf{s}, \mathbf{w})$ using (5);
7:    **end for**
8:    $\pi(\mathbf{s}) = \mathbf{w}$: argmin $dist(\mathbf{s}, \mathbf{w})$; %Associates $\mathbf{s}$ to line $\pi(\mathbf{s})$
9:    $d(\mathbf{s}) = dist(\mathbf{s}, \pi(\mathbf{s}))$;%Stores minimum distance of $\mathbf{s}$ to $d(\mathbf{s})$
10: **end for**

3.3 `Global_Guide_Assignment` Module: Step 6 of Algo. 1

Using Algo. 2, all particles in the swarm and global guides in G_t are associated with their respective nearest reference lines. In this module, first the global guide is found for each reference line ($\mathbf{w}$) and the same global guide is then assigned to all particles, which are associated with the same reference line $\mathbf{w}$. This module is developed using Algo. 5 in which a set of non-dominated global guides associated with the reference line $\mathbf{w}$ is stored in $T_{\mathbf{w}}$ in Step 2. If $T_{\mathbf{w}} \neq \emptyset$, the nearest solution $\mathbf{s}$ from $T_{\mathbf{w}}$ is assigned as the global guide for the line $\mathbf{w}$ in Step 4. In case $T_{\mathbf{w}} = \emptyset$, the non-dominated solution $\mathbf{s}$ from G_t, which is closest to the line $\mathbf{w}$, is selected as the global guide for the line $\mathbf{w}$ in Step 6. It is noted that Step 6 shows a different approach than NSGA-III in which the line is deleted if any solution is not associated with it. However, since particles of the current swarm (P_t) can be associated with the line for which $T_{\mathbf{w}} = \emptyset$, Step 6 helps RMaOPSO to find the nearest non-dominate global guide.

Algorithm 5 Global_Guide_Assignment(G_t)

1: **for all** $\mathbf{w} \in W$ **do**
2: $\quad T_{\mathbf{w}} = \mathbf{s} : \pi(\mathbf{s}) = \mathbf{w},\ \mathbf{s} \in G_t$ and $\mathbf{s}$ is non-dominated solution;
3: $\quad$ **if** $T_{\mathbf{w}} \neq \emptyset$ **then**
4: $\quad\quad GB_{\mathbf{w}} = GB_{\mathbf{w}} \cup \{\mathbf{s} : \mathrm{argmin}_{\mathbf{s} \in T_{\mathbf{w}}} d(\mathbf{s})\}$; %$GB_{\mathbf{w}}$ refers to the global best solution for line $\mathbf{w}$
5: $\quad$ **else**
6: $\quad\quad GB_{\mathbf{w}} = GB_{\mathbf{w}} \cup \{\mathbf{s} : \mathrm{argmin}\, d(\mathbf{s}),\ \mathbf{s} \in G_t$ and $\mathbf{s}$ is non-dominated solution$\}$;
7: $\quad$ **end if**
8: **end for**

3.4 Velocity and Position Update

The velocity and the position of a particle are updated using equations (2) and (3). The most commonly used approach for setting w, c_1, and c_2 parameters is to sample them randomly in their respective ranges. Since this approach did not work well with RMaOPSO, an adaptive approach is used in which the parameters are changed as given in equation (10).

$$\begin{aligned} w &= 0.9 \times (1 - t/T), \\ c_1 &= 2.5 \times (1 - t/T), \\ c_2 &= 2.5 \times (1 - t/T), \end{aligned} \tag{10}$$

where t is the current generation, and T is the maximum allowed generations.

3.5 `Local_Guide_Update` Module: Step 10 of Algo. 1

Once particles updated their positions in Step 8 and associated with reference lines in Step 9 of Algo. 1, the archive of the local guides is updated. Algo. 6 presents the local guide update rules, which depend on the rank and distance of a particle to the associated line. If the rank of a particle is better than its local guide, the local guide is updated in Step 3. If rank is the same, the local guide is updated based on the smaller Euclidean distance of a particle from the associated reference line in Step 6.

3.6 `Global_Guide_Update` Module: Step 11 of Algo. 1

The `global_guide_update` module is developed by adopting the niching mechanism of NSGA-III so that

Algorithm 6 `Local_Guide_Update`(P_t, L_t)

1: **for all** $i \in N$ **do**
2: $\quad$ **if** (rank of $P_t^i <$ rank of L_t^i) **then**
3: $\quad\quad$ Update $L_t^i = P_t^i$;%P_t^i and L_t^i are $i-$th swarm of P_t and L_t
4: $\quad$ **else if** (rank is same) **then**
5: $\quad\quad$ **if** ($d(P_t^i) < d(L_t^i)$) **then**
6: $\quad\quad\quad$ Update $L_t^i = P_t^i$; %Based on distance found in Step 8 of Algo. 4
7: $\quad\quad$ **end if**
8: $\quad$ **end if**
9: **end for**

Algorithm 7 `Global_Guide_Update`(R_t)

1: Classify solutions in different fronts based on the ranks obtained earlier by `Line_Assignment` module;
2: Initialize $S_t = \emptyset$ and $i = 1$;
3: **while** $|S_t| \leq N$ **do**
4: $\quad S_t = S_t \cup F_i$ and $i = i + 1$;
5: **end while**
6: **if** ($|S_t| = N$) **then**
7: $\quad G_t = S_t$, Stop;
8: **else**
9: $\quad G_t = \cup_{i=1}^{l-1} F_i$; %Inclusion of fronts till last but one.
10: $\quad$ Compute niche count of each reference line ($\mathbf{w}$) such that $\rho_{\mathbf{w}} = \sum_{\mathbf{s} \in S_t / F_l} ((\pi(\mathbf{s}) = j) \ ?\ 1 \ :\ 0)$;
11: $\quad$ **while** ($|G_t| \leq H$) **do**
12: $\quad\quad$ Find a line which has the least niche count. In case of multiple lines having minimum niche count, choose one of them ($\mathbf{w}$) at random;
13: $\quad\quad I_{\mathbf{w}} = \{\mathbf{s} : \pi(\mathbf{s}) = \mathbf{w},\ \mathbf{s} \in F_l\}$; %F_l is the last front to be used for filling G_t
14: $\quad\quad$ **if** ($I_{\mathbf{w}} \neq \emptyset$) **then**
15: $\quad\quad\quad$ **if** ($\rho_{\mathbf{w}} = 0$) **then**
16: $\quad\quad\quad\quad G_t = G_t \cup (\mathbf{s} : \mathrm{argmin}_{\mathbf{s} \in I_{\mathbf{w}}} d(\mathbf{s}))$; %Copy the solution closest to the line $\mathbf{w}$
17: $\quad\quad\quad$ **else**
18: $\quad\quad\quad\quad G_t = G_t \cup \mathrm{random}(I_{\mathbf{w}})$;
19: $\quad\quad\quad$ **end if**
20: $\quad\quad\quad \rho_{\mathbf{w}} = \rho_{\mathbf{w}} + 1,\ F_l = F_l \setminus \mathbf{s}$;
21: $\quad\quad$ **else**
22: $\quad\quad\quad$ Remove line $\mathbf{w}$
23: $\quad\quad$ **end if**
24: $\quad$ **end while**
25: **end if**

a set of good solutions can be stored in the archive of global guides. Algo. 7 presents the global guide update for Step 11 of Algo. 1 in which the combined population (M_t) is sent and for Step 4 of Algo. 8 in which ($G_t \cup \hat{G}_t$) is sent. Since ranking and association have already been done by `Line_Assignment` module, solutions of R_t are classified into different fronts based on their ranks in Step 1 of Algo. 7. Solutions of R_t are then copied front-wise into S_t till its size is more than N in Step 4. If the size of S_t is the same as N, all solutions of S_t are copied into G_t in Step 7 and the module is terminated. Otherwise, solutions in the fronts are copied to G_t, excluding the last front (F_l) solutions in Step 9. Thereafter, the niche count of each reference line is computed in Step

10. This count signifies the number of solutions associated with a line. If the niche count of a line is relatively lower than other lines, it signifies that the region around this line is less crowded. It means that a solution can be chosen to update the archive of global guides. The same procedure is followed in Step 12 to find the line which has the minimum niche count. If the line has no associated solution from S_t/F_l, its niche becomes zero. For this line, a solution from F_l, which is nearest to it, is copied to G_t in Step 16. In case, the niche count of the line is non-zero, any random solution from F_l, which is associated with it, is copied to G_t in Step 18. When a solution from F_l is copied to G_t, the niche count is updated and the selected solution is removed from F_l so that a distinct solution can be copied. If a line has no associated solution from S_t/F_l and also from F_l, this line is then removed for further consideration in Step 22. This update for the archive of the global guides ensures selection of a diverse set of the best-ranked solutions into G_t, which is useful for velocity update of particles and also to report the non-dominated solutions.

3.7 `Evolutionary_Search` Module: Step 12 of Algo. 1

An evolutionary search is performed on the archive of the global guides (G_t) in every generation so that the global guides do not stuck to any local optima and can improve further to steer the search of the swarm toward the PO front. Algo. 8 presents the four major steps in which crossover is performed using SBX operator and mutation is performed using polynomial mutation operator [17]. The new set of global guides ($\hat{G}_t$) along with the current global guides (G_t) are then ranked, normalized and associated together with the lines using `Line_Assignment`($G_t \cup \hat{G}_t$) module in Step 3 of Algo. 8. Thereafter, the global guides are selected through `Global_Guide_Update`($G_t \cup \hat{G}_t$) module in Step 4.

Algorithm 8 Evolutionary_Search(G_t)

1: $\bar{G}_t =$ crossover(G_t); %by using SBX crossover operator
2: $\hat{G}_t =$ mutate($\bar{G}_t$); %by using Polynomial mutation operator
3: `Line_Assignment`($G_t \cup \hat{G}_t$) using Algo. 2;
4: `Global_Guide_Update`($G_t \cup \hat{G}_t$) using Algo. 7;

3.8 Computational Complexity

The computational complexity of RMaOPSO is similar to NSGA-III since it involves all the key operations, such as non-dominated ranking, normalization, association, and niching. However, the non-dominated ranking and association are performed thrice in one generation. Therefore, the worst-case computational complexity of one generation of RMaOPSO is either the non-dominated ranking ($O(3 \times N^2 log^{M-2} N)$) or association ($O(3 \times N^2 M)$), whichever is larger.

4 Results and Discussion

In this section, RMaOPSO is tested on DTLZ [20] and WFG [28] problem instances having $M = \{3, 5, 8, 10, 15\}$ objectives. The number of variables for DTLZ problems is $n = M + k - 1$, where $k = 5$ is kept fixed for DTLZ1, and $k = 10$ is kept the same for DTLZ2-4 problems. Similarly, $n = k + l$ is used for WFG1-9 problems in which $k = 2 \times (M - 1)$ is the position-related variable and $l = 20$ is kept fixed for the distance-related variable. These test problems are chosen because they have different characteristics, such as DTLZ1 is linear and multi-modal; DTLZ2 is concave; DTLZ3 is concave and multi-modal; DTLZ4 is concave and biased; WFG1 is mixed and biased; WFG2 is convex, disconnected, multi-modal and non-separable; WFG3 is linear, degenerate and non-separable; WFG4 is concave and multi-modal; WFG5 is concave and deceptive; WFG6 is concave and non-separable; WFG7 is concave and biased; WFG8 is concave, biased and non-separable; WFG9 is concave, biased, multi-modal, deceptive and non-separable. These characteristics pose challenges for algorithms to converge to the PO front.

Two statistical indicators, such as inverse generalized distance (IGD) and hypervolume (HV) are used to assess the performance of RMaOPSO with respect to the existing multi-objective evolutionary algorithms and MOPSOs. IGD indicator measures convergence and diversity of a set of the obtained non-dominated solutions (P) with respect to the PO solutions (Q^*). It is calculated using (11) in which $d(q_i^*, p_j) = ||q_j^* - p_i||^2$ is the Euclidean distance in the objective space, $|Q^*|$ and $|P|$ are the cardinality of Q^* and P, respectively.

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

HV indicator measures the size of the objective space dominated by the solutions in P and bounded by $\mathbf{z}^r$. It is given in (12), where $VOL(.)$ represents the Lebesgue measure, and $\mathbf{z}^r = (z_1^r, \ldots, z_M^r)^T$ is the reference point, which is dominated by all PO solutions. Larger is the HV value, better is the quality of P for approximating the PO front. 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. The HV values presented in this paper are normalized between $[0, 1]$ by dividing $z = \prod_{i=1}^{M} z_i^r$. Both the indicators are determined by normalizing P, except for DTLZ1.

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

RMaOPSO is compared with six algorithms from the literature, that are, NSGA-III [18][1], SPEA/R [29] [2], VaEA [58], dMOPSO [3] [62], SMPSO [40], and MaPSO [59]. NSGA-III is chosen because RMaOPSO is developed using its reference-lines-based framework. However, RMaOPSO has adopted the concepts of NSGA-III for developing procedure for the global and local guides' assignment and update. SPEA/R is chosen because it uses $k-$layered reference direction search method in which it emphasis first on diversity followed by convergence. VaEA is chosen because it also uses the reference-lines-based framework; however, the environmental selection is performed using the maximum-vector-angle approach and worst-elimination principle. SMPSO and dMOPSO are chosen because these algorithms are established MOPSOs which have been tested successfully on many multi-objective optimization problems. MaPSO is the recently published MOPSO and its working principle was discussed in Section 2.

