IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 10, 2020, accepted June 29, 2020, date of publication July 7, 2020, date of current version July 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3007846

A Fast Multi-Objective Particle Swarm Optimization Algorithm Based on a New Archive Updating Mechanism

KHALIL ALKEBSI AND WENLI DU

Key Laboratory of Advanced Control and Optimization for Chemical Processes, East China University of Science and Technology, Shanghai 200237, China

Corresponding author: Wenli Du (wldu@ecust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China through the Basic Science Center Program under Grant 61988101, in part by the International (Regional) Cooperation and Exchange Project under Grant 61720106008, in part by the National Natural Science Fund for Distinguished Young Scholars under Grant 61725301, and in part by the National Natural Science Foundation of China under Grant 61703163.

ABSTRACT Multi-objective optimization has received increasing attention over the past few decades, and a large number of nature-inspired metaheuristic algorithms have been developed to solve multi-objective problems. An external archive is often used to store elite solutions in multi-objective algorithms. Since the archive size is limited, it must be truncated when the number of nondominated solutions exceeds its maximum size. Thus, the archive updating strategy is crucial due to its influence in the performance of the algorithm. However, achieving a fast convergence speed while assuring diversity of the obtained solutions is always a challenging task. In this paper, a novel multi-objective particle swarm optimization algorithm based on a new archive updating mechanism which depends on the nearest neighbor approach, called MOPSONN, is proposed. Two archive updating strategies are adopted to update nondominated solutions in the archive, which are beneficial to accelerate the convergence speed and maintain diversity of the swarm. The performance of MOPOSNN is evaluated on several benchmark problems and compared with seven state-of-the-art multi-objective algorithms, including four multi-objective particle swarm optimization algorithms and three multi-objective evolutionary algorithms. The experimental results demonstrate the significant effectiveness of MOPSONN in terms of convergence speed and spread of solutions.

INDEX TERMS Multi-objective optimization, particle swarm optimization, evolutionary algorithm, crowded sorting.

I. INTRODUCTION

In many real-world optimization problems, the problem involves two or more conflicting objective which need to be optimized simultaneously [1]–[3]. Such optimization problem is termed as multi-objective optimization problems (MOPs). Since there often exists a conflict between the objectives, in which an optimal solution for one objective may be the worst solution for another. In other words, improvement of the fitness of one objective is only possible by accepting weakness in the fitness of other objectives which is referred as Pareto optimality. Therefore, MOPs can only be described by a set of Pareto optimal solutions called as Pareto front [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Diego Oliva.

Multi-objective optimization has received an increasing attention over the past few decades, and a large number of nature-inspired meta-heuristic algorithms have been developed to solve MOPs based on differential evolution (DE) [5], genetic algorithm (GA) [6], particle swarm optimization (PSO) [7], etc. It is worth mentioning that population based meta-heuristic algorithms have been widely applied to solve real-word MOPs since they can handle nonlinear, multimodal and discontinuous problems. Among them, PSO is a promising swarm intelligence technique that is inspired by the behavior of birds and fish search for food. Due to it easy implementation and fast convergence toward the global optima, it has gained a great interest of researchers and has been extensively applied to many single-objective engineering applications. The promising results obtained by application of PSO in single-objective problems has proven its

effectiveness and efficiency [8]. Based on this motivation, many studies have extended PSO to solve MOPs and numerous of multi-objective PSO (MOPSO) algorithms were proposed and achieved great success [9]–[15].

In order to apply PSO to solve MOPs, we must address the following two fundamental issues. Firstly, how to select global best solutions (leaders) to guide the swarm search. As there is no single solution that outperforms all other solutions in all objectives, thus many nondominated solutions can be considered as candidates to be selected as leaders. To accomplish this task, one must adopt a suitable selection technique that take into consideration of solutions diversity and convergence speed. It should be noted that a suitable leader for one particle may not be suitable to others. Therefore, in order to enhance the exploration of the search space, it is advisable to a use competition technique between random nondominated solutions to select leader for each particle of the swarm. Secondly, how to keep balance between convergence and diversity of particles. Since PSO based multi-objective optimization algorithms have a fast convergence speed, they are more likely to be trapped in local optima [16]. Therefore, achieving balance between convergence and diversity is crucial. Moreover, in the external elitism archive based algorithms, the update strategy of the archive has a great influence on search performance. Thus, an efficient update mechanism is required to enhance the search performance and maintain balance between convergence and diversity.

To cope up with MOPs, numerous variants of MOPSO have been extensively developed from different aspects. The first category is to apply dynamic/adaptive mechanism for parameter selection (e.g., inertia weight ω or acceleration coefficients c_1 and c_2) in MOPSO and thus to keep balance between convergence and diversity [17], [18]. The second category is to incorporate multiple swarms to enhance the search capability in multimodal landscapes [19]–[21]. In the third category, the authors make use of hybridizing PSO with other meta-heuristic algorithms to achieve balance between global and local search [22]–[24]. The fourth category is to enhance the ranking scheme of the algorithm, in which an elitism archive is adopted to store elite solutions selected by the ranking scheme, and these solutions are used as candidates for leaders. Two representatives of this category are preference order ranking [25] and global margin ranking [9]. The last category is based on how Pareto fronts are defined. This category can be roughly divided into two subcategories; the first sub-category determines personal best and global best based on Pareto dominance [26]. The second sub-category adopts the decomposition strategy to transform MOPs into a set of single-objective problems SOPs, where the single-objective PSO is directly applied to solve the multi-objective problem [27], [28].

Although all aforementioned improvements on MOPSO, there is still a need to apply a sufficient approach to select leaders of the swarm. Due to the great influence of leaders in the search direction and convergence speed, randomly selecting leaders from the elitism archive may not effectively enhance the search speed and also may cause the algorithm to get trapped in local optima in complex multimodal landscapes [16]. Moreover, as most of domination-based multi-objective optimization algorithms make use of infinite external elitism archive to store nondominated solutions found so far, the update mechanism of this archive is crucial. When the archive reaches the maximum size limit, in order to add new nondominated solution into the archive, the nondominated solution in the archive need to be truncated considering the convergence, diversity and spread preservation [24]. Motivated by above mentioned, and to further improve the convergence speed and diversity preservation of PSO in solving MOPs, we propose a novel variant of MOPSO based on a newly developed archive update mechanism, termed MOPSONN. Moreover, to enhance the selection of leaders we incorporate the pairwise competition inspired by the recently developed competitive swarm optimizer [29]. The main contributions of this paper are as follows:

1) Vicinity distance metric is suggested for the updating of the external archive at the exploration phase. At the end of each iteration, nondominated solutions are added into the archive. When the archive exceeds it maximum size limit, vicinity distance metric is used compare the crowding of solutions in the archive and remove the most crowded one. Then update the crowding of all solutions in the neighborhood of the removed solutions (i.e. all solutions whose crowding is changed after removing a solution need to update according to vicinity distance). This procedure achieves better diversity and speared preservation in exploration phase.
2) Two new rules, Max-cost rule and Sum-of-cost rule, are proposed for the updating of external archive at exploitation phase. Max-cost rule is applied to the new nondominated solutions before adding them into the archive. This step is beneficial in maintaining only elite nondominated solution in the archive. Thus, it increases the convergence speed. After that, Sum-of-cost rule is implemented to the two most crowded solutions in the archive and remove the one with less total fitness value (in a minimization problem). Similar to vicinity distance update procedure, all solutions whose crowding distance is changed after removing a solution need to be updated by calculating their distance to their new closest solutions. By applying these two rules, we can achieve better balance between convergence and diversity.
3) A novel multi-objective particle swarm optimization algorithm, called MOPSONN is proposed based on a new archive updating procedure and pairwise competition mechanism. In MOPSONN, all elite nondominated solution in the archive are preserved by means of the proposed archive updating mechanism. Consequently, MOPSONN has a significant convergence speed while maintaining obtained solution well-spread along the true Pareto front.

The performance of the proposed MOPSONN is verified using several benchmarks and compared with seven state-of-the-art multi-objective algorithms, namely, four PSO based multi-objective algorithms, SMPSO [30], dMOPSO [31], MMOPSO [32] and CMOPSO [16], and three MOEAs, NSGA-II [33], MOAED [34] and NSGA-III [35]. The experimental results show that the proposed algorithm has the ability to fast converge towards the true Pareto front in a relatively small number of function evaluations. By contrast, in many test functions, the compared algorithms fail even to approach the true Pareto front in such number of function evaluations.

II. RELATED WORK

A. PARTICLE SWARM OPTIMIZATION

PSO is a nature-inspired optimization algorithm introduced by Eberhart and Kennedy in 1995 [36]. PSO was developed inspired by the social behavior of swarm of animals like birds. In PSO, individuals are called particles and each particle represents a potential solution. The swarm consists of a set of particles which flies through the search space searching for optimal solution, like birds in folks searching for food. Let $\boldsymbol{X}_i(t) = (x_{i1}(t),\ x_{i2}(t), \ldots, x_{in}(t))$ and $\boldsymbol{V}_i(t) = (\mathrm{v}_{i1}(t), \mathrm{v}_{i2}(t), \ldots, \mathrm{v}_{in}(t))$ be the position and velocity of particle p_i at time t in an n-dimensional hyperspace. Each particle p_i memorizes its historical best solutions as denoted by $pbest_i$ and the best among all particles is acknowledged as global best solution $gbest$. Each particle in the swarm is evolved based on its own experience $pbest_i$ and the positional information from the global leader in the swarm $gbest$ to update the position and velocity as defined by

$$\boldsymbol{V}_i(t+1) = \omega \boldsymbol{V}_i(t) + c_1 r_1(\boldsymbol{X}_{pbesti} - \boldsymbol{X}_i(t)) + c_2 r_2(\boldsymbol{X}_{gbest} - \boldsymbol{X}_i(t)) \quad (1)$$

$$\boldsymbol{X}_i(t+1) = \boldsymbol{X}_i(t) + \boldsymbol{V}_i(t+1) \quad (2)$$

where ω is the inertia weight, t is the generation number, c_1 and c_2 are the learning factors of the personal best position and global best position, respectively, and $r_1, r_2 \in [0, 1]$ are two random numbers [32].

B. EXISTING MOPSO ALGORITHMS

The first MOPSO variant was proposed by Coello *et al.* in [37]. The authors incorporated the concept of Pareto dominance into PSO to allow the algorithm to handle problems with multiple objectives. In the algorithm, an external archive was adopted to store nondominated solutions obtained in each iteration as global best particles. In spite of the fact that the proposed MOPSO algorithm has proven competitive performance in solving MOPs in comparison with classical multi-objective evolutionary algorithms (MOEAs) such as PAES [38] and NSGA-II [33], it was unable to solve MOPs with complex landscapes.

An improved PSO base multi-objective algorithm, called OMOPSO, was suggested by Sierra and Coello Coello [26], which uses Pareto dominance and crowding factor to identify list of available leader solutions. Different mutation operators were suggested for different sub-divisions of the swarm to enhance the algorithm search capability. Moreover, the algorithm keeps all nondominated solution obtained by the swarm in an external archive and ε-dominance is used to fix the size of the archive.

A speed-constrained multi-objective PSO algorithm, called SMPSO, was presented by Nebro *et al.* [30], which uses a procedure to limit the velocity of the particles by producing new effective particle solutions to handle MOPs with multimodal landscapes. Furthermore, polynomial mutation as a turbulence factor and an external archive are utilized to collect the nondominated solutions obtained during the search. As reported in [30], most of MOPSOs fails in solving multi-frontal problems due to the fact that velocities in such algorithms are too high. Thus, the speed constrains is and effective approach to improve the performance of MOPSOs.

Motivated by the decomposition approach adopted in MOEA/D [34], Peng and Zhang presented the first attempt to embed the decomposition mechanism into PSO based multi-objective optimization algorithms, called MOPSO/D [39]. The authors utilized the framework of MOEA/D and replaced the genetic operator with PSO search approach. An external archive based on ε-dominance is used to keep all global best particles of each SOP. An improved version of MOPSO/D is presented by Moubayed et al, called SDMOPSO [40]. In SDMOPSO, global best is only picked from the neighborhood of the particle and a crowding archive is adopted to preserve diversity of swarm leaders.

Martinez and Coello Coello [31] also suggested a multi-objective particle swarm optimizer based on decomposition approach, called dMOPSO, in which global particles are determined based on the scalar aggregated values. In the algorithm, the particles are updated using global best particles. Moreover, a memory re-initialization strategy is used when a particle reaches a certain age. The main aim of this approach is to preserve diversity and to avoid trapping in local fronts. However, as reported in Moubayed *et al.* [41], dMOPSO may fail to cover the entire PF in some complicated MOPs due to the absence of dominance relation.