All the algorithms are run 20 times with different initial populations or swarms and their results are compared. The Wilcoxon signed-rank test at 5% significance level is performed to compare the outcome of RMaOPSO with the existing algorithms. The population size for all algorithms is chosen based on the number of reference points calculated using (4). The details are given in Table 1. These algorithms are terminated based on the number of generations, which are presented in Table 2.

For a fair comparison, the algorithm parameters are kept same. The SBX and polynomial mutation operators are used as evolutionary search. The probability of crossover is kept 1.0, and the probability of mutation is $p_m = 1/n$. The distribution index for SBX operator is $\eta_c = 30$, and the distribution index for polynomial mutation operator is $\eta_m = 20$. For all MOPSOs, the parameters (w, c_1, c_2) are sampled randomly from their respective ranges, such as $w \in [0.1, 0.5]$, and $c_1, c_2 \in [1.5, 2.5]$. For SPEA/R, the archive size is set same as the population size, and the number of $k-$layers for 3-, 5-, 8-, 10-, and 15-objectives for all problems is $k = 7, 8, 5, 6$, and 3, respectively. The population size is determined as $N = 4 \times ceil(((M \times k \times (k+3)/2)+1)/4)$. For MaPSO, the parameters $K = 3$ and $\theta_{max} = 0.5$ are kept fixed.

Table 1 Number of reference points and corresponding population sizes for the 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

Table 2 Maximum number of generations for 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

4.1 Performance on DTLZ Problems

The performance of RMaOPSO is tested on scalable DTLZ1-4 problems in this section, and its outcome is tested using the IGD and HV indicators. Table 3 presents the best, median, and worst values of IGD indicator for each objective of DTLZ problems. The gray cells represent the best IGD value for each row among the algorithms. Here, smaller IGD value is better. It can be seen that RMaOPSO shows better IGD values in 44 out of 60 rows, which is the highest in number. The outcome from the Wilcoxon test is also shown for each instance in the same table. The symbol '+' indicates that RMaOPSO is significantly better than the corresponding algorithm. Similarly, the symbols '$-$' and '=' indicate significantly worse and equivalent performance of RMaOPSO with respect to the corresponding algorithm, respectively. At the bottom of the table, the collective outcome $(+/=/-)$ from the Wilcoxon test is shown. It can be seen that RMaOPSO outperforms all the algorithms on all instances of DTLZ1-4 problems.

Table 4 presents the best, median and worst HV indicator values for DTLZ problems. The gray cells again represent the best HV values for each row. Here, larger

[1] NSGA-III code developed by [49] is used, which is available in the public domain.

[2] The codes of SPEA/R, VaEA and MaPSO are provided by the authors.

[3] The source codes of dMOPSO and SMPSO are obtained from the jmetal framework [22].

HV value is better. RMaOPSO shows the best HV values in 33 out of 48 rows, which is the highest in number. It can be seen again at the bottom of the table that RMaOPSO again outperforms all the algorithms based on the outcome of Wilcoxon test.

The obtained non-dominated solutions for 3- and 10-objective DTLZ3 problem are shown here because DTLZ3 is concave and multi-modal multi-objective optimization problem. The plots are generated corresponding to the run of median IGD value. Fig. 2 shows the obtained non-dominated solutions for 3-objective problem from all the algorithms. It can be seen that RMaOPSO, NSGA-III, SPEA/R, and MaPSO are able to converge to the PO front of DTLZ3. However, the evenness in the distribution of solutions can be seen with RMaOPSO and NSGA-III. SMPSO and dMOPSO generates the solutions quite close to the PO front but the distribution of solutions is not as even as RMaOPSO. VaEA is only the algorithm which fails to converge to the PO front.

Fig. 3 shows the value path of the obtained non-dominated solutions for 10-objective DTLZ3 problem. It can be seen that RMaOPSO and NSGA-III generate a converged and well-distributed set of solutions, where the rest of the algorithms fail to converge to the PF. Except for the tenth objective, VaEA is also converged to the PO front.

4.2 Performance on WFG Problems

Now, RMaOPSO is tested on the WFG test problems and its outcome is compared with other algorithms based on the IGD and HV values, and on Wilcoxon test outcome. Table 5 presents the best, median, and worst IGD values obtained from the algorithms for WFG1-9 problem instances. The gray cells represent the best IGD values among the algorithm for each row. Here, smaller IGD value is better. For WFG problems, a scattered distribution of gray cells can be seen in which RMaOPSO shows better IGD values in 39 rows out of 135, which is the highest in number. The table also shows the outcome of Wilcoxon test for individual instances and the cumulative outcome is presented at the bottom of the table. It can be seen that RMaOPSO outperforms all MOPSOs and shows better performance over SPEA/R and VaEA. RMaOPSO shows an equivalent performance with NSGA-III.

Table 6 presents the statistical HV indicator values for WFG1-9 problem instances. The gray cells again represent the better HV value. Here, larger HV value is better. Again, a scatter gray cells can be seen in which RMaOPSO shows better HV values in 31 out of 108 rows, which is the highest in number. The outcome of Wilcoxon test is shown for each problem instance and the cumulative performance can be seen at the bottom of the table. It can be observed that RMaOPSO outperforms dMOPSO and SMPSO. It shows better performance than SPEA/R and VaEA and an equivalent performance with MaPSO. Based on HV values, NSGA-III shows slightly better outcome of Wilcoxon test over RMaOPSO. The main reason is the frequent jumping of the particles out of the bounds which are then brought back to the bounds.

The non-dominated solutions obtained from all algorithms are shown in Fig. 4 for 3-objective WFG6 problem in the normalized objective space. The plots are generated corresponding to the run of median IGD value. It can be seen that RMaOPSO and NSGA-III are converged to the PO front and the obtained solutions are evenly spread over the PO front. The rest of the algorithms are little far from the PO front. Fig. 5 shows value path plots of algorithms for 10-objective WFG6 problem. Except for dMOPSO, all algorithms have generated the extreme solutions in each objective. However, RMaOPSO and NSGA-III show a better distribution of the obtained non-dominated solutions over the PO front.

4.3 Average Performance

Since a large set of problem instances is solved in which none of the algorithms come out to be the clear winner, an average performance score of the algorithms for different objectives and problems is calculated [26, 53, 57, 60]. The score is calculated by comparing the median IGD values obtained in Table 3 and Table 4 and then, a rank is assigned to every algorithm for each instance. The smaller average performance score represents better performance. Fig. 6 shows the average performance score of all algorithms for different objectives. It can be clearly seen that RMaOPSO shows the best performance in all objective instances cumulatively. Fig. 7 shows the average performance score over different DTLZ and WFG problems. RMaOPSO shows the best performance for DTLZ2-4 problems, and WFG4-6 and WFG8 problems. RMaOPSO is the second best in DTLZ1, WFG3, WFG7, and WGF9 problems. Based on the above average performance scores, the ranking of the algorithms is calculated and shown in Fig. 8. It can be seen that RMaOPSO emerges as the best algorithm among the others. It is slightly better than NSGA-III but outperforms the others.

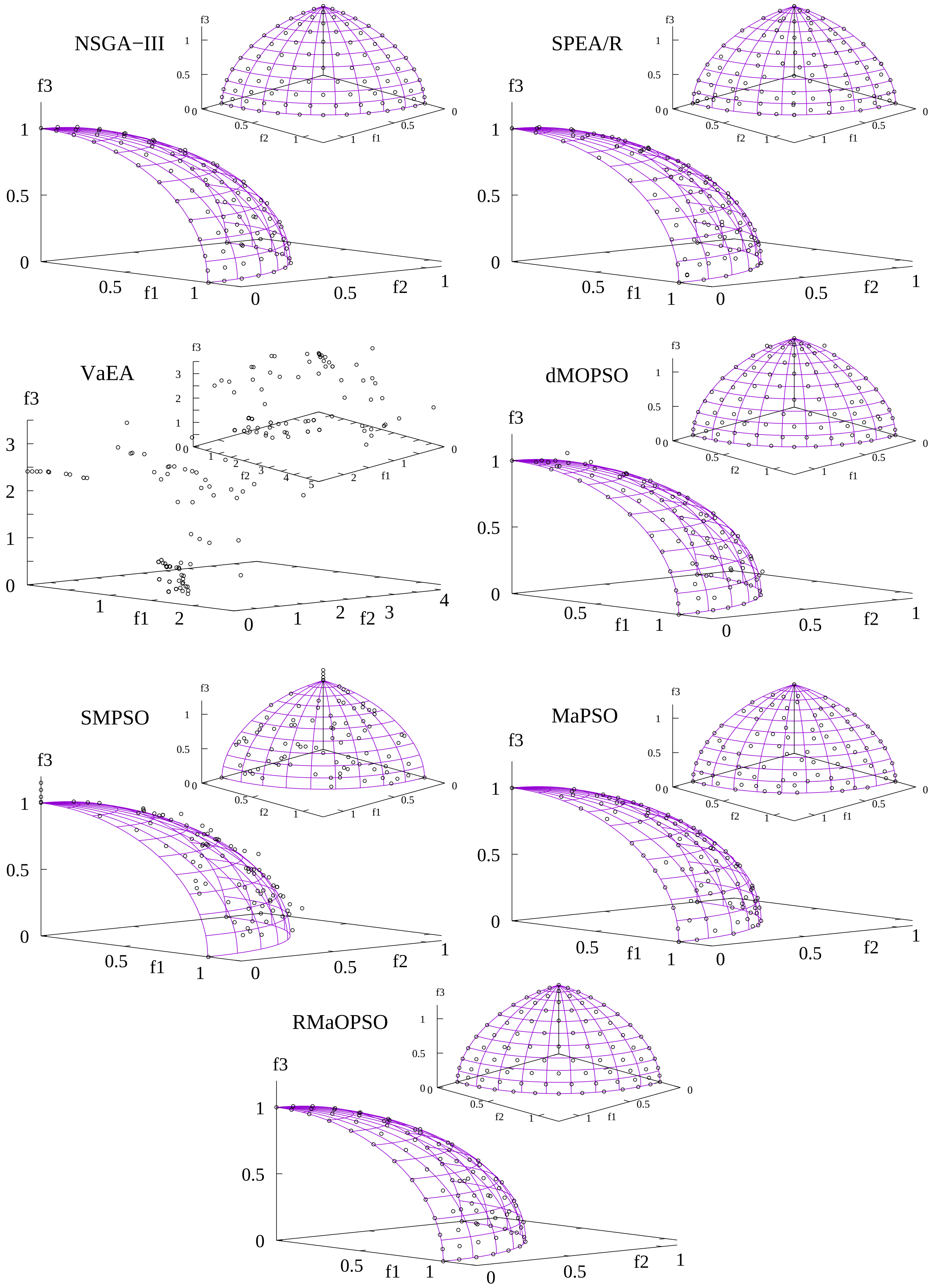

Fig. 2 Obtained non-dominated solutions by the algorithms for 3-objective DTLZ3 problem.

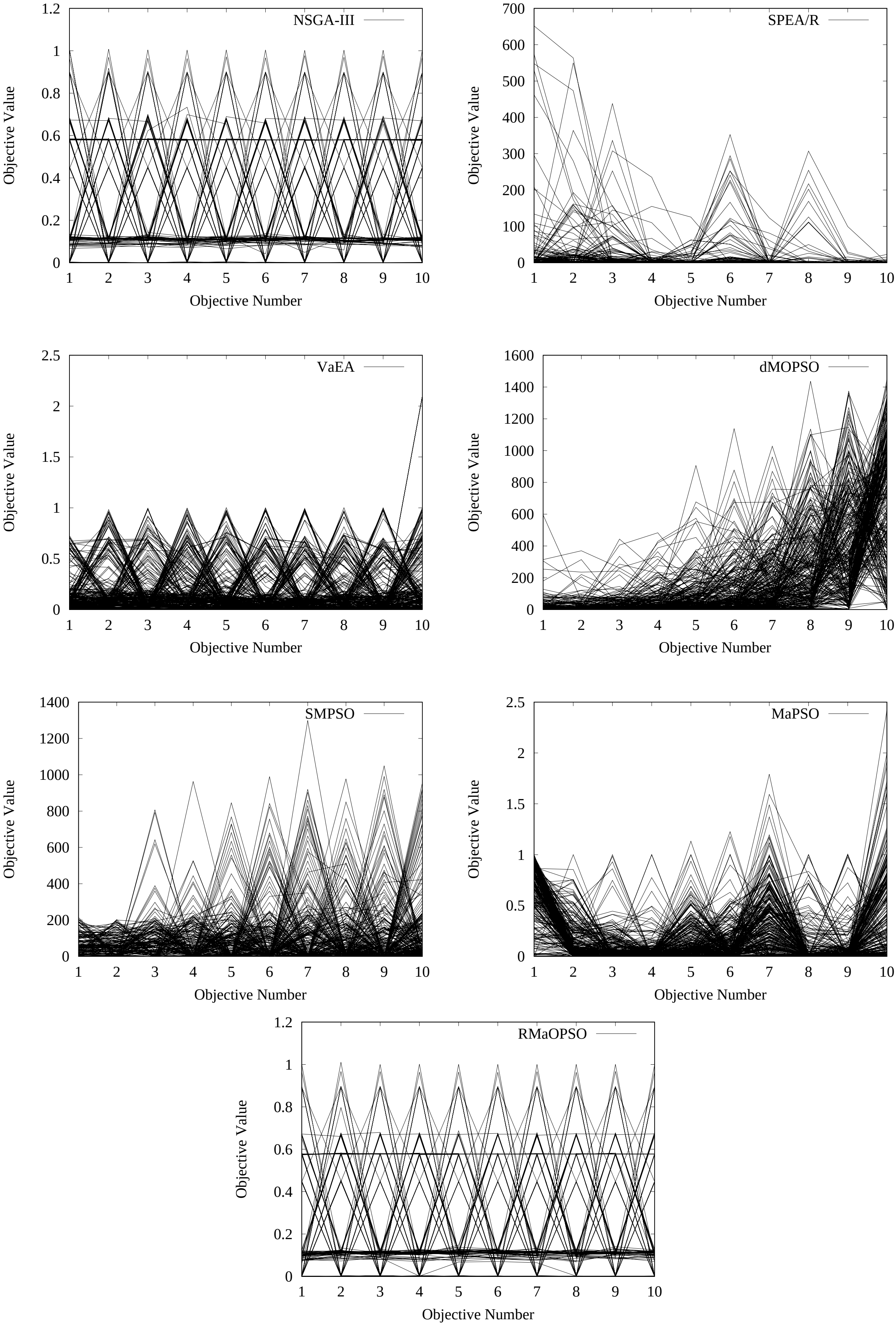

Fig. 3 The value path of the obtained non-dominated solutions by the algorithms for 10-objective DTLZ3 problem.

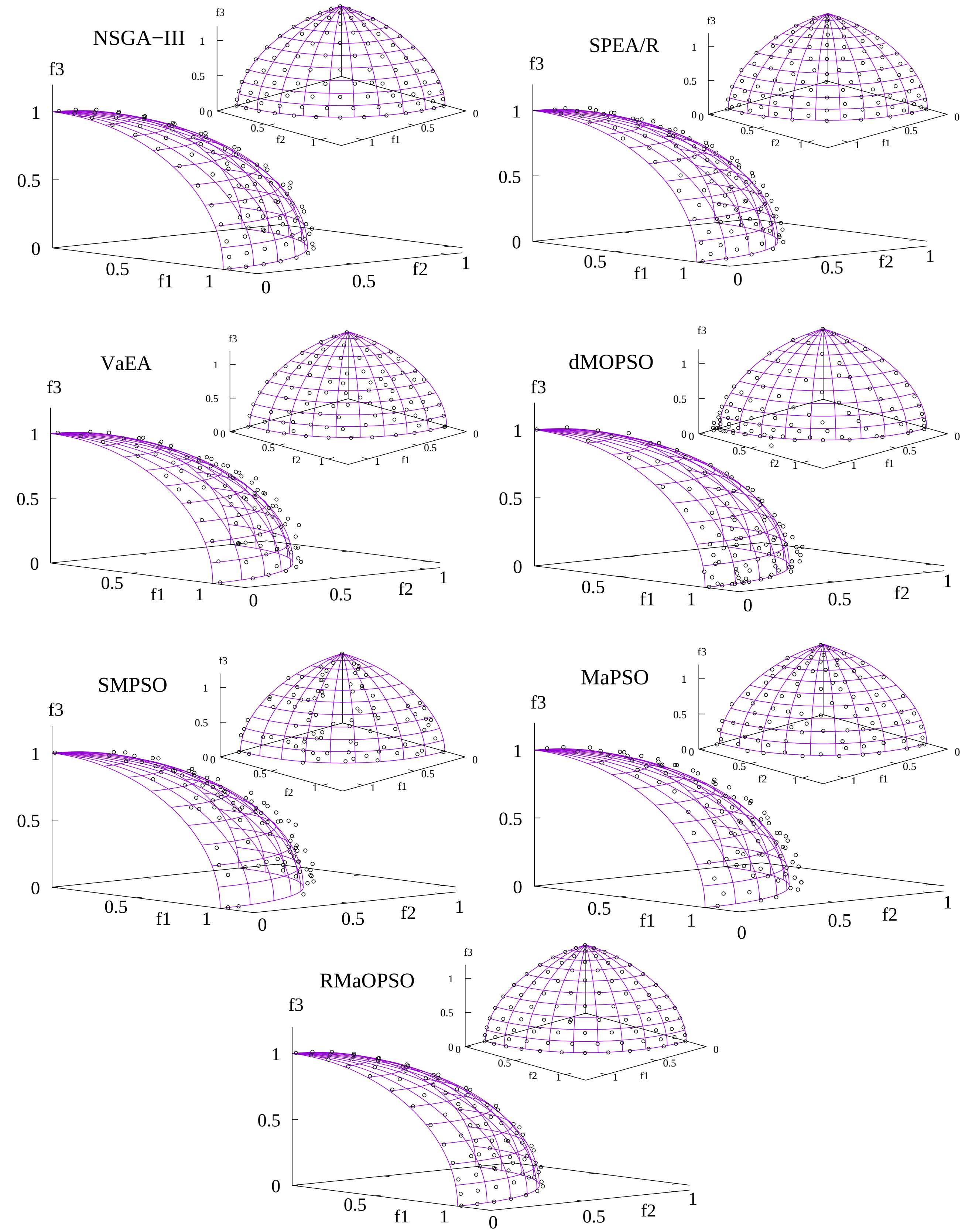

Fig. 4 Obtained non-dominated solutions by the algorithms for 3-objective WFG6 problem in the normalized objective space.

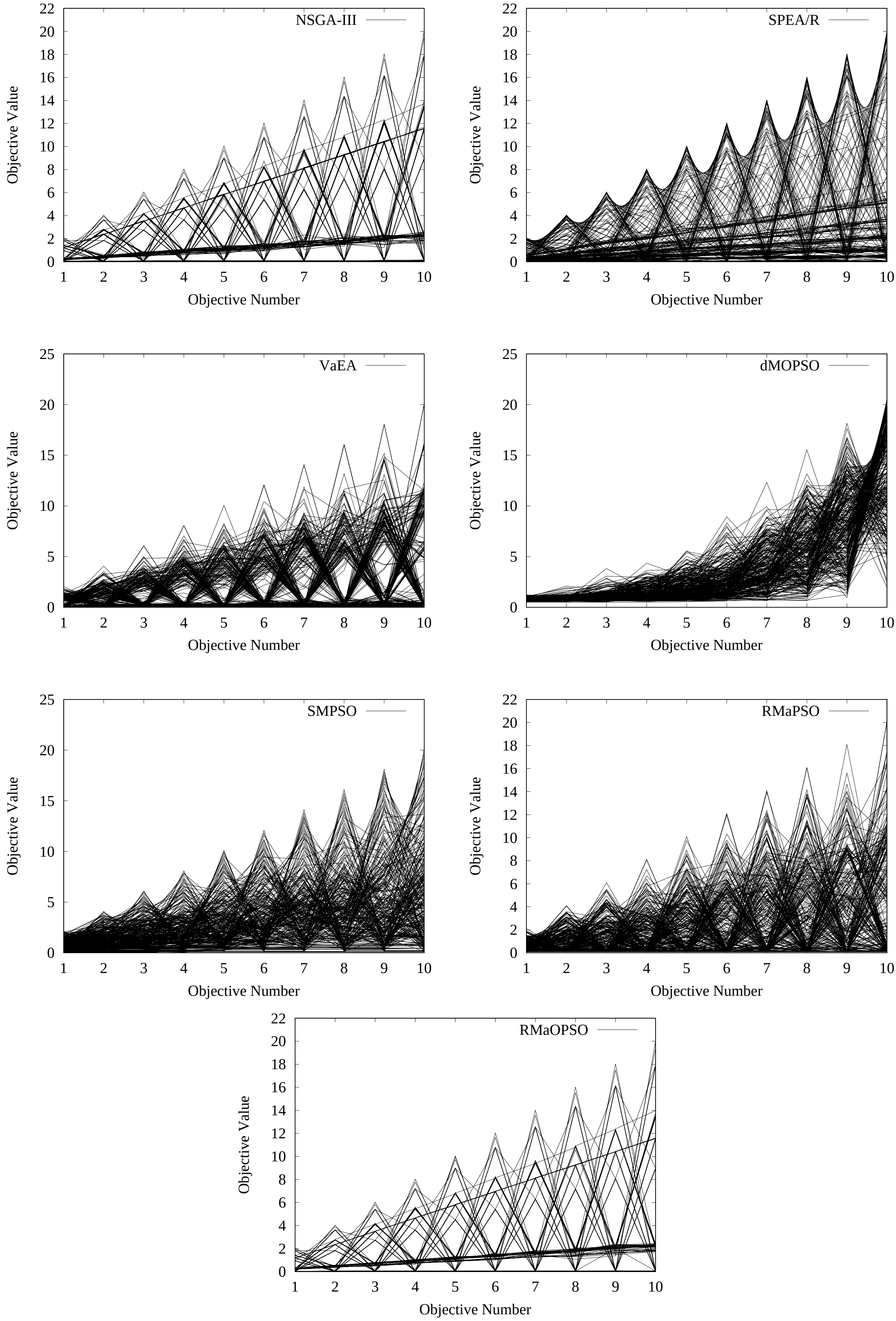

Fig. 5 The value path of the obtained non-dominated solutions by the algorithms for 10-objective WFG6 problem.

Table 3 Best, median and worst IGD values obtained by RMaOPSO and other algorithms on DTLZ instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
DTLZ1	3	3.510E-04	4.447E-03	1.280E-02	2.398E-02	2.900E-02	2.135E-02	**1.238E-04**
		1.536E-03 +	2.138E-02 +	4.899E-02 +	2.826E-02 +	3.197E-02 +	2.331E-02 +	**2.127E-04**
		5.787E-03	9.910E-02	4.039E-01	4.943E-02	3.485E-02	2.494E-02	**3.073E-04**
	5	4.962E-04	1.517E-02	1.898E-02	7.658E+00	9.169E-02	6.422E-02	**3.706E-04**
		7.431E-04 +	4.038E-02 +	3.401E-02 +	3.939E+01 +	1.094E-01 +	7.465E-02 +	**4.209E-04**
		1.246E-03	1.323E-01	6.372E-02	6.338E+01	1.343E-01	1.069E-01	1.870E-03
	8	2.175E-03	6.328E-02	1.933E-02	2.565E+01	1.344E-01	1.178E-01	**1.408E-03**
		3.582E-03 −	1.490E-01 +	2.722E-02 −	4.972E+01 +	3.216E-01 +	1.370E-01 +	7.047E-02
		6.645E-02	6.193E-01	**4.981E-02**	6.342E+01	7.597E+00	1.925E-01	7.767E-02
	10	2.279E-03	4.146E-02	2.214E-02	2.258E+01	1.564E-01	1.249E-01	**1.400E-03**
		2.583E-03 =	1.018E-01 +	2.939E-02 =	4.367E+01 +	5.223E-01 +	1.564E-01 +	9.153E-03
		9.297E-02	2.897E-01	**3.972E-02**	7.083E+01	1.304E+01	1.969E-01	7.405E-02
	15	1.922E-03	2.409E-01	4.884E-02	2.202E+01	3.361E-01	1.736E-01	**1.825E-03**
		2.853E-03 −	4.356E-01 +	5.362E-02 =	4.257E+01 +	5.630E-01 +	1.959E-01 +	5.433E-02
		4.324E-03	2.969E+00	5.938E-02	6.759E+01	1.778E+01	2.163E-01	2.249E-01
DTLZ2	3	1.045E-03	3.125E-03	8.292E-03	5.914E-02	7.385E-02	4.903E-02	**5.017E-04**
		1.270E-03 +	5.074E-03 +	1.485E-02 +	6.153E-02 +	7.758E-02 +	5.543E-02 +	**6.948E-04**
		2.870E-03	1.128E-02	2.572E-02	6.600E-02	8.210E-02	5.736E-02	**8.574E-04**
	5	3.058E-03	9.941E-03	1.334E-02	5.562E-01	2.922E-01	1.589E-01	**2.198E-03**
		4.481E-03 +	1.308E-02 +	1.606E-02 +	6.324E-01 +	3.624E-01 +	1.675E-01 +	**2.493E-03**
		1.128E-02	2.120E-02	2.064E-02	6.777E-01	4.056E-01	1.891E-01	**2.964E-03**
	8	1.152E-02	2.307E-02	2.830E-02	8.489E-01	6.857E-01	3.167E-01	**6.210E-03**
		1.293E-02 +	2.849E-02 +	3.524E-02 +	9.295E-01 +	8.782E-01 +	3.352E-01 +	**8.456E-03**
		1.691E-02	3.342E-02	4.904E-02	9.814E-01	1.074E+00	3.547E-01	**1.213E-02**
	10	1.142E-02	2.455E-02	2.270E-02	9.557E-01	8.171E-01	3.652E-01	**7.532E-03**
		1.279E-02 +	2.893E-02 +	3.838E-02 +	1.011E+00 +	1.157E+00 +	3.833E-01 +	**8.563E-03**
		1.486E-02	3.689E-02	4.143E-02	1.073E+00	1.342E+00	3.945E-01	**1.256E-02**
	15	1.052E-02	4.607E-02	3.692E-02	1.168E+00	1.474E+00	4.247E-01	**6.598E-03**
		1.428E-02 +	5.477E-02 +	6.588E-02 +	1.268E+00 +	1.635E+00 +	4.592E-01 +	**8.586E-03**
		1.758E-02	7.102E-02	1.485E-01	1.309E+00	2.134E+00	4.797E-01	8.236E-02
DTLZ3	3	8.723E-04	6.758E-03	1.955E-01	5.232E-02	7.106E-02	4.860E-02	**3.125E-04**
		3.991E-03 +	3.334E-02 +	1.052E+00 +	5.428E-02 +	7.516E-02 +	5.646E-02 +	**4.300E-04**
		9.847E-03	2.413E-01	4.125E+00	6.004E-02	8.325E-02	2.002E+00	**4.667E-03**
	5	2.174E-03	7.925E-02	2.226E-02	3.504E+02	2.716E-01	1.718E-01	**6.627E-04**
		3.675E-03 +	2.060E-01 +	1.936E-01 +	4.927E+02 +	3.844E-01 +	2.153E-01 +	**8.090E-04**
		1.014E-02	3.801E-01	5.687E-01	5.875E+02	4.890E-01	3.813E-01	**5.444E-03**
	8	1.256E-02	3.811E-01	8.289E-02	3.818E+02	3.710E+00	3.762E-01	**8.602E-03**
		2.444E-02 −	2.296E+00 +	8.487E-01 +	5.636E+02 +	6.459E+01 +	4.392E-01 +	5.039E-02
		5.287E-02	4.493E+00	1.145E+00	6.772E+02	1.321E+02	5.171E-01	3.975E-01
	10	8.236E-03	3.882E-01	5.758E-02	3.513E+02	4.408E+01	4.699E-01	**4.480E-03**
		1.069E-02 =	6.790E-01 +	3.287E-01 +	4.572E+02 +	7.938E+01 +	5.020E-01 +	**7.291E-03**
		1.929E-02	4.395E+00	1.169E+00	6.132E+02	1.009E+02	5.985E-01	4.270E-01
	15	1.121E-02	5.443E+00	6.600E-02	3.434E+02	1.533E+02	5.234E-01	**3.936E-03**
		1.766E-02 +	1.207E+01 +	1.280E+00 +	5.223E+02 +	2.075E+02 +	6.156E-01 +	**6.160E-03**
		3.671E-02	3.338E+01	1.301E+00	6.283E+02	2.247E+02	7.536E-01	2.272E-01
DTLZ4	3	**3.113E-04**	4.001E-04	7.698E-03	8.080E-02	6.947E-02	5.124E-02	3.153E-04
		3.918E-04 =	1.837E-03 +	2.267E-01 +	1.108E-01 +	7.344E-02 +	5.555E-02 +	**3.721E-04**
		5.314E-01	**4.983E-03**	9.503E-01	1.794E-01	7.985E-02	5.766E-02	9.503E-01
	5	3.641E-04	2.182E-03	1.641E-02	7.434E-01	1.974E-01	1.463E-01	**3.357E-04**
		4.334E-04 +	4.001E-03 +	1.939E-01 +	9.713E-01 +	2.325E-01 +	1.533E-01 +	**3.975E-04**
		5.072E-04	9.661E-03	3.947E-01	1.135E+00	2.813E-01	1.578E-01	**4.881E-04**
	8	2.541E-03	7.315E-03	3.326E-02	9.863E-01	4.266E-01	2.532E-01	**2.469E-03**
		3.442E-03 +	9.148E-03 +	2.380E-01 +	1.137E+00 +	5.405E-01 +	2.728E-01 +	**3.073E-03**
		5.319E-03	1.220E-02	6.228E-01	1.334E+00	6.252E-01	2.862E-01	**4.012E-03**
	10	3.578E-03	6.907E-03	4.088E-02	8.838E-01	5.627E-01	2.813E-01	**2.998E-03**
		4.228E-03 +	8.963E-03 +	1.843E-01 +	1.017E+00 +	6.641E-01 +	2.990E-01 +	**3.659E-03**
		5.174E-03	1.191E-02	3.770E-01	1.145E+00	7.548E-01	3.047E-01	**4.490E-03**
	15	5.257E-03	9.225E-03	1.226E-01	1.068E+00	1.118E+00	2.512E-01	**4.961E-03**
		7.298E-03 =	1.115E-02 +	2.898E-01 +	1.184E+00 +	1.268E+00 +	2.773E-01 +	7.697E-03
		9.578E-03	1.475E-02	9.067E-01	1.303E+00	1.486E+00	3.023E-01	1.048E-02
(+/=/-)		13/4/3	20/0/0	17/2/1	20/0/0	20/0/0	20/0/0	