A coevolutionary technique based multi-objective PSO algorithm, called CMPSO, was proposed by Zhan *et al.* [21]. In the algorithm, each swarm is associated with only one objective and an external archive is used to share information between the different swarms. This technique aimed at enhancing the diversity and avoiding local PFs. Moreover, there are some PSO based multi-objective algorithms proposed on basis of hybridization and multiple search strategies. A hybrid teaching learning based particle swarm optimization, called HTL-MOPSO, was proposed by Cheng *et al.* for solving MOPs [24]. The algorithm in [24] incorporates the teaching learning approach with the PSO search approach aiming at achieving good convergence and well-spread nondominated solutions along the true PF. The authors use external archive along with circular crowded sorting to store nondominated solutions obtained during the swarm search.

Algorithm 1 Nearest Neighbor Distance

For each $A_i \in A$ **do**

Compute Euclidean distance D_{ij} between A_i and A_j, $i \neq j$

$DNN_i^1 = \min(D_i)$;

$NN_i = \{j | D_{ij} = DNN_i^1\}$;

Attach DNN_i^1 and NN_i to A_i;

End For

In contrast to most existing MOPSO algorithms where the particles are updated according to only one search strategy, Lin *et al.* [32] proposed a MOPSO algorithm with multiple search strategies, called MMOPSO. In MMOPSO, two search strategies are suggested to update the velocity of each particle to enhance the convergence speed and maintain diversity. The algorithm is based on decomposition approach in which MOPs are transformed into a set of aggregation problems.

A recent MOPSO algorithm based on the competitive swarm optimizer, called CMOPSO, is presented by Zhang *et al.* [16]. The authors make use of a competitive mechanism inspired by competitive swarm optimizer [29] to replace the traditional velocity update equations of PSO. The selections of leaders are performed on the basis of the pairwise competitions performed in the current swarm at each iteration.

Based on the above, a multi-objective particle swarm algorithm based a novel archive updating procedure and pairwise competition is proposed to effectively enhance the swarm convergence speed and diversity preservation in solving MOPs. A detailed description of the proposed algorithm is presented in the following section.

III. THE PROPOSED MOPSONN ALGORITHM

In this section we describe the proposed algorithm MOPSONN. The algorithm begins with random initialization of all particles and an initial archive A is generated based on Pareto dominance. The Euclidean distance between each solution in A and its nearest neighbor is calculated forming a distance vector D_i for each solution in A. Each solution is attached a value of its shortest distance to other solutions DNN_i^1 and the index of the corresponding neighbor NN_i as described in Algorithm 1. Thereafter, velocities and positions of the swarm are updated according to equations (1), (2), and leaders X_{gbest} are selected to guide the search in the next generation.

A. SELECTION OF PERSONAL BEST AND GLOBAL BEST

In MOPSONN, the personal best position X_{pbesti} is updated after updating the particle velocity and position using (1) and (2) iteratively. If the new position X_{newi} dominates X_{pbesti}, it replaces the latter; otherwise the current X_{pbesti} is kept; if neither of them dominates the other, then we randomly selecting one of them. For selection of leader X_{gbest} for each particle in the swarm, we adopt the competitive mechanism proposed in [29], which is consist of three main components, elite particle selection, pairwise selection and particle learning.

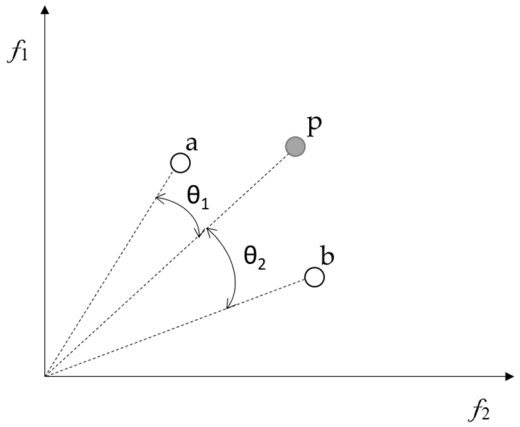

FIGURE 1. Illustrative example of pairwise competition. a and b are two random elite particles and p is the particle to be updated.

In our algorithm, all solutions in A are sorted in a descending order according to their DNN_i^1. A predefined number of particles γ is selected from the top particles of the archive A to form the elite set E. This step is supposed to preserve diversity by guiding the search towards the less occupied regions. After the elite particle set is created, pairwise competition is performed to select a leader for each particle in the swarm, in which two particles a and b are randomly selected from the elite particle set. The angle between a, b and p is calculated, respectively. The elite particle with the smaller angler with p wins the competition and is used as leader for particle p. An illustrative example of pairwise competition is described in Figure 1 where a and b are two random elite particles selected for pairwise competition while p the particle to be updated. As it can be seen, the angle θ_1 between a and p is smaller than the angle θ_2 between b and p, such that the elite particle a is used as leader for particle p. After determining the winner of elite particles, the particle p will be updated according to (1) and (2), where X_{gbest} is the position of the winner particle. A detailed description of elite particles selection, pairwise competition, and particles update is presented in Algorithm 2.

B. UPDATE OF EXTERNAL ARCHIVE

As the number of nondominated solutions in the external archive is finite and determined by the size of population, and there are an m number of new nondominated solutions $\boldsymbol{S} = (s_1, s_2, \ldots, s_m)$ added into A in each generation. The swarm search is divided into two phases, exploration and exploitation according to a threshold α of the generation index. It should be noted that the appropriate setting of α can keep good balance between convergence and diversity. In each phase, the archive update is performed in a different approach. Firstly, the vicinity distance metric is adopted to compare the most crowded solutions in the archive, this approach is useful in the exploration phase to maintain diversity. Secondly, the Max-cost rule is applied along with Sum-of-cost rule are applied to keep balance between convergence

Algorithm 2 Elite Particles Selection, Pairwise Competition and Particles Update

Input: A (current archive), P (current positions), V (current velocities), γ (size of elite particles set)

Output: P' (new positions)

/*Elite particles selection*/

Sorted all particles in A in a descending order according to their DNN_i^1;

$E \leftarrow$ select top γ particles from A to form the elite particles set;

/*Pairwise competition*/

For each $p_i \in P$ **do**

Randomly select two elite particles a and b from E;

Calculate the angle θ_1 between a and p_i, and θ_2 between b and p_i;

If $\theta_1 < \theta_2$ **then**

$X_{gbest} \leftarrow a$;

Else

$X_{gbest} \leftarrow b$;

End if

/*Particles update*/

$v_i' \leftarrow$ update the velocity of p_i according to equation (1);

$p_i' \leftarrow$ update the position of p_i according to equation (2);

$P' \leftarrow P' \cup \{p_i'\}$;

End for

Return P'

Algorithm 3 Archive Updating According to Vicinity Distance Metric

Input: A (current archive), S (new nondominated solutions)

Output: A' (new archive)

$A' \leftarrow A \cup S$;

For each $A_i \in A'$ **do**

Update DNN_i^1 and NN_i for A';

End for

While A' is larger than nA, **do**

Find A_k' with the minimum DNN_k^1;

$j \leftarrow NN_k$ let j be the index of the nearest neighbor of solution A_k';

Find DNN_k^2 and DNN_j^2;

Compute vicinity distance VD for A_k and A_j;

If $VD_k \leq VD_j$ **do**

Remove A_k from A';

Find A_m which its attached nearest neighbor NN_m is k;

Update DNN_j^1, DNN_m^1, NN_j and NN_m;

Else

Remove A_j from A'**;**

Find A_m which its attached nearest neighbor NN_m is j;

Update DNN_k^1, DNN_m^1, NN_k and NN_m;

End if

End while

Return A'

and diversity in the exploitation phase. The following subsections will explain in details how these two approaches are implemented.

1) VICINITY DISTANCE

As described in Algorithm 1, all solutions in A are assigned a distance value DNN_i^1 to their nearest neighbors, which is calculated using Euclidean distance metric. When finding the shortest DNN_k^1 in A, there will always be two solutions having the same distance value due to the symmetry property of the metric [42]. In other words, let A_k be the kth solution in A, and let DNN_k^1 and NN_k be its distance to the nearest neighbor and the index of the nearest neighbor, respectively. Therefore, A_k and A_{NN_k} will have the same distance to each other, which is DNN_k^1. To compare the crowding of A_k and A_{NN_k}, we use vicinity distance metric which takes into account the distance of second nearest neighbor DNN_i^2 for both solutions. More formally Vicinity distance is defined as:

$$VD_i = \prod_{k=1}^{m} DNN_i^k \tag{3}$$

where VD_i is the vicinity distance of particle i, DNN_i^k is the Euclidean distance of particle i to it its kth neighbor, $m = 2$ is the number of neighbors. If the vicinity distance of particle k VD_k is smaller than the vicinity distance of its nearest neighbor VD_{NN_k}, we consider A_k as the most crowded solution and remove it from A, and vice versa. When removing A_k from A, the distances of the remaining neighbors of A_k will be influenced. Therefore, the distances value DNN_i^1 (where i is the index of the remaining neighbor) and the attached indices of the remaining nearest neighbors NN_i must be updated iteratively until the number of solutions in A is equal to the maximum size of the archive nA. If there exists another solution A_m such that its attached nearest neighbor index NN_m is the same as the index of the removed solution k, then DNN_m^1 and NN_m must be updated as well. The pseudo-code of this step is presented in Algorithm 3.

The benefit of using vicinity distance against crowding distance is that the solutions whose ranks DNN_i^1 and nearest neighbor indices NN_i are affected by removing a solution from A are easy to be determined. Thus, Euclidean distance is recalculated only for the removed solution neighbors instead whole set of nondominated solutions after each solution removal.

2) MAX-COST RULE

The second approach is performed to tackle the problem of losing better candidates when updating the archive after

adding new nondominated solutions S. When randomly removing solutions from the archive or when adopting the crowding distance as a measure to remove extra solutions, we do not have a criterion that can compare nondominated solutions to each other. Thus, the newly added solutions may be worse than the removed solution in terms of convergence. For this reason, we adopt the following procedure to tackle this problem.

In exploitation phase, before adding new nondominated solutions into the archive the fitness values of each objective for all new nondominated solutions are tested against the maximum cost value of each objective. In an m-objective minimization problem let $\boldsymbol{F_i} = (f_{i1}, f_{i2}, \ldots, f_{im})$ be the fitness values of newly added solution s_i, and let $\boldsymbol{F}_{max} = [f_{max1}, f_{max2}, \ldots, f_{maxm}]$ be a vector consists of the maximum fitness values of each objective in the archive A. As shown in Algorithm 4, F_i is added to A if and only if F_i is less than or equal to F_{max} for all m objectives.

3) SUM-OF-COST RULE

After evaluating all new nondominated solutions against the Max-cost rule, the number of nondominated solutions in the external archive may exceeds the maximum size limit. To maintain the number of nondominated solutions within the maximum size of the archive in a way that assure preserving balance between convergence and diversity, another approach termed as Sum-of-cost rule is proposed. As presented in Algorithm 4, let A_i and A_j be the two solutions with the minimum Euclidean distance to each other (i.e. $DNN_i^1 = DNN_j^1$), and $\boldsymbol{F}_i$ and $\boldsymbol{F}_j$ are their fitness values, respectively. In an m-objective minimization problem, if

$$\sum_{k=1}^{m} \boldsymbol{F}_i \leq \sum_{k=1}^{m} \boldsymbol{F}_j, \quad \boldsymbol{F}_i = (f_{i1}, f_{i2}, \ldots, f_{im}), \quad \boldsymbol{F}_j = (f_{j1}, f_{j2}, \ldots, f_{jm}) \tag{4}$$

then A_i is remained and A_j is removed. Similarly, the rank DNN_i^1 and the attached nearest neighbors' indices NN_i of the any influenced solution must be updated iteratively until the number of solution in A is equal to the defined size of nA. A detailed pseudo-code of Max-cost rule and Sum-of-cost rule is presented in Algorithm 4. The major advantage of using such mechanism is that it overcomes the issue of convergence in MOPs algorithms which use the concept of nondomination sorting in an external archive. Moreover, the experimental results show that it enhances the ability of the algorithm to fast convergence towards the true Pareto front while preserving a relatively high diversity.