5 Conclusions

RMaOPSO has been developed in this paper for updating the archive of global guides and assigning them to particles using the reference-lines-based framework. The main objective was to select an appropriate global guide for each particle so that many-objective optimization problems can be solved efficiently. Therefore, a set of structured reference lines was used to assign and update guides in every generation for each particle in a swarm. In order to achieve the objective, five modules were developed for improving convergence and diversity of RMaOPSO. These modules included `Global_Guide_Assignment`, `Global_Guide_Update`, `Local_Guide_Update`, `Line_Assignment`, and `Evolutionary_Search`. The proposed RMaOPSO

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

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
DTLZ1	3	9.73656E-01 9.73423E-01 $^{+}$ 9.72470E-01	9.73376E-01 9.70047E-01 $^{+}$ 9.44695E-01	9.70545E-01 9.57151E-01^{+} 4.78116E-01	9.70931E-01 9.60109E-01^{+} 9.40745E-01	9.68321E-01 9.66712E-01^{+} 9.65165E-01	9.72251E-01 9.71936E-01^{+} 9.71078E-01	**9.73676E-01** **9.73663E-01** **9.73624E-01**
	5	9.98982E-01 9.98977E-01 $^{+}$ 9.98963E-01	9.97918E-01 9.97159E-01 $^{+}$ 9.91088E-01	9.98560E-01 9.98216E-01^{+} 9.97263E-01	9.70931E-01 9.60109E-01^{+} 9.40745E-01	9.94571E-01 9.90411E-01^{+} 9.77792E-01	9.98277E-01 9.97662E-01^{+} 9.94801E-01	**9.98986E-01** **9.98984E-01** **9.98981E-01**
	8	9.99974E-01 **9.99971E-01**$^{-}$ 9.99896E-01	9.99786E-01 9.96402E-01 $^{+}$ 4.83954E-01	9.99824E-01 9.99570E-01^{+} 9.98418E-01	9.70931E-01 9.60109E-01^{+} 9.40745E-01	9.94571E-01 9.90411E-01^{+} 9.77792E-01	9.99884E-01 9.99807E-01^{+} 9.99051E-01	**9.99979E-01** 9.99914E-01 **9.99906E-01**
	10	9.99998E-01 9.99997E-01 $^{=}$ 9.99988E-01	9.99974E-01 9.99923E-01 $^{+}$ 9.74683E-01	9.99932E-01 9.99864E-01^{+} 9.99656E-01	9.70931E-01 9.60109E-01^{+} 9.40745E-01	9.94571E-01 9.90411E-01^{+} 9.77792E-01	9.99994E-01 9.99972E-01^{+} 9.99870E-01	**9.99999E-01** **9.99999E-01** **9.99994E-01**
DTLZ2	3	9.26692E-01 9.26640E-01 $^{+}$ 9.26584E-01	9.26671E-01 9.26565E-01 $^{+}$ 9.26319E-01	9.25552E-01 9.24474E-01^{+} 9.22430E-01	9.23564E-01 9.22346E-01^{+} 9.21754E-01	9.19128E-01 9.17490E-01^{+} 9.14333E-01	9.26472E-01 9.26058E-01^{+} 9.25503E-01	**9.26729E-01** **9.26704E-01** **9.26684E-01**
	5	9.90505E-01 9.90474E-01 $^{+}$ 9.90418E-01	9.86881E-01 9.86822E-01 $^{+}$ 9.86721E-01	9.90383E-01 9.90274E-01^{+} 9.90117E-01	7.69201E-01 7.49667E-01^{+} 7.09305E-01	9.58619E-01 9.32828E-01^{+} 9.06122E-01	9.89463E-01 9.89135E-01^{+} 9.87376E-01	**9.90529E-01** **9.90511E-01** **9.90491E-01**
	8	9.99337E-01 9.99328E-01 $^{+}$ 9.99315E-01	9.98713E-01 9.98658E-01 $^{+}$ 9.98475E-01	9.99325E-01 9.99312E-01^{+} 9.99286E-01	7.48924E-01 7.08707E-01^{+} 6.75727E-01	8.46623E-01 7.11107E-01^{+} 4.29473E-01	9.99055E-01 9.98856E-01^{+} 9.98726E-01	**9.99347E-01** **9.99340E-01** **9.99325E-01**
	10	9.99919E-01 9.99917E-01 $^{+}$ 9.99916E-01	9.99764E-01 9.99744E-01 $^{+}$ 9.99721E-01	9.99919E-01 9.99876E-01^{+} 9.99872E-01	7.71284E-01 7.49630E-01^{+} 7.17320E-01	8.21622E-01 6.08050E-01^{+} 4.17720E-01	9.99825E-01 9.99801E-01^{+} 9.99783E-01	**9.99920E-01** **9.99919E-01** **9.99917E-01**
DTLZ3	3	9.26593E-01 9.25882E-01 $^{+}$ 9.24558E-01	9.26316E-01 9.24816E-01 $^{+}$ 8.99569E-01	3.29451E-03 6.83835E-03^{+} 1.11064E-01	9.26429E-01 9.25997E-01^{+} 9.17906E-01	9.20859E-01 9.19780E-01^{+} 9.17765E-01	9.26684E-01 9.26131E-01^{+} 4.40325E-03	**9.26775E-01** **9.26695E-01** **9.25775E-01**
	5	9.90525E-01 9.90469E-01 $^{+}$ 9.90135E-01	9.86510E-01 9.82919E-01 $^{+}$ 9.72970E-01	9.90161E-01 9.80306E-01^{+} 8.19713E-01	9.26429E-01 9.25997E-01^{+} 9.17906E-01	9.59236E-01 9.21241E-01^{+} 7.24203E-01	9.88524E-01 9.86276E-01^{+} 9.78306E-01	**9.90584E-01** **9.90563E-01** **9.90330E-01**
	8	9.99334E-01 9.99264E-01 $^{=}$ **9.99166E-01**	9.86510E-01 9.82919E-01 $^{=}$ 9.72970E-01	9.99072E-01 6.58526E-01^{+} 5.03707E-01	9.26429E-01 9.25997E-01^{+} 9.17906E-01	9.59236E-01 9.21241E-01^{+} 7.24203E-01	9.98633E-01 9.98135E-01^{+} 9.97371E-01	**9.99349E-01** **9.99278E-01** 9.95700E-01
	10	**9.99922E-01** 9.99918E-01 $^{=}$ **9.99909E-01**	9.86510E-01 9.82919E-01 $^{=}$ 9.72970E-01	9.99856E-01 9.96896E-01^{+} 5.07743E-01	9.26429E-01 9.25997E-01^{+} 9.17906E-01	9.59236E-01 9.21241E-01^{+} 7.24203E-01	9.99661E-01 9.99590E-01^{+} 9.99363E-01	9.99922E-01 **9.99920E-01** 9.99259E-01
DTLZ4	3	9.26777E-01 9.26733E-01 $^{=}$ 7.98533E-01	**9.26883E-01** **9.26823E-01**$^{-}$ **9.26724E-01**	9.26598E-01 9.14404E-01^{+} 5.00000E-01	9.21032E-01 9.18264E-01^{+} 9.13141E-01	9.21013E-01 9.20364E-01^{+} 9.18995E-01	9.26525E-01 9.26163E-01^{+} 9.25349E-01	9.26782E-01 9.26731E-01 4.99993E-01
	5	9.90593E-01 9.90578E-01 $^{=}$ **9.90571E-01**	9.87093E-01 9.87067E-01 $^{+}$ 9.87043E-01	**9.90628E-01** 9.88983E-01^{+} 9.71855E-01	7.25329E-01 5.23790E-01^{+} 3.76585E-01	9.84946E-01 9.83696E-01^{+} 9.80806E-01	9.90341E-01 9.90211E-01^{+} 9.90066E-01	9.90592E-01 **9.90580E-01** 9.90570E-01
	8	9.99365E-01 **9.99364E-01**$^{=}$ **9.99363E-01**	9.98833E-01 9.98828E-01 $^{+}$ 9.98811E-01	9.99380E-01 9.98877E-01^{+} 9.87270E-01	6.60632E-01 5.80339E-01^{+} 4.15375E-01	9.85841E-01 9.73631E-01^{+} 9.43560E-01	**9.99390E-01** 9.99339E-01^{+} 9.99309E-01	9.99365E-01 9.99364E-01 9.99363E-01
	10	9.99924E-01 **9.99923E-01**$^{=}$ **9.99923E-01**	9.99793E-01 9.99792E-01 $^{+}$ 9.99790E-01	**9.99925E-01** 9.99918E-01^{+} 9.99454E-01	8.36730E-01 7.06959E-01^{+} 5.60297E-01	9.87990E-01 9.64856E-01^{+} 9.52353E-01	9.99924E-01 9.99917E-01^{+} 9.99913E-01	9.99924E-01 9.99923E-01 9.99923E-01
(+/=/-)		8/7/1	13/2/1	16/0/0	16/0/0	16/0/0	16/0/0	

was tested on many-objective instances of DTLZ and WFG problems and the outcome was compared with six existing multi-objective evolutionary and MOPSO algorithms. Based on the obtained results using the IGD and HV indicators, and Wilcoxon test, it can be concluded that RMaOPSO emerges as the best among the chosen set of algorithms. Especially, RMaOPSO outperformed all three MOPSO algorithms.

An observation can be made that RMaOPSO is still unable to perform well in many instances for WFG problems. The primary reason is the frequent jumping of the particles out of the bound which are again brought back to the bound. Therefore, RMaOPSO still needs an efficient velocity update for better performance. Moreover, the concepts like diversity over dominance approaches [52], line prioritized environmental selection [49], etc. can be brought into the parlance of MOPSO to further improve the convergence and diversity among the particles and guides. Furthermore, RMaOPSO can be hybridized with other heuristic algorithms [1, 2, 3, 4] for better performance.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Abualigah L, Yousri D (2021) Advances in sine cosine algorithm: A comprehensive survey. Artifi-

Table 5 Best, median and worst IGD values obtained by RMaOPSO and other algorithms on WFG instances with different number of objectives. Best performances are highlighted in bold face with gray background.

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
WFG1	3	3.555E-01	4.010E-01	**1.544E-01**	5.180E-01	5.357E-01	1.815E-01	3.678E-01
		3.692E-01 −	4.230E-01 +	**1.830E-01** −	5.235E-01 +	5.390E-01 +	2.441E-01 −	3.868E-01
		3.803E-01	4.317E-01	**2.345E-01**	5.296E-01	5.420E-01	3.000E-01	3.976E-01
	5	3.994E-01	4.304E-01	3.126E-01	1.113E+00	5.720E-01	**2.557E-01**	4.129E-01
		4.042E-01 −	4.582E-01 +	3.803E-01 −	1.184E+00 +	5.782E-01 +	**3.263E-01** −	4.207E-01
		4.108E-01	4.662E-01	4.301E-01	1.210E+00	5.876E-01	**3.981E-01**	4.291E-01
	8	3.646E-01	3.464E-01	3.214E-01	1.134E+00	6.061E-01	**2.464E-01**	3.452E-01
		4.198E-01 −	4.010E-01 −	3.332E-01 −	1.168E+00 +	6.101E-01 +	**3.254E-01** −	4.755E-01
		4.492E-01	6.962E-01	**3.507E-01**	1.207E+00	6.183E-01	4.288E-01	6.149E-01
	10	3.151E-01	2.934E-01	2.806E-01	1.084E+00	6.024E-01	**1.998E-01**	2.237E-01
		3.648E-01 −	3.192E-01 =	2.924E-01 −	1.132E+00 +	6.095E-01 +	**2.257E-01** −	4.661E-01
		4.425E-01	5.884E-01	**3.066E-01**	1.165E+00	6.121E-01	3.988E-01	5.610E-01
	15	4.319E-01	3.326E-01	4.091E-01	1.091E+00	6.081E-01	**3.260E-01**	3.347E-01
		4.435E-01 +	6.362E-01 +	4.141E-01 +	1.110E+00 +	6.104E-01 +	**3.443E-01** −	3.689E-01
		4.899E-01	6.421E-01	4.187E-01	1.142E+00	6.170E-01	**3.629E-01**	4.390E-01
WFG2	3	1.769E-02	**1.745E-02**	4.304E-02	7.598E-02	6.709E-02	4.398E-02	1.815E-02
		2.082E-02 =	**2.050E-02** =	4.989E-02 +	8.019E-02 +	7.376E-02 +	4.785E-02 +	2.187E-02
		9.766E-02	9.922E-02	1.111E-01	8.788E-02	8.275E-02	5.373E-02	**2.682E-02**
	5	5.846E-02	**4.746E-02**	7.106E-02	3.437E-01	1.382E-01	7.156E-02	5.813E-02
		5.990E-02 =	**4.972E-02** −	7.732E-02 +	3.800E-01 +	1.560E-01 +	7.499E-02 +	6.089E-02
		1.629E-01	**5.139E-02**	1.736E-01	4.485E-01	1.673E-01	7.919E-02	6.406E-02
	8	8.938E-02	**6.683E-02**	1.110E-01	4.258E-01	1.676E-01	1.155E-01	1.511E-01
		1.452E-01 −	**7.197E-02** −	1.203E-01 −	4.595E-01 +	1.988E-01 =	1.216E-01 −	1.876E-01
		2.313E-01	2.049E-01	2.143E-01	5.131E-01	2.449E-01	**1.312E-01**	2.694E-01
	10	1.221E-01	**6.505E-02**	1.896E-01	4.590E-01	1.484E-01	1.875E-01	1.944E-01
		2.017E-01 =	**7.483E-02** −	2.042E-01 −	4.953E-01 +	1.952E-01 −	2.066E-01 −	2.115E-01
		3.199E-01	2.403E-01	**2.211E-01**	5.230E-01	2.262E-01	2.227E-01	3.205E-01
	15	3.069E-01	3.381E-01	4.713E-01	8.832E-01	1.830E-01	**1.187E-02**	6.990E-01
		6.278E-01 −	1.094E+00 +	5.550E-01 −	9.337E-01 +	2.091E-01 −	**2.191E-02** −	8.137E-01
		7.115E-01	1.128E+00	7.147E-01	9.740E-01	2.424E-01	**4.289E-02**	1.035E+00
WFG3	3	**1.788E-02**	3.485E-02	3.387E-02	2.770E-02	4.503E-02	2.465E-02	2.300E-02
		2.271E-02 −	4.155E-02 +	4.451E-02 +	3.802E-02 +	8.472E-02 +	2.923E-02 =	3.077E-02
		3.068E-02	6.476E-02	5.633E-02	5.093E-02	1.037E-01	3.721E-02	4.959E-02
	5	4.575E-02	9.751E-02	6.074E-02	2.224E-01	1.274E-01	4.339E-02	**3.234E-02**
		5.855E-02 =	1.149E-01 +	8.556E-02 +	2.515E-01 +	1.616E-01 +	**5.463E-02** =	6.123E-02
		8.149E-02	1.386E-01	1.585E-01	2.738E-01	1.980E-01	**6.321E-02**	1.175E-01
	8	**5.078E-02**	2.677E-01	7.622E-02	2.208E-01	1.164E-01	1.035E-01	7.925E-02
		6.922E-02 −	4.197E-01 +	1.108E-01 =	2.761E-01 +	2.069E-01 +	1.645E-01 +	9.846E-02
		1.342E-01	6.170E-01	1.882E-01	2.984E-01	2.527E-01	2.385E-01	2.635E-01
	10	5.661E-02	1.056E-01	7.716E-02	2.241E-01	6.873E-02	1.206E-01	**5.262E-02**
		7.434E-02 −	2.262E-01 +	1.723E-01 +	2.634E-01 +	1.893E-01 +	1.762E-01 +	9.190E-02
		1.176E-01	5.069E-01	2.733E-01	2.934E-01	2.387E-01	2.183E-01	1.202E-01
	15	**2.631E-02**	3.815E-01	5.268E-02	2.443E-01	1.340E-01	1.383E-01	3.088E-02
		9.104E-02 −	4.350E-01 +	2.071E-01 +	2.881E-01 +	2.066E-01 =	2.069E-01 +	1.356E-01
		2.671E-01	5.666E-01	2.794E-01	3.233E-01	**2.500E-01**	2.956E-01	4.494E-01
WFG4	3	4.735E-03	7.735E-03	5.244E-02	7.515E-02	9.717E-02	5.588E-02	**4.478E-03**
		6.065E-03 =	9.383E-03 +	5.576E-02 +	7.708E-02 +	1.008E-01 +	6.121E-02 +	6.328E-03
		7.093E-03	1.134E-02	6.114E-02	8.219E-02	1.082E-01	6.711E-02	8.813E-03
	5	**1.629E-02**	1.994E-02	1.601E-01	4.639E-01	1.948E-01	1.702E-01	1.741E-02
		2.119E-02 =	2.158E-02 +	1.681E-01 +	5.097E-01 +	2.117E-01 +	1.765E-01 +	**1.983E-02**
		2.988E-02	2.502E-02	1.774E-01	5.421E-01	2.263E-01	1.840E-01	**2.356E-02**
	8	3.175E-02	3.166E-02	2.460E-01	7.204E-01	3.287E-01	2.983E-01	**2.837E-02**
		3.562E-02 +	3.783E-02 +	2.759E-01 +	7.757E-01 +	3.721E-01 +	3.163E-01 +	**3.166E-02**
		4.319E-02	8.833E-02	2.961E-01	8.225E-01	4.031E-01	3.293E-01	4.467E-02
	10	3.537E-02	3.083E-02	3.191E-01	8.163E-01	3.310E-01	3.301E-01	**2.829E-02**
		4.334E-02 +	3.722E-02 +	3.355E-01 +	8.559E-01 +	3.758E-01 +	3.533E-01 +	**3.115E-02**
		4.730E-02	4.268E-02	3.522E-01	8.849E-01	4.024E-01	3.655E-01	**3.879E-02**
	15	4.962E-01	**3.081E-02**	4.893E-01	1.024E+00	5.075E-01	3.686E-01	3.937E-01
		6.030E-01 +	**3.180E-01** −	5.151E-01 −	1.058E+00 +	5.372E-01 =	4.035E-01 −	5.717E-01
		6.691E-01	6.991E-01	5.278E-01	1.076E+00	5.602E-01	**4.336E-01**	6.840E-01
WFG5	3	**2.996E-02**	3.330E-02	5.927E-02	7.405E-02	8.800E-02	6.614E-02	3.096E-02
		3.475E-02 =	**3.457E-02** =	6.271E-02 +	7.721E-02 +	9.903E-02 +	7.025E-02 +	3.536E-02
		4.383E-02	**3.851E-02**	6.667E-02	8.603E-02	1.191E-01	7.468E-02	4.077E-02
	5	3.666E-02	4.140E-02	1.592E-01	4.353E-01	2.756E-01	1.879E-01	**3.453E-02**
		4.005E-02 +	4.335E-02 +	1.638E-01 +	4.615E-01 +	2.996E-01 +	1.946E-01 +	**3.968E-02**
		4.432E-02	4.474E-02	1.707E-01	4.991E-01	3.220E-01	2.081E-01	**4.181E-02**