Algorithm 4 Archive Updating According to Max-Cost Rule and Sum-of-Cost Rule

Input: A (current archive), S (new nondominated solutions)

Output: A' (next archive)

/* Max-cost rule */

Find $\mathbf{F}_{max}$ from the current archive A;

For each $s_i \in S$ **do**

If all $f_j(s_i) \leq f_{\max j}$, $1 < j < m$, where j is the objective index **do**

$A' \leftarrow A \cup \{s_i\}$;

Else

reject s_i;

End for

/* Sum-of-cost rule */

While A' is larger than nA, **do**

Find A'_k with the minimum DNN_k^1;

$j \leftarrow NN_k$ let j be the index of the nearest neighbor of solution A'_k;

Calculate $\sum_{k=1}^{m} \mathbf{F}_i$ and $\sum_{k=1}^{m} \mathbf{F}_j$;

If $\sum_{k=1}^{m} \mathbf{F}_i \leq \sum_{k=1}^{m} \mathbf{F}_j$ **do**

Remove A_j from A';

Find A_m which its attached nearest neighbor NN_m is j;

Update DNN_i^1, DNN_m^1, NN_i and NN_m;

Else

Remove A_i from A';

Find A_m which its attached nearest neighbor NN_m is i;

Update DNN_j^1, DNN_m^1, NN_i and NN_m;

End if

End while

Return A'

C. THE COMPLETE MOPSONN ALGORITHM

The above subsections have described the procedure of selection of leader and archive update, which compose the main component of MOPSONN. Furthermore, the complete pseudo-code of MOPSONN algorithm is presented in Algorithm 5, where N is the population size and α is a threshold that control the two search phases, exploration phase and exploitation phase. The algorithm begins by randomly initializing the swarm with N particles and the external archive is initialized to be empty. After evaluating the fitness function for all particles, the nondominated sorting is employed to select initial nondominated solutions to be stored in the archive. Then, the algorithm turns into loop of evolution process until the termination criterion is fulfilled.

To generate elite particles set, the Euclidean distance is employed to calculate the distance of each particle in the archive to its corresponding nearest neighbor such that each particle in the archive is attached a value that represents its distance to the nearest particle as well as the index of the nearest particle as described in Algorithm 1. Thereafter, all particles in the archive are stored in a descending order according to the distance to their nearest particles. An elite particles set is selected from the top γ particles in the archive in which the leaders are chosen according to the pairwise competition approach. Then, the particles are updated using formula (1) and (2) as introduced in Algorithm 2. After

Algorithm 5 General Framework of MOPSONN

Input: N (population size)

Output: A (non-dominated solutions in the archive)

$P \leftarrow$ random Initialize (N);

$V \leftarrow$ random Initialize (N);

$A \leftarrow$ check Dominance (P);

Compute nearest neighbor distance for all solutions in A; **(Algorithm 1)**;

While termination criterion is not fulfilled **do**

$P' \leftarrow$ update particles (A, P, V, γ), according to Elite particles selection, Pairwise competition and Particle updates **(Algorithm 2)**;

$S \leftarrow$ check Dominance (P');

If the number of the index of current generation is less than a threshold $t < \alpha$ **do**

$A' \leftarrow$ update archive (A, S), according to Vicinity distance value **(Algorithm 3)**;

Else

$A' \leftarrow$ Update archive (A, S), according to Max-cost rule and Sum-of-cost rule **(Algorithm 4)**;

End If

End while

Return A'

evaluating the swarm particles, the archive updating process is accomplished in two ways determined by the predefined threshold α. In the exploration phase (i.e. when the iteration index is less than α), the archive is updated by applying the vicinity distance metric as illustrated in Algorithm 3. After that, in the exploitation phase (i.e. when the iteration index is equal or larger than α), Max-cost rule and Sum-of-cost rule are employed to update the nondominated solution in the archive as presented in Algorithm 4. At the end of algorithm, the nondominated solutions in the external archive are reported as the final approximated Pareto front.

IV. EXPERIMENTAL STUDY

A. STANDARAD BENCHMARK PROBLEMS

A total of fifteen benchmarks from two test suits, ZDT [43] and DTLZ [44] are used to evaluate the performance of the algorithms, where ZDT1 to ZDT4 and ZDT6 are bi-objective problems and DTLZ2, DTLZ4 to DTLZ7 are two-/three-objective problems. For all ZDT problems, the number of decision variables is 30. For bi-objective DTLZ problems the number of decision variables is 11 except for DTLZ7 is 21. For the three-objective DTLZ problems, the number of decision variables is set to 12. For each benchmark, 30 independent runs are conducted using a personal computer with an Intel Core i7-8650 1.90GHz 2.11GHz CPU and windows 10 operating system. The mean values and standard deviations (std) for each test instance are reported, where the best results are highlighted with bold font in the comparison tables.

TABLE 1. The parameter settings for all the algorithms.

Algorithms	Parameter settings
dMOPSO	$\omega \in [0.1, 0.5]$,
SMPSO	$\omega \in [0.1, 0.5]$, $c_1, c_2 \in [1.5, 2.5]$, $p_m = 1/n$, $\eta_m = 20$
MMOPSO	$\omega \in [0.1, 0.5]$, $c_1, c_2 \in [1.5, 2.0]$, $pc = 0.9$, $p_m = 1/n$, $\eta_c = 20$, $\eta_m = 20$, $\delta = 0.9$
CMOPSO	γ =10
NSGA-II	$p_c = 0.9$, $p_m = 1/n$, $\eta_c = 20$, $\eta_m = 20$
NSGA-III	$p_c = 1$, $p_m = 1/n$, $\eta_c = 30$, $\eta_m = 20$
MOEA/D	$pm = 1/n$, $\eta m = 20$, $T = 20$, $nr = 2$
MOPSONN	ω =0.5, ω_{damp}=0.99, c_1=1, c_2=2, γ =10, α=0.8

B. PERFORMANCE METRICS

In order to examine the performance of multi-objective optimization we consider the following two Pareto Frontier quality metrics:

1) INVERTED GENERATIONAL DISTANCE

In multi-objective optimization, the inverted generational distance (IGD) is used to measure the convergence and to assess how far are the obtained nondominated solutions from true Pareto Frontier assuming it is known [32]. This metric is useful when the solutions obtained are good in terms of optimality but do not cover the entire space. IGD can be defined as:

$$IGD(A', A) = \frac{\sum_{i=1}^{|A'|} d(A'_i, A)}{|A'|} \tag{5}$$

where A' is a subset uniformly selected from the true Pareto front, A is the approximated set, and $d(A'_i, A)$ is the Euclidean distance between each solution in the obtain Pareto Front font so far and its nearest solution in the true Pareto Front. A value of zero indicates that all optimal solutions are within the true Pareto Front.

2) SPACING

Spacing metric evaluates the spread of solutions throughout the Pareto Front found so far. it measures the range variance of the distance between each solution and its neighbors [45]. A value of zero indicates that all Pareto-optimal solutions are uniformly distributed in the objective space.

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d} - d_i)^2} \tag{6}$$

where $d_i = \min_{i,i\neq j} \sum_{k=1}^{m} \left| f_k^i(\vec{x}) - f_k^j(\vec{x}) \right|$, $i, j = 1, \ldots, n$, $\bar{d}$ is the mean of all d_i, m is the number of objectives and n is number of the Pareto-optimal solution found so far.

The mean values and standard deviations (Std.) of IGD and Spacing metrices are collected for each test instance. Moreover, the Wilcoxon rank sum test is performed at a

TABLE 2. IGD results of the proposed MOPSONN and four existing multi-objective PSO algorithms, SMPSO, dMOPSO, MMOPSO and CMOPSO on ZDT and DTLZ test suits. The best values are highlighted with bold fonts.

Problem		Obj.	SMPSO	dMOPSO	MMOPSO	CMOPSO	MOPSONN
ZDT1	Mean	2	3.18E+00+	1.80E-01+	3.35E-02+	1.64E-01+	**4.35E-03**
	Std.		3.72E+00	2.14E-01	1.52E-01	1.14E-01	4.40E-03
ZDT2	Mean	2	4.55E+00+	5.55E-01+	3.05E+00+	4.66E-01+	**4.27E-03**
	Std.		4.53E+00	1.66E-01	5.29E+00	1.24E-01	1.53E-04
ZDT3	Mean	2	4.54E+00+	9.89E-02+	6.72E-01+	3.84E-01+	**4.84E-03**
	Std.		3.81E+00	1.55E-01	1.68E+00	1.74E-01	2.46E-04
ZDT4	Mean	2	1.18E+02+	6.48E+00+	9.84E+01+	1.73E+02+	**4.08E-03**
	Std.		3.84E+01	1.97E+01	2.56E+01	2.65E+01	2.55E-04
ZDT6	Mean	2	1.05E+01+	4.22E-01+	5.48E-01+	3.94E+00+	**2.42E-03**
	Std.		2.53E+00	2.31E-01	1.74E+00	9.46E-01	1.35E-04
DLTZ2	Mean	2	5.39E-03+	3.52E-02+	5.31E-03+	**4.50E-03-**	5.12E-03
	Std.		3.08E-04	3.72E-03	3.28E-04	8.10E-05	2.02E-04
DLTZ4	Mean	2	3.54E-01+	1.15E-01+	1.03E-01+	3.49E-01+	**5.21E-03**
	Std.		3.70E-01	3.59E-02	2.55E-01	3.74E-01	2.33E-04
DLTZ5	Mean	2	5.36E-03=	3.37E-02+	5.22E-03=	**4.52E-03-**	5.34E-03
	Std.		2.61E-04	5.04E-03	2.36E-04	9.52E-05	2.72E-04
DLTZ6	Mean	2	3.79E-01+	7.55E-03+	5.80E-03+	**4.13E-03-**	4.39E-03
	Std.		4.68E-01	6.07E-03	3.78E-04	3.77E-05	8.50E-05
DLTZ7	Mean	2	3.83E-01+	5.15E-02+	1.51E-01+	3.54E-02+	**5.10E-03**
	Std.		2.50E-01	7.77E-02	2.10E-01	1.11E-01	2.86E-04
DLTZ2	Mean	3	8.88E-02+	1.44E-01+	7.25E-02+	**5.85E-02-**	6.39E-02
	Std.		5.89E-03	9.43E-03	2.20E-03	1.02E-03	2.66E-04
DLTZ4	Mean	3	3.93E-01+	2.65E-01+	7.09E-02+	9.08E-02+	**6.72E-02**
	Std.		1.67E-01	2.13E-02	2.79E-03	1.62E-01	3.26E-03
DLTZ5	Mean	3	5.51E-03+	4.11E-02+	6.28E-03+	6.59E-03+	**5.09E-03**
	Std.		3.20E-04	6.91E-03	7.11E-04	5.79E-04	2.65E-04
DLTZ6	Mean	3	1.08E+00+	3.29E-02+	6.74E-03+	1.13E-01+	**4.78E-03**
	Std.		7.62E-01	2.11E-04	8.25E-04	3.03E-01	2.04E-04
DLTZ7	Mean	3	2.14E-01+	2.40E-01+	2.03E-01+	1.46E-01+	**5.68E-02**
	Std.		1.58E-01	2.65E-01	2.01E-01	1.90E-01	1.35E-03
+/=/-			14/1/0	15/0/0	14/1/0	11/0/4	\

significance level of 0.05 to examine that where the results obtained by MOPSONN are statistically different from that obtained by other algorithms. The symbols '+', '=' and '−' indicate that the result obtained by MOPSONN is significantly better, statistically similar, and significantly worse than that obtained by the compared algorithm, respectively.

C. EXPERIMENTAL SETTING

The performance of the proposed MOPSONN algorithm is verified by comparison with four existing multi-objective PSO algorithms, SMPSO, dMOPSO, MMOPSO and CMOPSO, and three popular MOEAs, NSGA-II, MOAED and NSGA-III. It is noted that the source code of all compared algorithms can be found in PlatEMO [46]. For fair comparison, the parameter setting of all compared algorithms are set to the recommended values in the original papers as summarized in Table 1. For SMPSO, the coefficient c_1 and c_2 are randomly chosen from [1.5, 2.5] and the inertia weight ω is also randomly chosen from [0.1,0.5]. In MMOPSO and dMOPSO, the coefficient c_1 and c_2 are randomly chosen from [1.5, 2.0] and the inertia weight ω is also randomly chosen from [0.1,0.5]. In CMOPSO the parameter γ is set to 10. In NSGA-II, the crossover probability p_c and the mutation probability p_m are set to be 0.9 and $1/n$ (where n is the number of decision variables), respectively, and distribution index for crossover and mutation operators are $\eta_c = 20$ and $\eta_m = 20$, respectively. In MOEA/D, the mutation probability p_m is set to $1/n$, distribution index for mutation operators η_m is set to be 20. The size of the neighborhood in the weight coefficients T is set to be 20 and the maximum number of parent solutions that are replaced by each child solution n_r is 2. In NSGA-III, the crossover probability p_c and the mutation probability

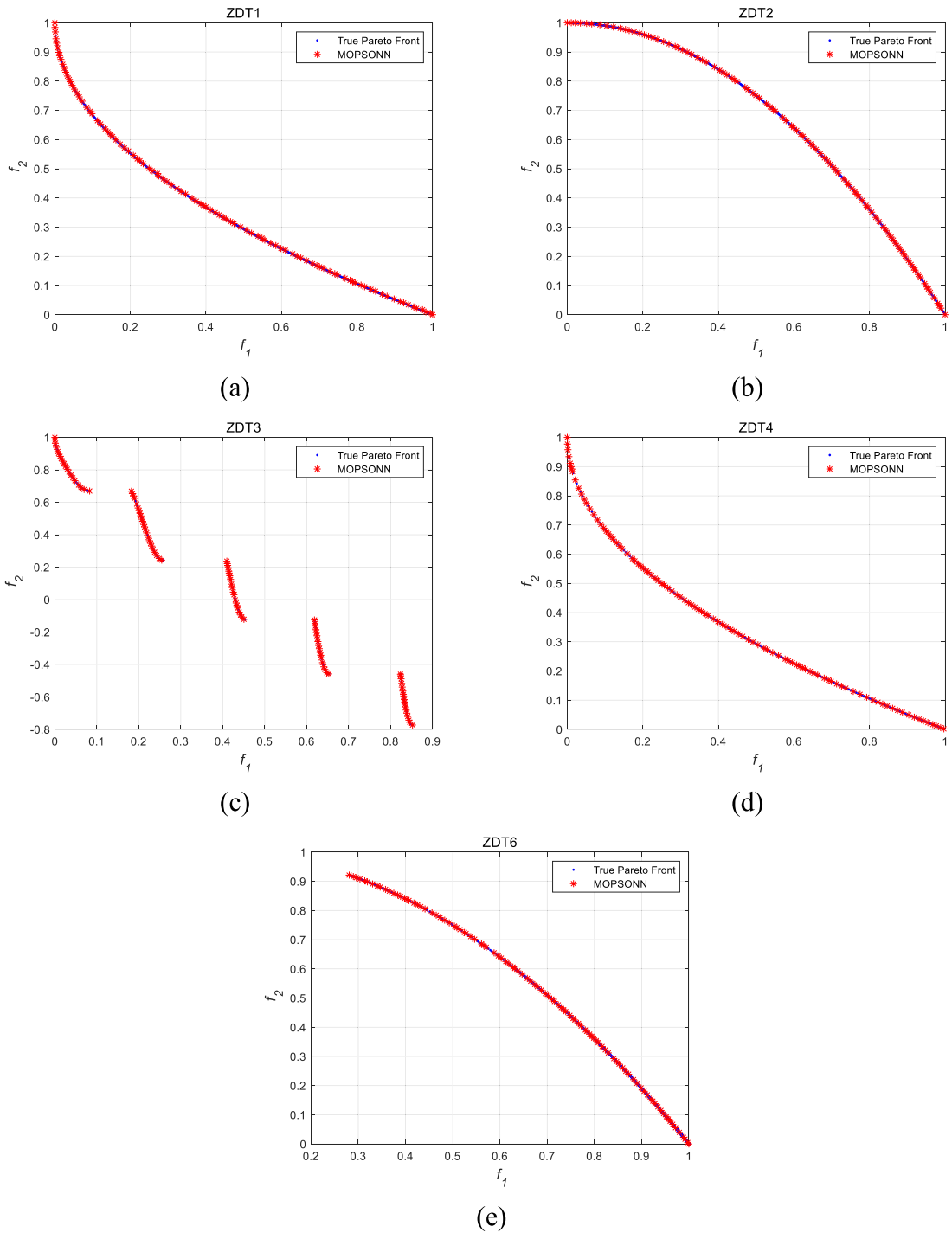

FIGURE 2. The Pareto front of ZDT test suit obtained by MOPSONN. (a) ZDT1; (b) ZDT2; (c) ZDT3; (d) ZDT4; (e) ZDT6.

p_m are set to be 1 and $1/n$, respectively, and distribution index for crossover and mutation operators are $\eta_c = 30$ and $\eta_m = 20$, respectively. In the proposed MOPSONN, the coefficient c_1 and c_2 are set to1and 2, respectively, the inertia weight ω is to be 0.5 with an inertia weight damping rate $\omega_{\text{damp}} = 0.99$, the number of elite particles set γ is set be 10 and the threshold α is set to be 0.8. It should be noted that the size of both population and the external archive are set to be 100 for all algorithms. The number of generations is adopted as a termination criterion for all algorithms. The maximal number of generations is set to 50 for all ZDT test instances, and to 100 for all bi-objective DTLZ test instances. For three-objective DTLZ problems, the maximal number of generations is set to 100 in DTLZ2, DTLZ6 and DTLZ7 and to 250 in DTLZ4 and DTLZ5. It is noted that the IGD of each benchmark is calculated using roughly 5000 and 10000 points

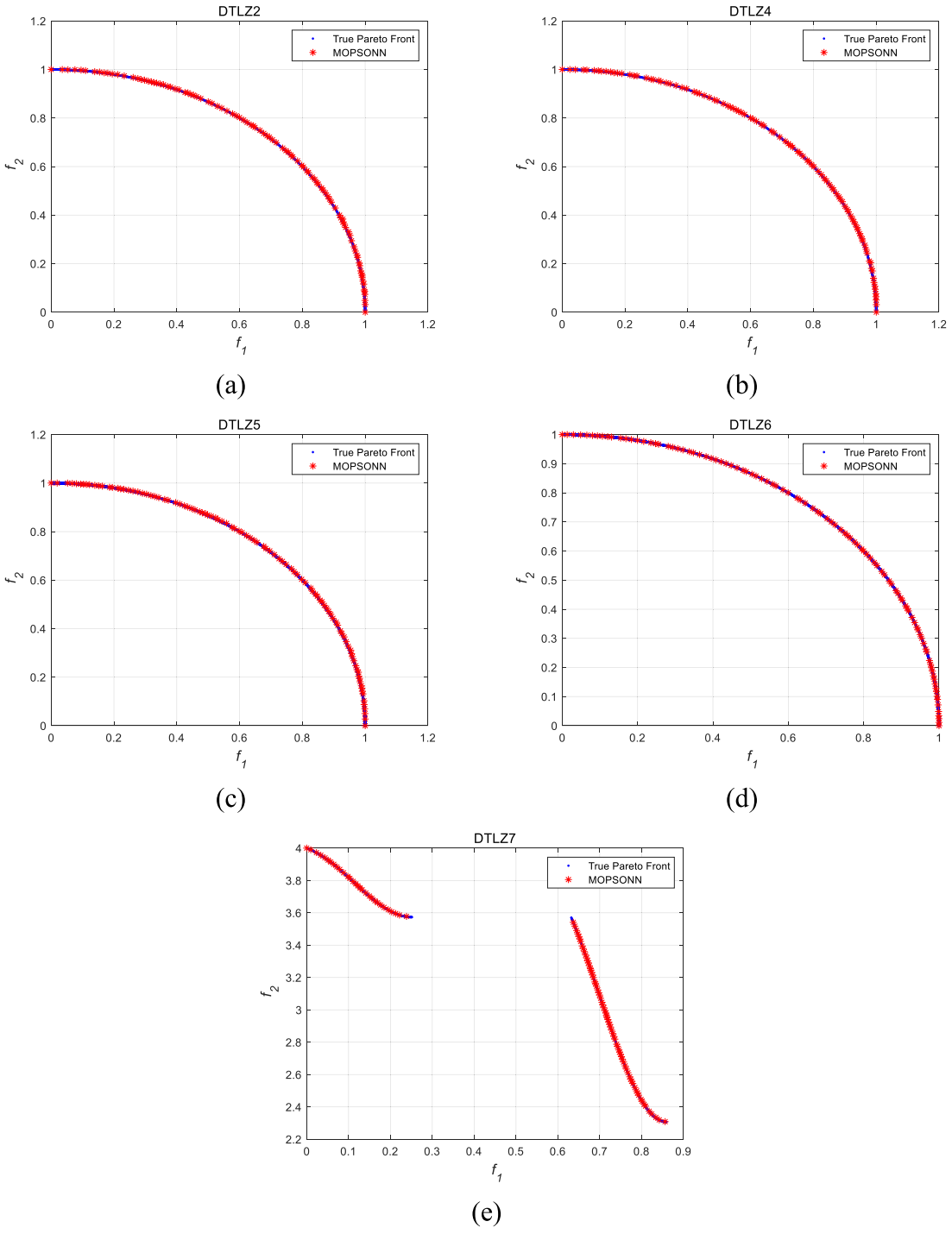

FIGURE 3. The Pareto front of bi-objective DTLZ test suit obtained by MOPSONN. (a) DTLZ2; (b) DTLZ4; (c) DTLZ5; (d) DTLZ6; (e) DTLZ7.

uniformly sampled on the Pareto fronts for bi-objective and three-objective test problems, respectively.

D. COMPARISON WITH EXSITING MULTI-OBJECTIVE PSO ALGORITHMS

Table 2 summarizes the mean and standard deviation of IGD of SMPSO, dMOPSO, MMOPSO and CMOPSO compared with MOPSONN for all test problems. It can be observed that MOPSONN obtains the best results on all bi-objective test problems of ZDT test suit in such relatively low number of evaluations. Figure 2 shows the nondominated solutions obtained by the proposed MOPSONN on ZDT1 to ZDT4 and ZDT6. For the bi-objective DTLZ benchmarks, it can be seen that

TABLE 3. Spacing results of the proposed MOPSONN and four existing multi-objective PSO algorithms, SMPSO, dMOPSO, MMOPSO and CMOPSO on ZDT and DTLZ test suits. The best values are highlighted with bold fonts.

Problem		Obj.	SMPSO	dMOPSO	MMOPSO	CMOPSO	MOPSONN
ZDT1	Mean	2	-	1.66E-02+	9.17E-03+	1.03E-02+	**4.40E-03**
	Std.		-	9.36E-03	2.53E-03	6.16E-03	4.50E-04
ZDT2	Mean	2	-	-	-	-	**4.35E-03**
	Std.		-	-	-	-	3.76E-04
ZDT3	Mean	2	-	4.47E-02+	-	1.41E-02+	**5.23E-03**
	Std.		-	3.79E-02	-	1.80E-02	6.54E-04
ZDT4	Mean	2	-	-	-	-	**6.34E-03**
	Std.		-	-	-	-	1.50E-03
ZDT6	Mean	2	3.75E-01+	1.51E-01=	2.46E-01=	3.11E-01+	**1.45E-01**
	Std.		1.92E-01	1.51E-01	2.57E-01	2.25E-01	1.43E-01
DLTZ2	Mean	2	6.51E-03+	1.07E-02+	6.80E-03+	**3.42E-03-**	5.61E-03
	Std.		6.77E-04	2.66E-03	6.21E-04	3.20E-04	5.95E-04
DLTZ4	Mean	2	-	1.94E-01+	-	-	**5.73E-03**
	Std.		-	8.46E-02	-	-	6.26E-04
DLTZ5	Mean	2	6.55E-03+	1.21E-02+	6.83E-03+	**3.46E-03-**	5.23E-03
	Std.		6.63E-04	4.32E-03	7.94E-04	3.19E-04	7.09E-04
DLTZ6	Mean	2	3.64E-02+	1.04E-02+	8.98E-03+	**3.22E-03-**	4.48E-03
	Std.		5.93E-02	8.15E-03	6.01E-04	2.51E-04	5.68E-04
DLTZ7	Mean	2	1.27E-02+	2.33E-02+	6.71E-03+	**4.11E-03-**	4.85E-03
	Std.		1.23E-02	9.65E-03	2.86E-03	8.98E-04	5.47E-04
DLTZ2	Mean	3	5.83E-02+	6.69E-02+	6.14E-02+	**2.49E-02-**	2.97E-02
	Std.		4.86E-03	1.04E-02	5.31E-03	2.13E-03	3.60E-03
DLTZ4	Mean	3	6.62E-02=	1.22E-01+	6.10E-02+	2.46E-02+	**2.90E-02**
	Std.		7.46E-02	7.49E-02	5.80E-03	5.43E-03	2.73E-03
DLTZ5	Mean	3	8.17E-03+	1.47E-01+	1.01E-02+	5.12E-03+	**6.77E-03**
	Std.		7.27E-04	5.83E-02	7.45E-04	4.81E-04	8.50E-04
DLTZ6	Mean	3	1.88E-01+	9.89E-02+	1.22E-02+	1.16E-02+	**6.15E-03**
	Std.		1.45E-01	5.37E-02	1.21E-03	2.01E-02	6.84E-04
DLTZ7	Mean	3	6.17E-02+	1.64E-01+	7.07E-02+	4.48E-02+	**3.45E-02**
	Std.		1.87E-02	5.16E-02	2.26E-02	9.86E-03	3.74E-03
+/=/-			9/1/0	12/1/0	10/1/0	7/0/5	\

MOPSONN performs best on DTLZ4 and DTLZ7 while CMOPSO gets best on DTLZ2, DTLZ5 and DTLZ6. Figure 3 presents the nondominated solutions obtained by MOPSONN. For three-objective DTLZ test problems, we can see that MOPSONN achieves the best IGD values on DTLZ4 to DTLZ7 and CMOPSO obtains best IGD on DTLZ2. To visualize the performance of MOPSONN on the three-objective DTLZ test suit, the best results of MOPSONN on these benchmarks are plotted in Figure 4. In Table 3, the mean and standard deviation of spacing are presented where the sign '−' means that the algorithm fails to approximate an acceptable Pareto front at one or more time among 30 runs. It is clear that MOPSONN achieves the best performance in all bi-objective ZDT problems. On bi-objective DTLZ test problems, it is shown that CMOPSO has better spacing on DTLZ2, DTLZ5 to DTLZ7 and MOPSONN achieves best on DTLZ4. For three-objective problems, it is shown that CMOPSO obtains best spacing values on DTLZ2, ZDTLZ4 and DTLZ5, while on DTLZ6 and DTLZ7, our algorithm MOPSONN performs best. Although, CMOPSO achieves better spacing in four bi-objective DTLZ problem, but one can see that the spacing values of MOPSONN are still within an acceptable range.