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
	8	4.727E-02	4.904E-02	2.549E-01	6.436E-01	4.073E-01	3.410E-01	**4.595E-02**
		5.208E-02 =	5.219E-02 =	2.823E-01+	6.827E-01+	4.293E-01 +	3.560E-01+	**5.035E-02**
		7.502E-02	**5.425E-02**	2.953E-01	7.112E-01	4.518E-01	3.628E-01	2.228E-01
	10	**4.529E-02**	4.961E-02	3.197E-01	7.083E-01	4.408E-01	3.941E-01	4.570E-02
		5.011E-02 =	5.238E-02 =	3.323E-01+	7.390E-01+	4.564E-01 +	4.037E-01+	**4.939E-02**
		6.825E-02	**5.565E-02**	3.480E-01	7.867E-01	4.748E-01	4.128E-01	2.130E-01
	15	**3.739E-02**	6.776E-02	5.023E-01	9.562E-01	5.524E-01	4.691E-01	3.793E-02
		4.339E-02 =	2.125E-01 +	5.228E-01+	9.652E-01+	5.716E-01 +	4.930E-01+	**4.238E-02**
		1.529E-01	9.339E-01	5.380E-01	9.800E-01	5.905E-01	5.144E-01	1.776E-01
WFG6	3	2.447E-02	1.904E-02	6.184E-02	6.988E-02	7.806E-02	5.568E-02	**1.371E-02**
		2.831E-02 +	2.820E-02 +	6.474E-02+	7.579E-02+	8.240E-02 +	7.527E-02+	**1.732E-02**
		3.511E-02	3.417E-02	6.728E-02	8.748E-02	8.858E-02	8.048E-02	**2.215E-02**
	5	2.903E-02	2.855E-02	1.518E-01	4.544E-01	2.025E-01	1.667E-01	**1.961E-02**
		3.479E-02 +	3.485E-02 +	1.598E-01+	4.764E-01+	2.149E-01 +	1.726E-01+	**2.307E-02**
		4.261E-02	3.874E-02	1.665E-01	5.265E-01	2.313E-01	1.768E-01	**2.520E-02**
	8	3.350E-02	3.854E-02	2.217E-01	6.502E-01	2.925E-01	3.032E-01	**2.203E-02**
		3.837E-02 +	4.378E-02 +	2.472E-01+	6.808E-01+	3.031E-01 +	3.264E-01+	**2.573E-02**
		4.294E-02	5.208E-02	2.730E-01	7.088E-01	3.235E-01	3.351E-01	3.048E-01
	10	2.793E-02	3.624E-02	2.957E-01	7.208E-01	2.893E-01	3.523E-01	**2.029E-02**
		3.728E-02 +	4.392E-02 +	3.189E-01+	7.314E-01+	3.072E-01 +	3.641E-01+	**2.478E-02**
		4.548E-02	5.207E-02	3.290E-01	7.663E-01	3.303E-01	3.742E-01	**2.962E-02**
	15	2.911E-02	5.071E-02	5.190E-01	9.254E-01	3.386E-01	4.146E-01	**1.597E-02**
		3.599E-02 +	4.863E-01 +	5.275E-01+	9.467E-01+	3.528E-01 +	4.442E-01+	**1.984E-02**
		4.709E-02	1.109E+00	5.425E-01	9.698E-01	3.711E-01	4.742E-01	**2.384E-02**
WFG7	3	**2.136E-03**	3.927E-03	5.002E-02	8.934E-02	9.725E-02	5.180E-02	2.739E-03
		2.572E-03−	4.800E-03 +	5.353E-02+	9.496E-02+	1.077E-01 +	5.561E-02+	3.575E-03
		3.344E-03	6.481E-03	5.731E-02	9.885E-02	1.205E-01	5.824E-02	4.250E-03
	5	**6.704E-03**	1.002E-02	1.407E-01	4.085E-01	2.129E-01	1.586E-01	1.184E-02
		8.688E-03−	1.162E-02 −	1.487E-01+	4.516E-01+	2.340E-01 +	1.638E-01+	1.385E-02
		1.727E-02	**1.381E-02**	1.540E-01	4.860E-01	2.436E-01	1.715E-01	2.369E-02
	8	**1.720E-02**	3.053E-02	2.347E-01	6.595E-01	3.255E-01	3.037E-01	2.693E-02
		2.039E-02−	3.868E-02 +	2.623E-01+	7.024E-01+	3.384E-01 +	3.108E-01+	3.424E-02
		2.553E-02	7.388E-02	2.899E-01	7.434E-01	3.674E-01	3.255E-01	4.047E-02
	10	**2.167E-02**	3.458E-02	3.089E-01	7.288E-01	3.138E-01	3.275E-01	2.264E-02
		2.292E-02−	4.152E-02 +	3.158E-01+	7.656E-01+	3.463E-01 +	3.424E-01+	2.466E-02
		2.441E-02	4.932E-02	3.350E-01	7.961E-01	3.629E-01	3.562E-01	2.634E-02
	15	7.278E-02	3.375E-01	4.973E-01	9.715E-01	3.953E-01	3.683E-01	**1.052E-02**
		1.312E-01 +	6.146E-01 +	5.163E-01+	9.905E-01+	4.254E-01 +	4.085E-01+	**1.333E-02**
		4.640E-01	1.104E+00	5.233E-01	1.017E+00	4.440E-01	4.599E-01	**8.189E-02**
WFG8	3	7.309E-02	**4.154E-02**	9.408E-02	1.402E-01	1.495E-01	8.379E-02	6.507E-02
		7.659E-02 +	**4.488E-02**−	9.869E-02+	1.443E-01+	1.613E-01 +	8.751E-02+	7.534E-02
		7.818E-02	**5.137E-02**	1.030E-01	1.547E-01	1.694E-01	9.144E-02	7.977E-02
	5	1.222E-01	**5.621E-02**	2.044E-01	4.470E-01	2.805E-01	2.092E-01	8.486E-02
		1.307E-01 +	**7.042E-02**−	2.195E-01+	5.026E-01+	2.983E-01 +	2.170E-01+	1.129E-01
		1.408E-01	**7.473E-02**	2.291E-01	5.278E-01	3.156E-01	2.227E-01	1.244E-01
	8	2.483E-01	1.205E-01	3.931E-01	6.844E-01	3.800E-01	3.715E-01	**7.524E-02**
		2.577E-01 +	**1.327E-01**−	4.070E-01+	7.101E-01+	4.036E-01 +	3.925E-01+	1.767E-01
		2.642E-01	**1.409E-01**	4.259E-01	7.445E-01	4.218E-01	4.047E-01	4.293E-01
	10	2.813E-01	1.361E-01	3.785E-01	7.578E-01	3.851E-01	4.332E-01	**5.194E-02**
		3.062E-01 +	1.549E-01 +	4.452E-01+	7.831E-01+	4.055E-01 +	4.442E-01+	**6.690E-02**
		3.191E-01	**1.618E-01**	4.767E-01	8.191E-01	4.210E-01	4.511E-01	3.980E-01
	15	5.435E-01	6.018E-01	5.788E-01	9.827E-01	**4.629E-01**	5.401E-01	5.593E-01
		5.902E-01 =	9.638E-01 +	5.949E-01=	9.946E-01+	**4.827E-01**−	5.563E-01−	5.912E-01
		6.306E-01	1.113E+00	6.115E-01	1.020E+00	**4.913E-01**	5.715E-01	6.100E-01
WFG9	3	3.281E-02	**2.943E-02**	6.125E-02	7.135E-02	7.693E-02	6.145E-02	3.069E-02
		5.194E-02 =	5.358E-02 =	7.244E-02+	7.735E-02+	8.423E-02 +	8.471E-02+	**4.163E-02**
		6.727E-02	**5.840E-02**	8.152E-02	8.645E-02	1.088E-01	8.889E-02	6.557E-02
	5	5.806E-02	**5.311E-02**	1.683E-01	4.213E-01	2.118E-01	1.940E-01	8.684E-02
		9.513E-02 =	**6.368E-02**−	1.867E-01+	4.586E-01+	2.597E-01 +	2.052E-01+	8.806E-02
		1.036E-01	**7.271E-02**	1.953E-01	4.720E-01	2.794E-01	2.095E-01	8.986E-02
	8	1.182E-01	**8.827E-02**	2.596E-01	6.317E-01	4.075E-01	3.504E-01	9.126E-02
		1.340E-01 =	**1.188E-01**=	2.944E-01+	6.853E-01+	4.280E-01 +	3.625E-01+	1.245E-01
		2.165E-01	**1.980E-01**	3.239E-01	7.153E-01	4.538E-01	3.760E-01	2.342E-01
	10	1.071E-01	**9.076E-02**	3.271E-01	7.213E-01	4.388E-01	3.983E-01	1.396E-01
		1.535E-01 −	**1.233E-01**−	3.489E-01+	7.425E-01+	4.595E-01 +	4.063E-01+	1.902E-01
		1.994E-01	**1.675E-01**	3.639E-01	7.687E-01	4.793E-01	4.165E-01	2.407E-01
	15	1.702E-01	**1.036E-01**	4.975E-01	9.396E-01	5.585E-01	5.051E-01	2.120E-01
		2.322E-01 −	**1.688E-01**−	5.239E-01+	9.557E-01+	6.046E-01 +	5.221E-01+	4.356E-01
		4.239E-01	**2.898E-01**	5.425E-01	9.995E-01	6.397E-01	5.607E-01	5.835E-01
(+/=/-)		15/14/16	26/7/12	35/2/8	45/0/0	39/3/3	33/2/10	

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

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
WFG1		7.12840E-01	6.82790E-01	**8.74760E-01**	6.01700E-01	6.06790E-01	8.45320E-01	7.05130E-01
	3	7.03900E-01 −	6.70940E-01 +	**8.32610E-01** −	5.91580E-01 +	6.05430E-01 +	7.94780E-01 −	6.92700E-01
		6.95820E-01	6.65730E-01	**7.94030E-01**	5.86330E-01	6.03010E-01	7.54300E-01	6.83840E-01
		6.50050E-01	6.40660E-01	**7.67260E-01**	3.00280E-01	5.59280E-01	7.64530E-01	6.37340E-01
	5	6.44320E-01 −	6.23950E-01 +	6.95360E-01 −	2.74170E-01 +	5.54510E-01 +	**7.00990E-01** −	6.33590E-01
		6.40330E-01	6.19950E-01	**6.51580E-01**	2.64420E-01	5.52630E-01	6.47850E-01	6.26480E-01
		6.61610E-01	6.74480E-01	**8.75640E-01**	2.87930E-01	5.01750E-01	8.05640E-01	6.99510E-01
	8	6.00340E-01 =	6.34840E-01 −	**8.59300E-01** −	2.75920E-01 +	4.99990E-01 +	6.95770E-01 −	5.71700E-01
		5.86340E-01	4.66590E-01	**8.27990E-01**	2.63190E-01	4.98380E-01	6.00520E-01	4.95100E-01
		6.93100E-01	6.93930E-01	9.32610E-01	2.98690E-01	4.85500E-01	**9.61450E-01**	8.31540E-01
	10	6.51950E-01 −	6.78220E-01 =	**9.20530E-01** −	2.82990E-01 +	4.76400E-01 +	9.17990E-01 −	5.67250E-01
		5.66840E-01	4.51850E-01	**9.08160E-01**	2.72020E-01	4.74660E-01	6.20230E-01	5.04820E-01
WFG2		**9.87120E-01**	9.86730E-01	9.81860E-01	9.43810E-01	9.68160E-01	9.82720E-01	9.85280E-01
	3	9.84290E-01 =	**9.84940E-01** −	9.78080E-01 +	9.39520E-01 +	9.62460E-01 +	9.80360E-01 +	9.83090E-01
		8.93220E-01	8.92270E-01	8.88500E-01	9.31490E-01	9.52570E-01	9.70600E-01	**9.77640E-01**
		9.96560E-01	9.96080E-01	9.94770E-01	7.44550E-01	9.74820E-01	**9.98280E-01**	9.93210E-01
	5	9.95630E-01 −	9.95120E-01 −	9.90850E-01 =	6.88980E-01 +	9.65900E-01 +	**9.97770E-01** −	9.89420E-01
		8.97790E-01	9.93820E-01	8.94370E-01	6.57400E-01	9.56030E-01	**9.95190E-01**	9.81090E-01
		9.97490E-01	9.97090E-01	9.95660E-01	6.96120E-01	9.57910E-01	**9.99280E-01**	9.84450E-01
	8	9.93110E-01 =	9.95430E-01 −	9.91880E-01 −	6.65430E-01 +	9.39680E-01 =	**9.98750E-01** −	9.64060E-01
		8.94710E-01	8.97440E-01	8.92850E-01	6.42620E-01	9.18920E-01	**9.97110E-01**	8.55900E-01
		9.98030E-01	9.98120E-01	9.97080E-01	7.18580E-01	9.65180E-01	**9.99730E-01**	9.93650E-01
	10	9.94850E-01 =	9.97130E-01 −	9.95500E-01 −	6.66150E-01 +	9.49230E-01 =	**9.99500E-01** −	9.85120E-01
		8.95970E-01	8.97380E-01	9.92410E-01	6.52070E-01	9.25840E-01	**9.99040E-01**	8.87610E-01
WFG3		**8.75020E-01**	8.74280E-01	8.71150E-01	8.62320E-01	8.63000E-01	8.74740E-01	8.71650E-01
	3	**8.71570E-01** −	8.67970E-01 =	8.61310E-01 +	8.53680E-01 +	8.47220E-01 +	8.65220E-01 =	8.65570E-01
		8.66020E-01	8.53310E-01	8.52490E-01	8.42350E-01	8.36440E-01	8.59630E-01	8.52050E-01
		8.75090E-01	8.60600E-01	8.65350E-01	6.92590E-01	8.48550E-01	**8.86780E-01**	8.78340E-01
	5	8.66150E-01 =	8.38480E-01 +	8.43920E-01 +	6.71660E-01 +	8.35830E-01 +	**8.72820E-01** −	8.66090E-01
		8.60270E-01	8.20580E-01	8.28910E-01	6.47770E-01	8.25210E-01	**8.63440E-01**	8.49740E-01
		8.73700E-01	7.13170E-01	8.64960E-01	6.76230E-01	8.45170E-01	8.73460E-01	8.39750E-01
	8	8.55450E-01 −	6.62100E-01 +	8.48120E-01 −	6.49880E-01 +	8.27250E-01 −	**8.62380E-01** −	8.07590E-01
		7.91750E-01	5.89310E-01	8.29240E-01	6.40190E-01	8.18250E-01	**8.39310E-01**	7.12320E-01
		8.67930E-01	7.70670E-01	8.68470E-01	6.70530E-01	8.51050E-01	**8.75170E-01**	8.52380E-01
	10	8.62790E-01 −	7.33000E-01 +	8.38090E-01 −	6.60290E-01 +	8.42760E-01 −	**8.63960E-01** −	8.33230E-01
		8.40580E-01	6.32980E-01	8.27910E-01	6.37920E-01	8.22140E-01	**8.48880E-01**	7.78990E-01
WFG4		9.23800E-01	9.22330E-01	9.20650E-01	9.06790E-01	8.82340E-01	9.18590E-01	**9.23970E-01**
	3	**9.22670E-01** =	9.20690E-01 +	9.18680E-01 +	8.97430E-01 +	8.79280E-01 +	9.16450E-01 +	9.22310E-01
		9.21520E-01	9.17950E-01	9.14610E-01	8.89830E-01	8.74120E-01	9.11490E-01	9.21120E-01
		9.82040E-01	9.80220E-01	9.73930E-01	6.55620E-01	9.34800E-01	9.82000E-01	9.81170E-01
	5	9.78910E-01 =	9.78940E-01 =	9.69580E-01 +	6.36480E-01 +	9.25850E-01 +	9.78390E-01 =	**9.79520E-01**
		9.74300E-01	9.76700E-01	9.64310E-01	6.05550E-01	9.18370E-01	9.73290E-01	**9.76980E-01**
		9.87140E-01	**9.92110E-01**	9.88010E-01	6.49310E-01	9.27150E-01	9.87910E-01	9.86530E-01
	8	9.82120E-01 =	**9.89240E-01** −	9.80910E-01 =	6.14980E-01 +	9.10920E-01 +	9.82390E-01 =	9.83520E-01
		9.77580E-01	**9.85780E-01**	9.74290E-01	5.85030E-01	8.85470E-01	9.73770E-01	9.77480E-01
		9.86710E-01	**9.95840E-01**	9.85140E-01	6.57320E-01	9.40030E-01	9.89000E-01	9.89340E-01
	10	9.83350E-01 +	**9.94740E-01** −	9.81830E-01 +	6.31970E-01 +	9.27630E-01 +	9.83050E-01 +	9.87020E-01
		9.77770E-01	**9.93070E-01**	9.78790E-01	6.03330E-01	9.11570E-01	9.73810E-01	9.80330E-01
WFG5		**9.03370E-01**	8.98040E-01	9.02990E-01	8.82540E-01	8.72420E-01	8.92250E-01	9.02860E-01
	3	**8.99610E-01** =	8.93180E-01 +	8.98100E-01 =	8.80560E-01 +	8.65590E-01 +	8.87070E-01 +	8.97510E-01
		8.95120E-01	8.89130E-01	8.93790E-01	8.70580E-01	8.54300E-01	8.82730E-01	8.91570E-01
		9.59720E-01	9.50670E-01	9.56040E-01	6.24710E-01	8.87640E-01	9.51630E-01	**9.61260E-01**
	5	9.58590E-01 +	9.47760E-01 +	9.53650E-01 +	6.09680E-01 +	8.70720E-01 +	9.40420E-01 +	**9.60360E-01**
		9.57080E-01	9.46240E-01	9.49460E-01	5.96880E-01	8.55520E-01	9.32700E-01	**9.58510E-01**
		9.62480E-01	9.58040E-01	9.62240E-01	6.01020E-01	8.77750E-01	9.45370E-01	**9.63990E-01**
	8	9.60390E-01 +	9.55180E-01 +	9.60370E-01 +	5.67660E-01 +	8.56950E-01 +	9.38590E-01 +	**9.62690E-01**
		9.56410E-01	9.51790E-01	9.58730E-01	5.40930E-01	8.42830E-01	9.26950E-01	**9.59470E-01**
		9.61040E-01	9.58640E-01	9.60640E-01	6.10130E-01	8.80580E-01	9.44740E-01	**9.62420E-01**
	10	9.60030E-01 +	9.56310E-01 +	9.58770E-01 +	5.86790E-01 +	8.65980E-01 +	9.40820E-01 +	**9.61760E-01**
		9.57960E-01	9.52040E-01	9.53100E-01	5.68660E-01	8.50480E-01	9.33540E-01	**9.60510E-01**