It is important to point out that the existing multi-objective PSO algorithms fail in finding the Pareto front on ZDT2 and ZDT4 in some runs due to the low number evaluations, while MOPSONN is capable to fast converge towards the true Preto front and achieves great IGD and spacing values. From above empirical results, we can conclude that MOPSONN has a promising performance compared with exciting MOPSOs.

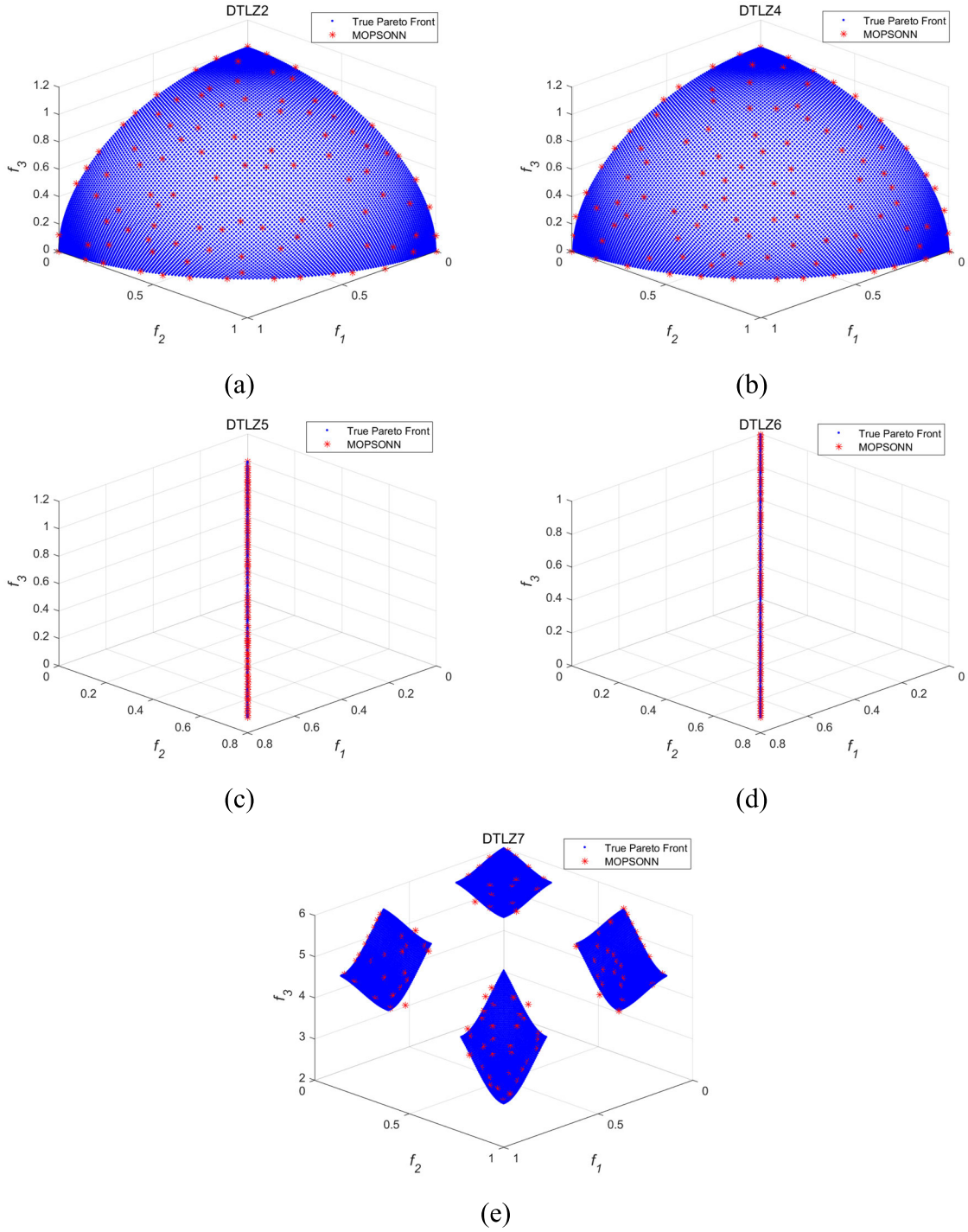

FIGURE 4. The Pareto front of three-objective DTLZ test suit obtained by MOPSONN. (a) DTLZ2; (b) DTLZ4; (c) DTLZ5; (d) DTLZ6; (e) DTLZ7.

Moreover, another crucial performance metric is the convergence speed. In the following, we compare the convergence speed of the proposed MOPSONN and the exciting PSO based multi-objective algorithms on two benchmark problems, namely ZDT1 and ZDT3. Figure 5 shows the convergence trajectory of the compared algorithms on ZDT1 and ZDT3 averaging 30 runs. The nondominated solution sets obtained by all algorithms are plotted in Figure 6. It can be seen that MOPSONN is capable to produce a well-spread set of solution along the pareto front after only 50 generation (i.e., 5,000 function evaluations), whereas the compared algorithms either still far from convergence or fails to cover the entire Pareto front.

TABLE 4. IGD results of the proposed MOPSONN and four MOEAs, NSGA-II, MOAED and NSGA-III, on ZDT and DTLZ test suits. The best values are highlighted with bold fonts.

Problem		Obj.	NSGAII	MOEAD	NSGAIII	MOPSONN
ZDT1	Mean	2	3.41E+00+	2.19E+00+	6.90E+00+	**4.35E-03**
	Std.		9.49E-01	1.40E+00	1.67E+00	4.40E-03
ZDT2	Mean	2	3.95E+00+	2.28E+00+	8.48E+00+	**4.27E-03**
	Std.		9.81E-01	9.70E-01	2.03E+00	1.53E-04
ZDT3	Mean	2	3.15E+00+	1.72E+00+	6.47E+00+	**4.84E-03**
	Std.		1.17E+00	1.30E+00	1.65E+00	2.46E-04
ZDT4	Mean	2	4.08E+01+	4.78E+01+	7.37E+01+	**4.08E-03**
	Std.		7.92E+00	1.03E+01	1.06E+01	2.55E-04
ZDT6	Mean	2	8.53E+00+	9.72E+00+	1.02E+01+	**2.42E-03**
	Std.		5.25E-01	1.49E+00	4.96E-01	1.35E-04
DLTZ2	Mean	2	5.09E-03=	4.33E-03-	**4.19E-03-**	5.12E-03
	Std.		1.57E-04	1.17E-04	8.88E-05	2.02E-04
DLTZ4	Mean	2	1.53E-01+	2.28E-01+	7.81E-02+	**5.21E-03**
	Std.		3.00E-01	3.43E-01	2.25E-01	2.33E-04
DLTZ5	Mean	2	5.07E-03-	4.31E-03-	**4.23E-03-**	5.34E-03
	Std.		1.49E-04	1.45E-04	1.28E-04	2.72E-04
DLTZ6	Mean	2	7.69E-02+	4.50E-01+	1.58E-01+	**4.39E-03**
	Std.		2.27E-01	5.52E-01	2.42E-01	8.50E-05
DLTZ7	Mean	2	6.58E-02+	4.25E-01+	2.00E-01+	**5.10E-03**
	Std.		1.86E-02	2.10E-01	5.67E-02	2.86E-04
DLTZ2	Mean	3	6.93E-02+	**5.49E-02-**	5.49E-02+	6.39E-02
	Std.		2.79E-03	2.91E-04	1.49E-04	2.66E-04
DLTZ4	Mean	3	6.73E-02+	4.28E-01+	1.82E-01+	**6.72E-02**
	Std.		1.85E-03	3.44E-01	2.44E-01	3.26E-03
DLTZ5	Mean	3	5.86E-03+	3.36E-02+	1.30E-02+	**5.09E-03**
	Std.		2.86E-04	2.46E-04	1.40E-03	2.65E-04
DLTZ6	Mean	3	3.68E-02+	1.21E-01+	3.72E-02+	**4.78E-03**
	Std.		1.49E-01	2.93E-01	7.23E-02	2.04E-04
DLTZ7	Mean	3	1.05E-01+	1.74E-01+	1.36E-01+	**5.68E-02**
	Std.		8.35E-02	1.19E-01	1.18E-01	1.35E-03
+/=/-			13/1/1	12/0/3	13/0/2	\

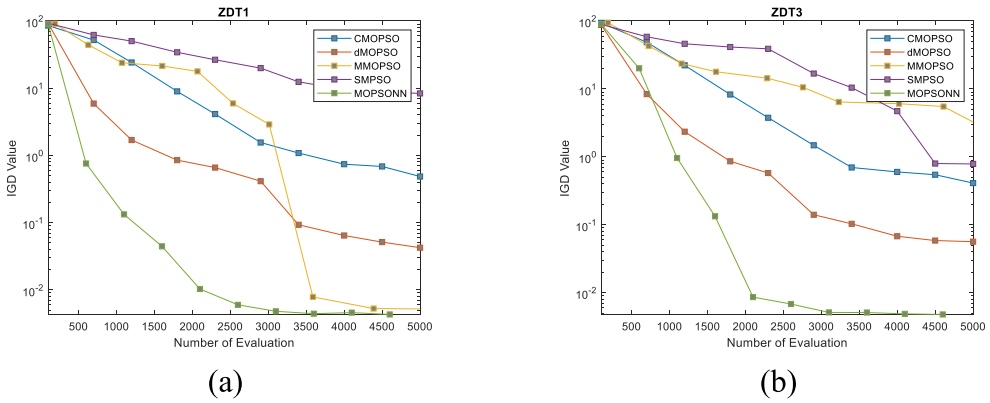

FIGURE 5. Convergence trajectory of MOPSONN and four existing MOPSOs, SMPSO, dMOPSO, MMOPSO and CMOPSO, on ZDT1 and ZDT3.

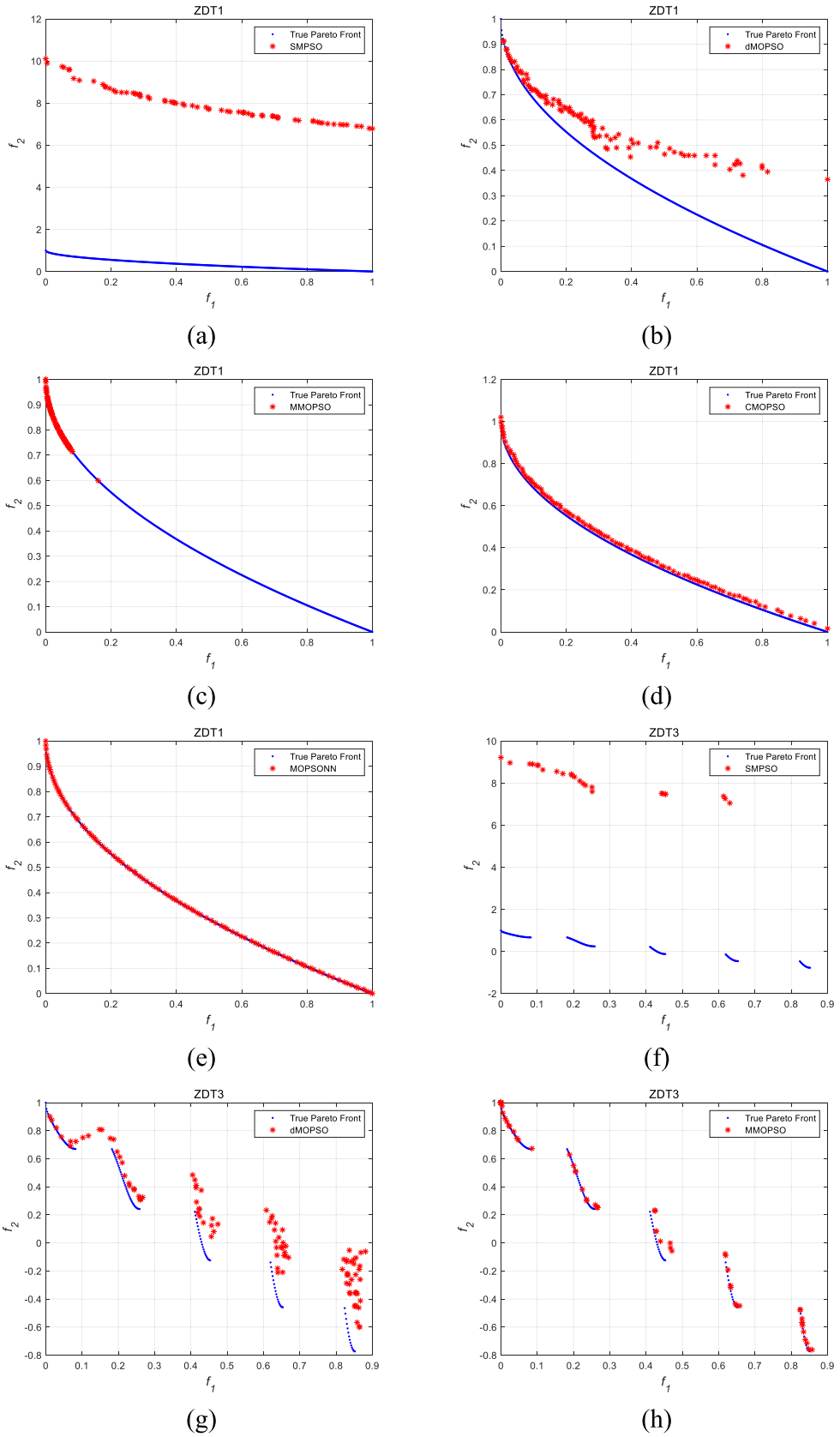

FIGURE 6. The nondominated solutions associated with the best performance of MOPSONN and four exciting PSO based multi-objective algorithms, averaging 30 runs. (a) SMPSO on ZDT1; (b) dMOPSO on ZDT1; (c) MMOPSO on ZDT1; (d) CMOPSO on ZDT1; (e) MOPSONN on ZDT1; (f) SMPSO on ZDT3; (g) dMOPSO on ZDT3; (h) MMOPSO on ZDT3; (i) CMOPSO on ZDT3; (j) MOPSONN on ZDT3.

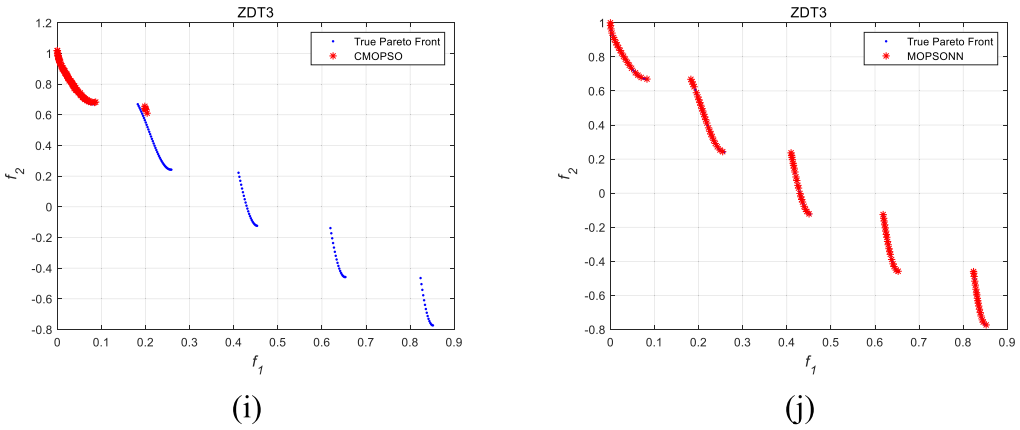

FIGURE 6. *(Continued.)* **The nondominated solutions associated with the best performance of MOPSONN and four exciting PSO based multi-objective algorithms, averaging 30 runs. (a) SMPSO on ZDT1; (b) dMOPSO on ZDT1; (c) MMOPSO on ZDT1; (d) CMOPSO on ZDT1; (e) MOPSONN on ZDT1; (f) SMPSO on ZDT3; (g) dMOPSO on ZDT3; (h) MMOPSO on ZDT3; (i) CMOPSO on ZDT3; (j) MOPSONN on ZDT3.**

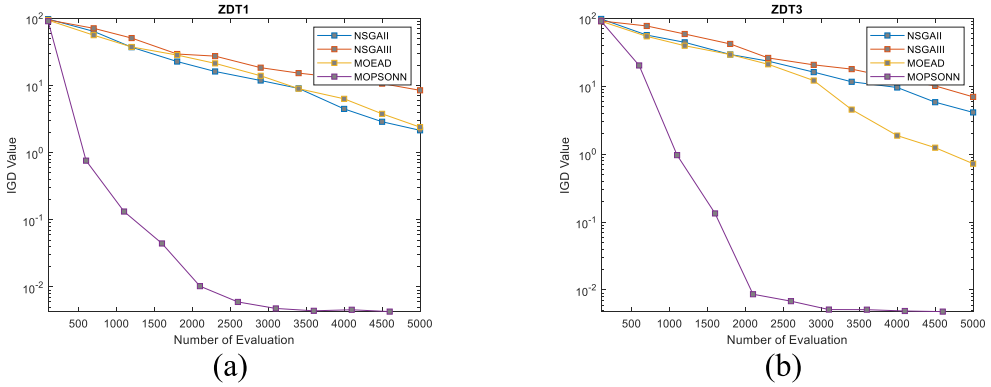

FIGURE 7. **Convergence trajectory of MOPSONN and three MOEAs, NSGAII, MOEAD and NSGAIII, on ZDT1 and ZDT3.**

E. COMPARISON WITH EXISTING MULTI-OBJECTIVE MOEAS

Table 4 presents the mean and standard deviation of IGD values for NSGA-II, MOAED, NSGA-III and the MOPSONN on bi-objective ZDT1 to ZDT4 and ZDT6, and two-/three-objective DTLZ2, DTLZ4 to DTLZ7 where the best mean for each benchmark is highlighted with bold fonts. It can be seen that MOPSONN produces the best approximation along the Pareto front on all ZDT test problems. On bi-objective DTLZ problems, we can see that MOPSONN performs best on DTLZ4, DTLZ6 and DTLZ7 and NSGA-III has better IGD values on DTLZ2 and DTLZ5. On three-objective DTLZ problems, NSGA-III achieves better IGD value on DTLZ2 while MOPSONN programs the best IGD performance on DTLZ4 to DTLZ7. Although NSGA-III performs better IGD values on three test functions, but the difference between IGD values obtained by NSGA-III and MOPSONN is not significant. It is evident that MOPSONN has also achieved promising overall IGD performance on the test instances in comparison with state-of-the-art MOAEs, where it has the best performance on 12 out of 15 benchmarks.

Table 5 shows the mean and standard deviation of spacing values for the compared algorithms. As evidenced by the results, nondominated solutions obtained by MOPSONN are spread throughout the Pareto fronts on all test problems and the obtained spacing values are the best among the compared algorithms except on DTLZ4. The overall Spacing results show that MOPSONN outperforms NSGA-II, MOEAD and NSGA-III on all benchmarks. Although, MOAED and NSGAIII outperform the IGD value of MOPSONN on three benchmarks, but MOPSONN has the overall IGD and spacing best performances compared with existing multi-objective PSO algorithms and state-of-the-art MOEA algorithms.

In the following, the convergence speed of the proposed MOPSONN is verified against the compared MOEAs, namely, NSGA-II, MOEA/D and NSGA-III. The convergence trajectory of these algorithms on ZDT1 and ZDT3 after 50 generations is shown in figure 7. It is evidence that MOPSONN can approximate the true Pareto front while the compared algorithms cannot converge towards the true Pareto front in such small number of function evaluations.

TABLE 5. Spacing results of the proposed MOPSONN and four MOEAs, NSGA-II, MOAED and NSGA-III, on ZDT and DTLZ test suits. The best values are highlighted with bold fonts.

Problem		Obj.	NSGAII	MOEAD	NSGAIII	MOPSONN
ZDT1	Mean	2	1.55E-01+	1.30E-01+	2.67E-01+	**4.40E-03**
	Std.		6.75E-02	1.40E-01	1.55E-01	4.50E-04
ZDT2	Mean	2	-	-	-	**4.35E-03**
	Std.		-	-	-	3.76E-04
ZDT3	Mean	2	1.54E-01+	1.44E-01+	2.55E-01+	**5.23E-03**
	Std.		9.57E-02	1.96E-01	1.40E-01	6.54E-04
ZDT4	Mean	2	-	3.37E-02+	-	**6.34E-03**
	Std.		-	1.67E-01	-	1.50E-03
ZDT6	Mean	2	**1.23E-01=**	-	1.38E-01=	1.45E-01
	Std.		1.05E-01	-	1.33E-01	1.43E-01
DLTZ2	Mean	2	6.89E-03+	5.83E-03+	6.91E-03+	**5.61E-03**
	Std.		7.68E-04	1.70E-04	3.35E-04	5.95E-04
DLTZ4	Mean	2	-	**4.80E-03-**	6.42E-03+	5.73E-03
	Std.		-	3.52E-03	2.20E-03	6.26E-04
DLTZ5	Mean	2	6.56E-03+	5.91E-03+	6.96E-03+	**5.23E-03**
	Std.		6.87E-04	2.03E-04	5.04E-04	7.09E-04
DLTZ6	Mean	2	3.57E-02+	1.54E-02+	6.00E-02+	**4.48E-03**
	Std.		5.00E-02	1.43E-02	8.71E-02	5.68E-04
DLTZ7	Mean	2	1.64E-02+	3.39E-02+	2.46E-02+	**4.85E-03**
	Std.		5.04E-03	3.56E-02	5.98E-03	5.47E-04
DLTZ2	Mean	3	5.86E-02+	5.58E-02+	5.72E-02+	**2.97E-02**
	Std.		5.61E-03	9.25E-04	1.34E-03	3.60E-03
DLTZ4	Mean	3	5.92E-02+	**2.76E-02-**	4.64E-02+	2.90E-02
	Std.		3.92E-03	2.37E-02	2.04E-02	2.73E-03
DLTZ5	Mean	3	9.53E-03+	1.31E-02+	1.48E-02+	**6.77E-03**
	Std.		7.51E-04	3.38E-04	3.01E-03	8.50E-04
DLTZ6	Mean	3	1.84E-02+	3.18E-02+	2.47E-02+	**6.15E-03**
	Std.		2.63E-02	6.89E-02	3.79E-02	6.84E-04
DLTZ7	Mean	3	6.77E-02+	1.83E-01+	6.71E-02+	**3.45E-02**
	Std.		1.31E-02	2.87E-02	1.17E-02	3.74E-03
+/=/-			11/1/0	11/0/2	12/1/0	\

The produced nondominated solution sets by all compared algorithms on ZDT1 and ZDT3 are plotted in figure 8. The comparison results show that MOPSONN has a promising convergence speed, which manifests that the pairwise competition and the proposed archive updating mechanism are capable to speed up the convergence speed and producing a high spread of nondominated solution along the Pareto front.

F. BENEFITS OF VICINITY DISTANCE METRIC

In the exploration phase of the proposed MOPSONN algorithm, we adopt the vicinity distance metric to measure the crowding of solutions in the archive such that the archive is updated by removing the most crowded solutions. In this subsection, we demonstrate the benefits of using this metric by comparing it with crowding distance which is adopted in most of multi-objective optimization algorithms. Figure 9 (a) shows the original solutions in the archive before truncation. Figure 9 (b) and (c) present the selected solutions to be kept in the archive by crowding distance and vicinity distance, respectively. Comparing Figure 9 (b) to (c) at the same solution scale, we can clearly see that the solution maintained by vicinity distance metric have a better spacing and diversity than those obtained by crowding distance metric. According to the analysis, we can conclude that truncating the archive using vicinity distance metric is beneficial to maintain a better diversity and spacing performance of solutions in the archive. Consequently, the elite particles set which is used to select global best particles to guide the swarm has a better diversity. Therefore, the diversity of the overall swarm is enhanced.