cial Intelligence Review 54:2567–2608, DOI https://doi.org/10.1007/s10462-020-09909-3

2. Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Computer Methods in Applied Mechanics and Engineering 376:113609, DOI https://doi.org/10.1016/j.cma.2020.113609
3. Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, Gandomi AH (2021) Aquila optimizer: A novel meta-heuristic optimization algorithm. Computers & Industrial Engineering 157:107250, DOI https://doi.org/10.1016/j.cie.2021.107250
4. Abualigah LMQ (2019) Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering, vol 816, 1st edn. Springer International Publishing

	M	NSGA-III	SPEA/R	VaEA	dMOPSO	SMPSO	MaPSO	RMaOPSO
WFG6	3	9.07810E-01	9.12380E-01	9.06840E-01	9.15370E-01	9.10770E-01	**9.16270E-01**	9.16150E-01
		9.04990E-01 +	9.04200E-01 +	9.01570E-01+	9.08540E-01+	9.07230E-01+	8.82970E-01 +	**9.13290E-01**
		8.99480E-01	9.00880E-01	8.97230E-01	8.81980E-01	8.97030E-01	8.82230E-01	**9.09570E-01**
	5	9.65910E-01	9.68320E-01	9.62430E-01	6.03800E-01	9.38640E-01	9.36850E-01	**9.74700E-01**
		9.61300E-01 +	9.58200E-01 +	9.56510E-01+	5.85100E-01+	9.31730E-01+	9.36550E-01 +	**9.71390E-01**
		9.54490E-01	9.54280E-01	9.51670E-01	5.55760E-01	9.28400E-01	9.36340E-01	**9.69520E-01**
	8	9.71930E-01	9.76720E-01	9.74290E-01	5.60780E-01	9.45080E-01	9.36130E-01	**9.80850E-01**
		9.65630E-01 +	9.65940E-01 +	9.65950E-01+	5.44850E-01+	9.41560E-01+	9.36070E-01 +	**9.77300E-01**
		9.60550E-01	9.58560E-01	9.59030E-01	5.33000E-01	9.40570E-01	9.36000E-01	9.58550E-01
	10	9.76850E-01	9.80160E-01	9.73060E-01	5.79590E-01	9.49290E-01	9.32110E-01	**9.83180E-01**
		9.65770E-01 +	9.67310E-01 +	9.64340E-01+	5.55450E-01+	9.44460E-01+	9.32100E-01 +	**9.78080E-01**
		9.55710E-01	9.56200E-01	9.50880E-01	5.40360E-01	9.42270E-01	9.32090E-01	**9.71950E-01**
WFG7	3	9.25530E-01	9.24960E-01	9.22930E-01	8.82550E-01	8.83970E-01	**9.26440E-01**	9.25370E-01
		9.25160E-01 −	9.24160E-01 +	9.22650E-01+	8.75930E-01+	8.66970E-01+	**9.26070E-01**−	9.24830E-01
		9.24530E-01	9.23570E-01	9.21630E-01	8.67330E-01	8.53260E-01	**9.25590E-01**	9.24550E-01
	5	9.87490E-01	9.84000E-01	9.83410E-01	6.85290E-01	8.84020E-01	**9.89750E-01**	9.85980E-01
		9.86590E-01 −	9.83520E-01 +	9.80370E-01+	6.47060E-01+	8.75270E-01+	**9.89640E-01**−	9.84320E-01
		9.85090E-01	9.81920E-01	9.77780E-01	6.17670E-01	8.67030E-01	**9.89450E-01**	9.82400E-01
	8	9.95210E-01	9.95600E-01	9.95010E-01	6.28080E-01	8.97910E-01	**9.99130E-01**	9.93540E-01
		9.93990E-01 −	9.94890E-01 −	9.94400E-01−	6.13800E-01+	8.84050E-01+	**9.99050E-01**−	9.90620E-01
		9.91930E-01	9.93800E-01	9.93570E-01	6.00910E-01	8.68130E-01	**9.98940E-01**	9.89140E-01
	10	9.96800E-01	9.97810E-01	9.96620E-01	6.51580E-01	9.17520E-01	**9.99840E-01**	9.95780E-01
		9.96000E-01 −	9.97570E-01 −	9.95440E-01−	6.28300E-01+	8.99570E-01+	**9.99810E-01**−	9.94840E-01
		9.94820E-01	9.96790E-01	9.93230E-01	6.10660E-01	8.89000E-01	**9.99730E-01**	9.93570E-01
WFG8	3	9.07130E-01	**9.12060E-01**	8.93340E-01	8.35130E-01	8.45640E-01	9.01710E-01	9.05690E-01
		9.04450E-01 −	**9.09470E-01**−	8.90140E-01+	8.22450E-01+	8.29490E-01+	8.99280E-01 +	9.03170E-01
		9.03040E-01	**9.05760E-01**	8.86740E-01	8.14720E-01	8.13670E-01	8.97750E-01	9.01150E-01
	5	9.68530E-01	**9.73780E-01**	9.53750E-01	6.33600E-01	8.62840E-01	9.70330E-01	9.66030E-01
		9.65260E-01 −	**9.70780E-01**−	9.45730E-01+	6.07020E-01+	8.48690E-01+	9.69380E-01 −	9.60450E-01
		9.60950E-01	**9.66910E-01**	9.35660E-01	5.94110E-01	8.34860E-01	9.64090E-01	9.52750E-01
	8	9.75980E-01	**9.89670E-01**	9.62950E-01	6.12310E-01	8.77300E-01	9.87860E-01	9.68900E-01
		9.67820E-01 −	**9.88220E-01**−	9.52950E-01=	5.89980E-01+	8.48820E-01+	9.83540E-01 −	9.46650E-01
		9.55860E-01	**9.85050E-01**	9.36220E-01	5.54250E-01	8.29400E-01	9.77240E-01	9.26980E-01
	10	9.78740E-01	**9.94820E-01**	9.67200E-01	6.27390E-01	8.97600E-01	9.92790E-01	9.78060E-01
		9.73090E-01 −	**9.94030E-01**−	9.56930E-01+	6.01290E-01+	8.70360E-01+	9.91020E-01 −	9.67140E-01
		9.64010E-01	**9.92960E-01**	9.39640E-01	5.81340E-01	8.51410E-01	9.81690E-01	9.33220E-01
WFG9	3	8.93710E-01	8.89040E-01	8.93400E-01	8.80610E-01	8.82210E-01	8.92260E-01	**8.94030E-01**
		8.74290E-01 =	8.62340E-01 =	8.73110E-01=	8.68350E-01+	8.73650E-01=	8.53540E-01 +	**8.82850E-01**
		8.59390E-01	8.59100E-01	8.53690E-01	8.57600E-01	8.45330E-01	8.49270E-01	**8.60660E-01**
	5	**9.45610E-01**	9.21170E-01	9.35730E-01	6.61650E-01	8.90160E-01	9.39710E-01	9.14570E-01
		9.07190E-01 =	9.01670E-01 +	9.01900E-01=	6.23030E-01+	8.72930E-01+	8.98630E-01 +	**9.09710E-01**
		9.03020E-01	8.96610E-01	8.97920E-01	6.06940E-01	8.64720E-01	8.93420E-01	**9.07220E-01**
	8	9.03250E-01	9.03290E-01	**9.39060E-01**	6.38670E-01	8.78890E-01	8.98110E-01	9.34060E-01
		8.98530E-01 +	8.91790E-01 +	8.97570E-01=	5.85210E-01+	8.66720E-01+	8.91610E-01 +	**8.99950E-01**
		8.85810E-01	8.80880E-01	8.90390E-01	5.43030E-01	8.46450E-01	8.87440E-01	**8.95370E-01**
	10	9.43600E-01	9.22390E-01	**9.44250E-01**	6.38010E-01	8.94820E-01	8.96880E-01	9.14540E-01
		9.01710E-01=	8.94760E-01 =	8.98490E-01=	5.97190E-01+	8.82230E-01+	8.91780E-01 +	9.00330E-01
		8.91960E-01	8.87560E-01	8.91280E-01	5.46390E-01	8.64140E-01	8.85640E-01	**8.95410E-01**
(+/=/-)		9/12/15	18/5/13	18/8/10	36/0/0	31/3/2	16/3/17	

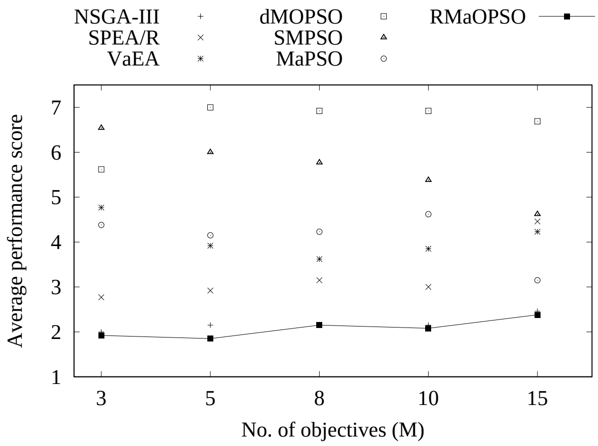

Fig. 6 Average performance score based on the median IGD values over all objectives for different DTLZ and WFG problem instances. The solid line represents the performance of RMaOPSO.

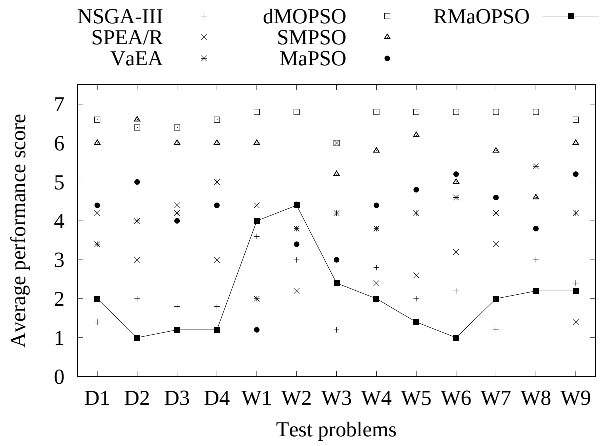

Fig. 7 Average performance score based on the median IGD values over all DTLZ (Dx) and WFG (Wx) problem instances. The solid line represents the performance of RMaOPSO.

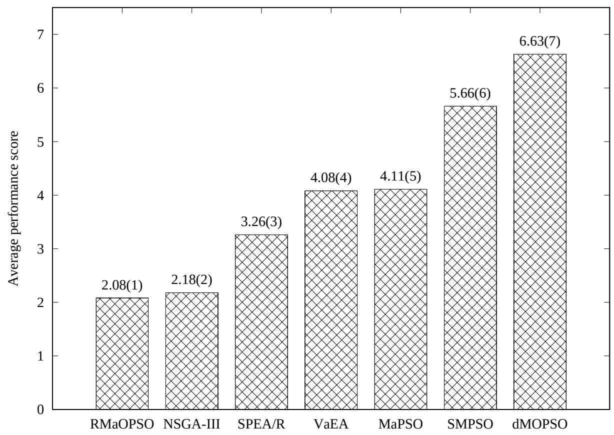

Fig. 8 Ranking of algorithms over all DTLZ and WFG test instances. The smaller rank represents better performance.