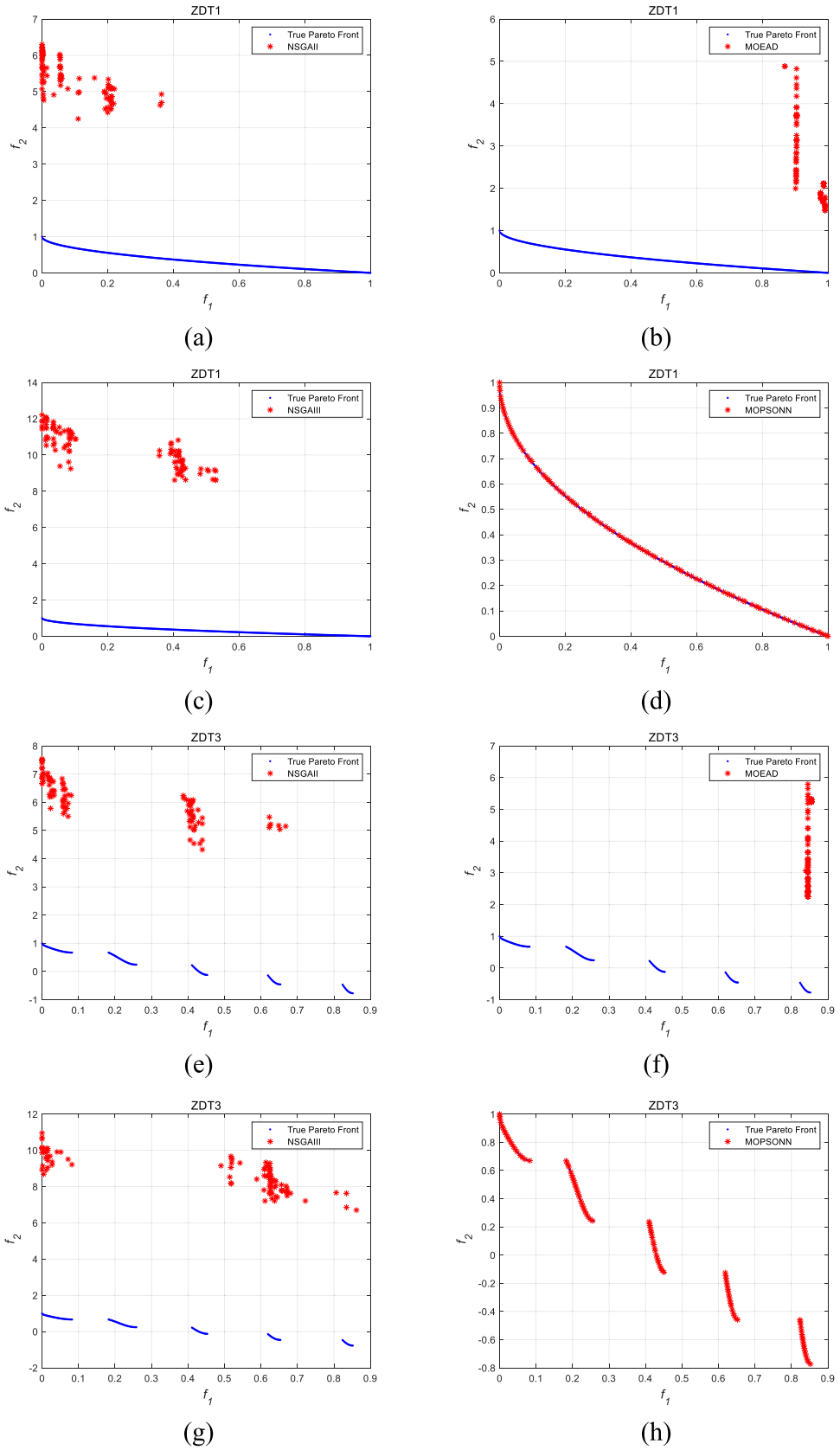

FIGURE 8. The nondominated solutions associated with the best performance of MOPSONN and three MOEAs, averaging 30 runs. (a) NSGAII on ZDT1; (b)MOEAD on ZDT1; (c) NSGAIII on ZDT1; (d)MOPSONN on ZDT1; (e) NSGAII on ZDT3; (f) MOEAD on ZDT3; (g) NSGAIII on ZDT3; (h)MOPSONN on ZDT3.

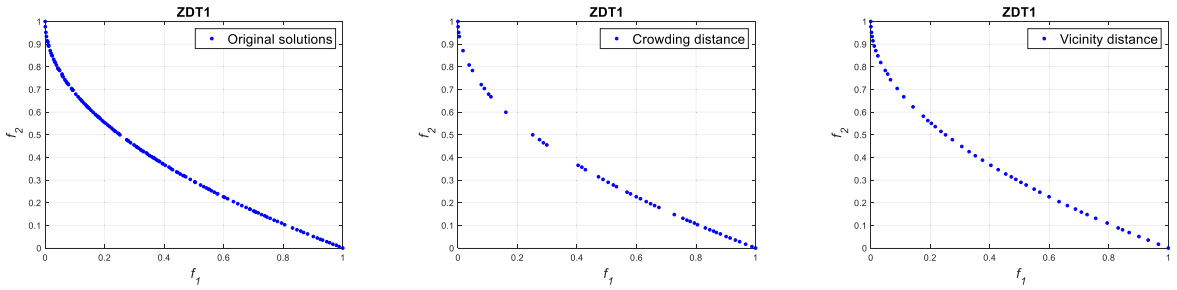

FIGURE 9. Comparison of the archive truncation using crowding distance and vicinity distance. (a) original solution; (b) selected solutions by crowding distance; (d) selected solutions by vicinity distance.

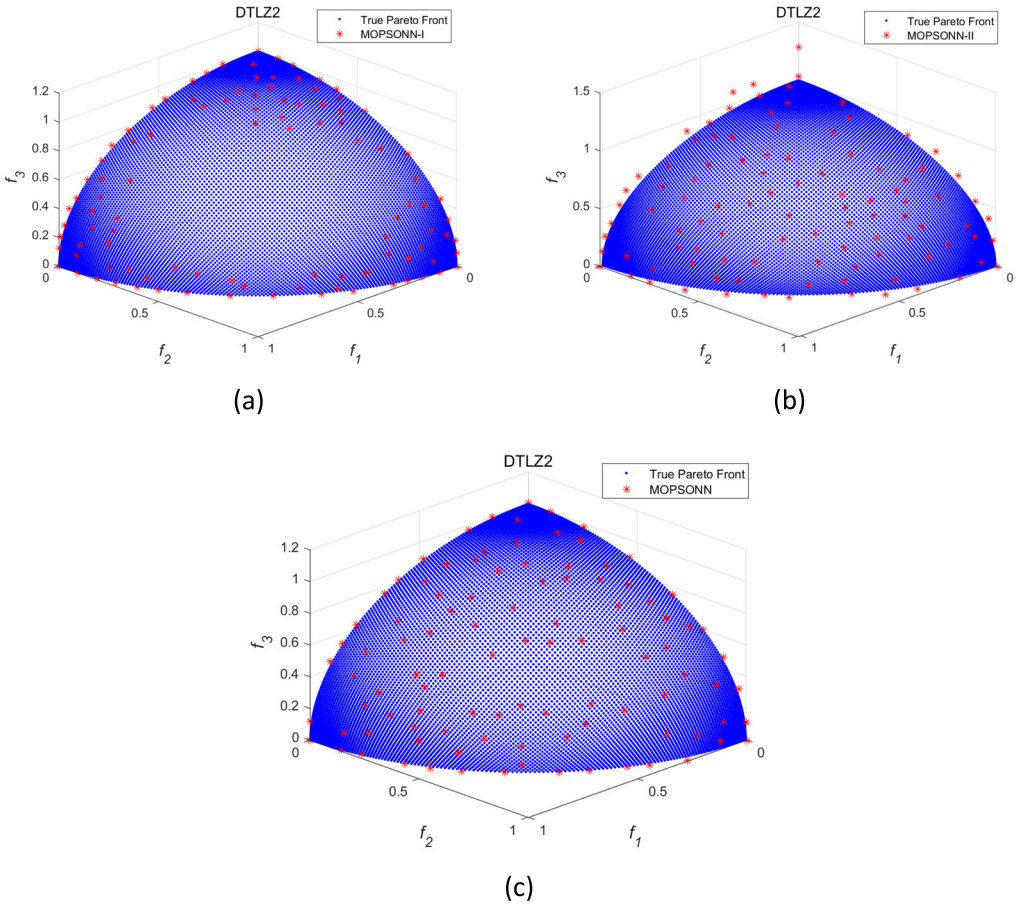

FIGURE 10. Comparison of nondominated solutions on DTLZ2. (a) MOPSONN-I; (b) MOPSONN-II; (c) MOPSONN.

G. BENEFITS OF THE PROPOSED ARCHIVE UPDATING APPROACH

In order to evaluate the benefits of the proposed archive updating approach in MOPSONN, a comparison is conducted with two variants of MOPSONN i.e. MOPSONN-I and MOPSONN-II, which both have the same component of MOPSONN, except that MOPSONN-I adopts only Max-cost rule along with Sum-of-cost rule to update the archive and MOPSONN-II adopts only vicinity distance metric. Table 5 presents the IGD values of the three compared algorithms (i.e. MOPSONN-I, MOPSONN-II and MOPSONN) on all adopted benchmark problems. It can be observed that MOPOSNN has the best IGD performance on 10 out of 15 test problems which validates the benefits of MOPSONN against MOPSONN-I and MOPSONN-II. It is noted that, in the test problems which the two variants outperform MOPSONN, the difference of the IGD values between MOPSONN and its two variants are not significant.

Figure 10 plots the obtained nondominated solution set of MOPSONN and its two variants on DTLZ2. Observed from Figure 10, the nondominated solutions obtained by MOPSONN-I are concentrated at the boundary of the true

TABLE 6. IGD results of MOPSONN-I, MOPSONN-II and MOPSONN, on ZDT and DTLZ test suits. The best values are highlighted with bold fonts.

Problem		Obj.	MOPSONN-I	MOPSONN-II	MOPSONN
ZDT1	Mean	2	4.56E-03	5.27E-03	**4.35E-03**
	Std.		3.42E-04	9.27E-04	4.40E-03
ZDT2	Mean	2	4.45E-03	**4.23E-03**	4.27E-03
	Std.		1.75E-04	1.04E-04	1.53E-04
ZDT3	Mean	2	2.29E-01	7.16E-02	**4.84E-03**
	Std.		2.80E-01	1.67E-01	2.46E-04
ZDT4	Mean	2	2.29E+00	5.07E-01	**4.08E-03**
	Std.		1.20E+00	6.00E-01	2.55E-04
ZDT6	Mean	2	2.83E-03	**2.28E-03**	2.42E-03
	Std.		5.82E-04	1.44E-04	1.35E-04
DLTZ2	Mean	2	**5.10E-03**	5.32E-03	5.12E-03
	Std.		1.28E-04	2.86E-04	2.02E-04
DLTZ4	Mean	2	5.52E-03	5.79E-03	**5.21E-03**
	Std.		4.91E-04	4.18E-04	2.33E-04
DLTZ5	Mean	2	5.09E-03	5.43E-03	**5.34E-03**
	Std.		2.19E-04	3.24E-04	2.72E-04
DLTZ6	Mean	2	1.02E-02	4.41E-03	**4.39E-03**
	Std.		7.47E-04	1.24E-04	8.50E-05
DLTZ7	Mean	2	5.38E-02	**5.00E-03**	5.10E-03
	Std.		1.78E-03	1.17E-04	2.86E-04
DLTZ2	Mean	3	8.10E-02	7.18E-02	**6.39E-02**
	Std.		1.06E-02	2.88E-03	2.66E-04
DLTZ4	Mean	3	1.58E-01	7.95E-02	**6.72E-02**
	Std.		8.83E-03	3.04E-03	3.26E-03
DLTZ5	Mean	3	1.83E-02	1.21E-02	**5.09E-03**
	Std.		3.07E-03	9.89E-04	2.65E-04
DLTZ6	Mean	3	1.06E-02	**4.37E-03**	4.78E-03
	Std.		1.59E-03	6.82E-05	2.04E-04
DLTZ7	Mean	3	9.47E-02	6.16E-02	**5.68E-02**
	Std.		9.87E-03	1.11E-03	1.35E-03

Pareto frons due to the aggressive usage of Max-cost rule and Sum-of-cost rule which may influence the diversity and spacing of obtained solutions. The nondominated set of solutions obtained by MOPSONN-II are still a bit far from the boundary of the true Pareto front. This demonstrates the fact that there is a need to adopted another strategy to enhance the convergence of solutions which is solved in MOPSONN by using the Max-cost rule and Sum-of-cost rule.

H. IMPACT OF PARAMETER SETTING

In the proposed MOPSONN, the setting of the parameter α is crucial due to its influence in the performance of MOPOSNN by controlling the balance between convergence and diversity of the swarm. As mentioned above, when the generation index exceeds the threshold α the swarm search changes from exploration phase into exploitation phase by changing the archive updating strategy. Before the generation index exceeds the threshold α the archive is updated according to vicinity distance metric described in 3.2.1. Adopting vicinity distance in this step is advantageous to preserve diversity of the archive and swarm by allowing all new nondominated solution to be added to the archive and then truncate solutions in the archive according to their crowding. When the generation index exceeds the threshold, the archive is updated using Max-cost rule along with Sum-of-cost rule as presented in 3.2.2 and 3.2.3. The Max-cost rule enhances the convergence speed by filtering the new nondominated solutions before adding them into the archive. The Sum-of-cost rule preserve diversity of solutions in the archive by comparing the total fitness value of the all objectives of the two most crowded solutions in the archive and remove the one with less total fitness value (in a minimization problem). Consequently, the diversity of the elite particles set in which the global best solution is selected is also enhanced. In order to carefully set the value of the threshold α such that we can achieve

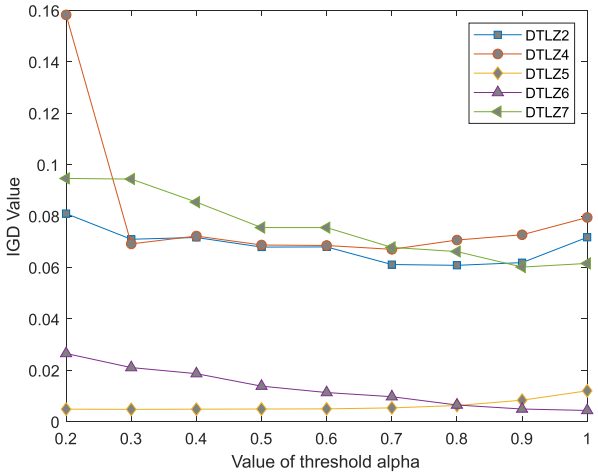

FIGURE 11. IGD values of the proposed MOPSONN with different setting of the threshold α on DTLZ2 and DTLZ4 to DTLZ7.

better balance of convergence and diversity we perform a sensitivity test analysis of this parameter. Figure 11 shows the IGD value of MOPSONN on three-objective DTLZ2 and DTLZ4 to DTLZ7, averaging 30 independent runs, where α varies from 0.2 to 1.

It can be seen that the performance of MOPSONN is sensitive to the value of α, where the IGD value of all test instances are high when α is small except for DTLZ5 where the IGD values increases when alpha increases. For DTLZ2 and DTLZ4, it is observed that the IGD values decrease when α increases to reach their minimum when α is 0.7 and then turns to increase by increasing α. For DTLZ5 and DTLZ7, we can see that the IGD values decrease when increasing α to reach their minimum at 1 and 0.9, respectively. For DTLZ6, the IGD value is behaving in a different way where its minimum and maximum are achieved at $\alpha = 0.2$, $\alpha = 1$, respectively. From the graph in Figure 11, we can observe that a moderate IGD value for all test instances can be achieved at $\alpha = 0.8$. Therefore, we suggest the value of alpha to be 0.8 in the proposed MOPSONN to solve MOPs.

V. CONCLUSION

In this paper, a novel multi-objective particle swarm optimization MOPSONN based on a new archive updating mechanism is presented. Two archive updating strategies are designed to enhance the performance of the algorithm. In early generations, the archive is updated according to vicinity distance metric which has proven its capability to enhance the diversity of solutions. In later generations, the archive is updated using two new rules, namely, Max-cost rule and Sum-of-cost rule, to accelerate the convergence speed. The performance of MOPSONN is verified by benchmark comparison with several state-of-the-art multi-objective algorithms. The IGD and Spacing values has evidenced how MOPSONN is a competitive algorithm where it outperforms the performance of the compared algorithms on most of the benchmarks. The capability of MOPSONN to fast converge towards the true Pareto front within a relatively small number of evaluations is also demonstrated.

The selection of the generation index which controls archive updating strategies is investigated by conducting a sensitivity test analysis on DTLZ test suit. The benefits of using vicinity distance metric against the most common crowding distance metric to measure the crowding of solutions is presented. The performance of MOPSONN is also validated by comparison with to variant of MOPSONN i.e. MOPSONN-I and MOPSONN-II. The experimental results demonstrate the promising performance of MOPSONN and justify the benefits of incorporating of two archive updating strategies.

REFERENCES

[1] L. Zhang, H. Pan, Y. Su, X. Zhang, and Y. Niu, "A mixed representation-based multiobjective evolutionary algorithm for overlapping community detection," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2703–2716, Sep. 2017.
[2] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, "A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 321–344, Jun. 2013.
[3] X. Zhang, F. Duan, L. Zhang, F. Cheng, Y. Jin, and K. Tang, "Pattern recommendation in task-oriented applications: A multi-objective perspective [application notes]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 3, pp. 43–53, Aug. 2017.
[4] Q. Lin and J. Chen, "A novel micro-population immune multiobjective optimization algorithm," *Comput. Oper. Res.*, vol. 40, no. 6, pp. 1590–1601, Jun. 2013.
[5] X. Chen, W. Du, and F. Qian, "Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization," *Chemometric Intell. Lab. Syst.*, vol. 136, pp. 85–96, Aug. 2014.
[6] D. Han, W. Du, W. Du, Y. Jin, and C. Wu, "An adaptive decomposition-based evolutionary algorithm for many-objective optimization," *Inf. Sci.*, vol. 491, pp. 204–222, Jul. 2019.
[7] T. Guan, F. Han, and H. Han, "A modified multi-objective particle swarm optimization based on levy flight and double-archive mechanism," *IEEE Access*, vol. 7, pp. 183444–183467, 2019.
[8] X. Chen, B. Xu, and W. Du, "An improved particle swarm optimization with biogeography-based learning strategy for economic dispatch problems," *Complexity*, vol. 2018, pp. 1–15, Jul. 2018.
[9] L. Li, W. Wang, and X. Xu, "Multi-objective particle swarm optimization based on global margin ranking," *Inf. Sci.*, vol. 375, pp. 30–47, Jan. 2017.
[10] J. Liang, Q. Guo, C. Yue, B. Qu, and K. Yu, "A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi-objective problems," in *Proc. Int. Conf. Swarm Intell.*, 2018, pp. 550–560.
[11] S. Cheng, H. Zhan, and Z. Shu, "An innovative hybrid multi-objective particle swarm optimization with or without constraints handling," *Appl. Soft Comput.*, vol. 47, pp. 370–388, Oct. 2016.
[12] Y.-X. Su and R. Chi, "Multi-objective particle swarm-differential evolution algorithm," *Neural Comput. Appl.*, vol. 28, no. 2, pp. 407–418, Feb. 2017.
[13] R. J. Kuo, M. Gosumolo, and F. E. Zulvia, "Multi-objective particle swarm optimization algorithm using adaptive archive grid for numerical association rule mining," *Neural Comput. Appl.*, vol. 31, no. 8, pp. 3559–3572, Aug. 2019.
[14] V. Trivedi, P. Varshney, and M. Ramteke, "A simplified multi-objective particle swarm optimization algorithm," *Swarm Intell.*, vol. 14, pp. 83–116, Jul. 2019.
[15] J. Meza, H. Espitia, C. Montenegro, E. Giménez, and R. González-Crespo, "MOVPSO: Vortex multi-objective particle swarm optimization," *Appl. Soft Comput.*, vol. 52, pp. 1042–1057, Mar. 2017.
[16] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, "A competitive mechanism based multi-objective particle swarm optimizer with fast convergence," *Inf. Sci.*, vol. 427, pp. 63–76, Feb. 2018.
[17] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, "Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients," *Inf. Sci.*, vol. 177, no. 22, pp. 5033–5049, Nov. 2007.

[18] H. Han, W. Lu, and J. Qiao, "An adaptive multiobjective particle swarm optimization based on multiple adaptive methods," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2754–2767, Sep. 2017.
[19] W.-F. Leong and G. G. Yen, "PSO-based multiobjective optimization with dynamic population size and adaptive local archives," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 38, no. 5, pp. 1270–1293, Oct. 2008.
[20] G. G. Yen and W. F. Leong, "Dynamic multiple swarms in multiobjective particle swarm optimization," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 39, no. 4, pp. 890–911, Jul. 2009.
[21] Z.-H. Zhan, J. Li, J. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.
[22] G. Xu, Y.-Q. Yang, B.-B. Liu, Y.-H. Xu, and A.-J. Wu, "An efficient hybrid multi-objective particle swarm optimization with a multi-objective dichotomy line search," *J. Comput. Appl. Math.*, vol. 280, pp. 310–326, May 2015.
[23] J. Luo, Y. Qi, J. Xie, and X. Zhang, "A hybrid multi-objective PSO-EDA algorithm for reservoir flood control operation," *Appl. Soft Comput.*, vol. 34, pp. 526–538, Sep. 2015.
[24] T. Cheng, M. Chen, P. J. Fleming, Z. Yang, and S. Gan, "A novel hybrid teaching learning based multi-objective particle swarm optimization," *Neurocomputing*, vol. 222, pp. 11–25, Jan. 2017.
[25] Y. Wang and Y. Yang, "Particle swarm optimization with preference order ranking for multi-objective optimization," *Inf. Sci.*, vol. 179, no. 12, pp. 1944–1959, May 2009.
[26] M. R. Sierra and C. A. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance," in *Proc. Int. Conf. Evol. Multi-Criterion Optim.*, 2005, pp. 505–519.
[27] C. Dai, Y. Wang, and M. Ye, "A new multi-objective particle swarm optimization algorithm based on decomposition," *Inf. Sci.*, vol. 325, pp. 541–557, Dec. 2015.
[28] X. Yu, H. Wang, and H. Sun, "Decomposition-based multi-objective comprehensive learning particle swarm optimisation," *Int. J. Comput. Sci. Eng.*, vol. 18, no. 4, pp. 349–360, 2019.
[29] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 191–204, Feb. 2015.
[30] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, "SMPSO: A new PSO-based Metaheuristic for multi-objective optimization," in *Proc. IEEE Symp. Comput. Intell. Milti-Criteria Decis.-Making*, Mar. 2009, pp. 66–73.
[31] S. Zapotecas Martínez and C. A. Coello Coello, "A multi-objective particle swarm optimizer based on decomposition," in *Proc. 13th Annu. Conf. Genetic Evol. Comput. (GECCO)*, 2011, pp. 69–76.
[32] Q. Lin, J. Li, Z. Du, J. Chen, and Z. Ming, "A novel multi-objective particle swarm optimization with multiple search strategies," *Eur. J. Oper. Res.*, vol. 247, no. 3, pp. 732–744, Dec. 2015.
[33] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[34] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[35] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.
[36] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw. (ICNN)*, 1995, pp. 1942–1948.
[37] C. A. C. Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. Congr. Evol. Comput. (CEC)*, May 2002, pp. 1051–1056.
[38] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evol. Comput.*, vol. 8, no. 2, pp. 149–172, Jun. 2000.
[39] W. Peng and Q. Zhang, "A decomposition-based multi-objective particle swarm optimization algorithm for continuous optimization problems," in *Proc. IEEE Int. Conf. Granular Comput.*, Aug. 2008, pp. 534–537.
[40] N. Al Moubayed, A. Petrovski, and J. McCall, "A novel smart multi-objective particle swarm optimisation using decomposition," in *Proc. Int. Conf. Parallel Problem Solving Nature*, 2010, pp. 1–10.
[41] N. Al Moubayed, A. Petrovski, and J. McCall, "D2MOPSO: MOPSO based on decomposition and dominance with archiving using crowding distance in objective and solution spaces," *Evol. Comput.*, vol. 22, no. 1, pp. 47–77, Mar. 2014.
[42] S. Kukkonen and K. Deb, "A fast and effective method for pruning of non-dominated solutions in many-objective problems," in *Parallel Problem Solving From Nature—PPSN IX*. Cham, Switzerland: Springer, 2006, pp. 553–562.
[43] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.
[44] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Proc. Congr. Evol. Comput. (CEC)*, May 2002, pp. 825–830.
[45] J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," M.S. thesis, Dept. Aeronaut. Astronaut., Massachusetts Inst. Technol., Cambridge, MA, USA, 1995.
[46] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.