5. Al Moubayed N, Petrovski A, McCall J (2012) D^2mopso: Multi-objective particle swarm optimizer based on decomposition and dominance. In: Hao JK, Middendorf M (eds) Evolutionary Computation in Combinatorial Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 75–86
6. Barakat N, Sharma D (2019) Evolutionary multi-objective optimization for bulldozer and its blade in soil cutting. International Journal of Management Science and Engineering Management 14(2):102–112, DOI 10.1080/17509653.2018.1500953
7. Britto A, Pozo A (2012) I-mopso: A suitable pso algorithm for many-objective optimization. In: 2012 Brazilian Symposium on Neural Networks, pp 166–171, DOI 10.1109/SBRN.2012.20
8. Britto A, Pozo A (2012) Using archiving methods to control convergence and diversity for many-objective problems in particle swarm optimization. In: 2012 IEEE Congress on Evolutionary Computation, pp 1–8, DOI 10.1109/CEC.2012.6256149
9. Britto A, Pozo A (2014) Using reference points to update the archive of mopso algorithms in many-objective optimization. Neurocomputing 127:78 – 87, DOI https://doi.org/10.1016/j.neucom.2013.05.049, advances in Intelligent Systems
10. de Carvalho AB, Pozo A (2012) Measuring the convergence and diversity of cdas multi-objective particle swarm optimization algorithms: A study of many-objective problems. Neurocomputing 75(1):43 – 51, DOI https://doi.org/10.1016/j.neucom.2011.03.053, brazilian Symposium on Neural Networks (SBRN 2010) International Conference on Hybrid Artificial Intelligence Systems (HAIS 2010)
11. Castro OR, Santana R, Pozo A (2016) C-multi: A competent multi-swarm approach for many-objective problems. Neurocomputing 180:68 – 78, DOI doi.org/10.1016/j.neucom.2015.06.097, progress in Intelligent Systems Design
12. Cheng T, Chen M, Fleming PJ, Yang Z, Gan S (2017) A novel hybrid teaching learning based multi-objective particle swarm optimization. Neurocomputing 222:11 – 25, DOI https://doi.org/10.1016/j.neucom.2016.10.001
13. Coello CAC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3):256–279, DOI 10.1109/TEVC.2004.826067
14. Coello Coello CA, Lechuga MS (2002) MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), vol 2, pp 1051–1056 vol.2, DOI 10.1109/CEC.2002.1004388
15. Dai C, Wang Y, Ye M (2015) A new multi-objective particle swarm optimization algorithm based on decomposition. Information Sciences 325:541 – 557, DOI https://doi.org/10.1016/j.ins.2015.07.018
16. Das I, Dennis JE (1998) Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. SIAM Journal on Optimization 8(3):631–657, DOI 10.1137/S1052623496307510
17. Deb K, Agrawal RB (1995) Simulated binary crossover for continuous search space. Complex Systems 9(2):115–148
18. Deb K, Jain H (2014) 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):577–601, DOI 10.1109/TEVC.2013.2281535
19. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. Evolutionary Computation, IEEE Transactions on 6(2):182 –197
20. Deb K, Thiele L, Laumanns M, Zitzler E (2005) Scalable test problems for evolutionary multiobjective optimization. In: Abraham A, Jain L, Goldberg R (eds) Evolutionary multiobjective optimization: theoritical advances and applications, Springer-Verlag, London, pp 105–145
21. Deb K, Gupta S, Daum D, Branke J, Mall A, Padmanabhan D (2009) Reliability-based optimization using evolutionary algorithms. IEEE Transaction on Evolutionary Computation 13(5):1054–1074
22. Durillo JJ, Nebro AJ (2011) jmetal: A java framework for multi-objective optimization. Advances

in Engineering Software 42:760–771, DOI DOI:10.1016/j.advengsoft.2011.05.014
23. Figueiredo E, Ludermir T, Bastos-Filho C (2016) Many objective particle swarm optimization. Information Sciences 374:115 – 134, DOI https://doi.org/10.1016/j.ins.2016.09.026
24. Han H, Lu W, Zhang L, Qiao J (2018) Adaptive gradient multiobjective particle swarm optimization. IEEE Transactions on Cybernetics 48(11):3067–3079, DOI 10.1109/TCYB.2017.2756874
25. Hirano H, Yoshikawa T (2013) A study on two-step search based on pso to improve convergence and diversity for many-objective optimization problems. In: 2013 IEEE Congress on Evolutionary Computation, pp 1854–1859, DOI 10.1109/CEC.2013.6557785
26. Hu W, Yen GG (2015) Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system. IEEE Transactions on Evolutionary Computation 19(1):1–18, DOI 10.1109/TEVC.2013.2296151
27. Hu W, Yen GG, Luo G (2017) Many-objective particle swarm optimization using two-stage strategy and parallel cell coordinate system. IEEE Transactions on Cybernetics 47(6):1446–1459, DOI 10.1109/TCYB.2016.2548239
28. Huband S, Hingston P, Barone L, While L (2006) A review of multiobjective test problems and a scalable test problem toolkit. IEEE Transactions on Evolutionary Computation 10(5):477–506, DOI 10.1109/TEVC.2005.861417
29. Jiang S, Yang S (2017) A strength pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization. IEEE Transactions on Evolutionary Computation 21(3):329–346, DOI 10.1109/TEVC.2016.2592479
30. Junior O, Britto A, Pozo A (2012) A comparison of methods for leader selection in many-objective problems. In: 2012 IEEE Congress on Evolutionary Computation, pp 1–8, DOI 10.1109/CEC.2012.6256415
31. Köppen M, Yoshida K (2007) Many-objective particle swarm optimization by gradual leader selection. In: Beliczynski B, Dzielinski A, Iwanowski M, Ribeiro B (eds) Adaptive and Natural Computing Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 323–331
32. Li L, Wang W, Xu X (2017) Multi-objective particle swarm optimization based on global margin ranking. Information Sciences 375:30 – 47, DOI https://doi.org/10.1016/j.ins.2016.08.043
33. Li L, Chang L, Gu T, Sheng W, Wang W (2020) On the norm of dominant difference for many-objective particle swarm optimization. IEEE Transactions on Cybernetics (in press) pp 1–13
34. Lin Q, Li J, Du Z, Chen J, Ming Z (2015) A novel multi-objective particle swarm optimization with multiple search strategies. European Journal of Operational Research 247(3):732 – 744, DOI https://doi.org/10.1016/j.ejor.2015.06.071
35. Lin Q, Liu S, Zhu Q, Tang C, Song R, Chen J, Coello CAC, Wong K, Zhang J (2018) Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems. IEEE Transactions on Evolutionary Computation 22(1):32–46, DOI 10.1109/TEVC.2016.2631279
36. Liu X, Zhan Z, Gao Y, Zhang J, Kwong S, Zhang J (2019) Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization. IEEE Transactions on Evolutionary Computation 23(4):587–602
37. Luo J, Huang X, Yang Y, Li X, Wang Z, Feng J (2020) A many-objective particle swarm optimizer based on indicator and direction vectors for many-objective optimization. Information Sciences 514:166 – 202, DOI https://doi.org/10.1016/j.ins.2019.11.047
38. Mostaghim S, Schmeck H (2008) Distance based ranking in many-objective particle swarm optimization. In: Rudolph G, Jansen T, Beume N, Lucas S, Poloni C (eds) Parallel Problem Solving from Nature – PPSN X, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 753–762
39. Mostaghim S, Schmeck H (2008) Distance based ranking in many-objective particle swarm optimization. In: Rudolph G, Jansen T, Beume N, Lucas S, Poloni C (eds) Parallel Problem Solving from Nature – PPSN X, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 753–762
40. Nebro AJ, Durillo JJ, Garcia-Nieto J, Coello Coello CA, Luna F, Alba E (2009) Smpso: A new pso-based metaheuristic for multi-objective optimization. In: 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making(MCDM), pp 66–73, DOI 10.1109/MCDM.2009.4938830
41. Padhye N, Branke J, Mostaghim S (2009) Empirical comparison of mopso methods - guide selection and diversity preservation -. In: 2009 IEEE Congress on Evolutionary Computation, pp 2516–2523, DOI 10.1109/CEC.2009.4983257
42. Pan A, Wang L, Guo W, Wu Q (2018) A diversity enhanced multiobjective particle swarm optimization. Information Sciences 436-437:441 – 465,

DOI https://doi.org/10.1016/j.ins.2018.01.038
43. Peng W, Zhang Q (2008) A decomposition-based multi-objective particle swarm optimization algorithm for continuous optimization problems. In: 2008 IEEE International Conference on Granular Computing, pp 534–537, DOI 10.1109/GRC.2008.4664724
44. Purshouse RC, Fleming PJ (2007) On the evolutionary optimization of many conflicting objectives. IEEE Transactions on Evolutionary Computation 11(6):770–784, DOI 10.1109/TEVC.2007.910138
45. Qin S, Sun C, Zhang G, He X, Tan Y (2020) A modified particle swarm optimization based on decomposition with different ideal points for many-objective optimization problems. Complex & Intelligent Systems (in press) pp 1 – 12, DOI https://doi.org/10.1007/s40747-020-00134-7
46. Ram L, Sharma D (2017) Evolutionary and gpu computing for topology optimization of structures. Swarm and Evolutionary Computation 35:1 – 13, DOI https://doi.org/10.1016/j.swevo.2016.08.004
47. Ray T, Tai K, Seow KC (2001) Multiobejctive design optimization by an evolutionary algorithm. Engineering Optimization 33(4):399–424
48. Ser JD, Osaba E, Molina D, Yang XS, Salcedo-Sanz S, Camacho D, Das S, Suganthan PN, Coello CAC, Herrera F (2019) Bio-inspired computation: Where we stand and what's next. Swarm and Evolutionary Computation 48:220 – 250, DOI doi.org/10.1016/j.swevo.2019.04.008
49. Sharma D, Shukla PK (2019) Line-prioritized environmental selection and normalization scheme for many-objective optimization using reference-lines-based framework. Swarm and Evolutionary Computation 51:100592, DOI https://doi.org/10.1016/j.swevo.2019.100592
50. Sharma D, Deb K, Kishore NN (2011) Domain-specific initial population strategy for compliant mechanisms using customized genetic algorithm. Structural and Multidisciplinary Optimization 43(4):541–554
51. Sharma D, Deb K, Kishore NN (2014) Customized evolutionary optimization procedure for generating minimum weight compliant mechanisms. Engineering Optimization 46(1):39–60
52. Sharma D, Basha SZ, Kumar SA (2019) Diversity over dominance approach for many-objective optimization on reference-points-based framework. In: Deb K, Goodman E, Coello Coello CA, Klamroth K, Miettinen K, Mostaghim S, Reed P (eds) Evolutionary Multi-Criterion Optimization, Springer International Publishing, Cham, pp 278–290
53. Sharma D, Vats S, Saurabh S (2021) Diversity preference-based many-objective particle swarm optimization using reference-lines-based framework. Swarm and Evolutionary Computation 65:100910, DOI https://doi.org/10.1016/j.swevo.2021.100910
54. Sierra MR, Coello Coello CA (2005) Improving pso-based multi-objective optimization using crowding, mutation and epsilon-dominance. In: Coello Coello CA, Hernández Aguirre A, Zitzler E (eds) Evolutionary Multi-Criterion Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 505–519
55. Wickramasinghe UK, Li X (2009) Using a distance metric to guide pso algorithms for many-objective optimization. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, ACM, New York, NY, USA, GECCO '09, pp 667–674, DOI 10.1145/1569901.1569993
56. Woolard MM, Fieldsend JE (2013) On the effect of selection and archiving operators in many-objective particle swarm optimisation. In: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, ACM, New York, NY, USA, GECCO '13, pp 129–136, URL `http://doi.acm.org/10.1145/2463372.2463380`
57. Wu B, Hu W, Hu J, Yen GG (2020) Adaptive multiobjective particle swarm optimization based on evolutionary state estimation. IEEE Transactions on Cybernetics (in press) pp 1–14
58. Xiang Y, Zhou Y, Li M, Chen Z (2017) A vector angle-based evolutionary algorithm for unconstrained many-objective optimization. IEEE Transactions on Evolutionary Computation 21(1):131–152, DOI 10.1109/TEVC.2016.2587808
59. Xiang Y, Zhou Y, Chen Z, Zhang J (2020) A many-objective particle swarm optimizer with leaders selected from historical solutions by using scalar projections. IEEE Transactions on Cybernetics 50(5):2209–2222
60. Yang W, Chen L, Wang Y, Zhang M (2020) A reference points and intuitionistic fuzzy dominance based particle swarm algorithm for multi/many-objective optimization. Applied Intelligence 50:1133 – 1154, DOI https://doi.org/10.1007/s10489-019-01569-3
61. Yu H, Wang Y, Xiao S (2020) Multi-objective particle swarm optimization based on cooperative hybrid strategy. Applied Intelligence 50:256 – 269, DOI https://doi.org/10.1007/s10489-019-01496-3
62. Zapotecas Martínez S, Coello Coello CA (2011) A multi-objective particle swarm optimizer based on decomposition. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Compu-

tation, ACM, New York, NY, USA, GECCO '11, pp 69–76, DOI 10.1145/2001576.2001587
63. Zhang X, Zheng X, Cheng R, Qiu J, Jin Y (2018) A competitive mechanism based multi-objective particle swarm optimizer with fast convergence. Information Sciences 427:63 – 76, DOI https://doi.org/10.1016/j.ins.2017.10.037
64. Zhu Q, Lin Q, Chen W, Wong K, Coello Coello CA, Li J, Chen J, Zhang J (2017) An external archive-guided multiobjective particle swarm optimization algorithm. IEEE Transactions on Cybernetics 47(9):2794–2808, DOI 10.1109/TCYB.2017.2710133