A Fuzzy Decomposition-Based Multi/Many-Objective Evolutionary Algorithm

Liu, Songbai; Lin, Qiuzhen; Tan, Kay Chen; Gong, Maoguo; Coello Coello, Carlos A.

Published in:
IEEE Transactions on Cybernetics

Published: 01/05/2022

Document Version:
Post-print, also known as Accepted Author Manuscript, Peer-reviewed or Author Final version

License:
Other

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.1109/TCYB.2020.3008697

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.
A Fuzzy Decomposition-Based Multi/Many-Objective Evolutionary Algorithm

Songbai Liu, Qiuzhen Lin, Member, IEEE, Kay Chen Tan, Fellow, IEEE, Maoguo Gong, Senior Member, IEEE, and Carlos A. Coello Coello, Fellow, IEEE

Abstract—Performance of multi/many-objective evolutionary algorithms (MOEAs) based on decomposition is highly impacted by the Pareto front (PF) shapes of multi/many-objective optimization problems (MOPs), as their adopted weight vectors may not properly fit the PF shapes. To avoid this mismatch, some MOEAs treat solutions as weight vectors to guide the evolutionary search, which can adapt to the target MOP’s PF automatically. However, their performance is still affected by the similarity metric used to select weight vectors. To address this issue, this article proposes a fuzzy decomposition-based MOEA. First, a fuzzy prediction is designed to estimate the population’s shape, which helps to exactly reflect the similarities of solutions. Then,  𝑁  least similar solutions are extracted as weight vectors to obtain  𝑁  constrained fuzzy subproblems (  𝑁  is the population size), and accordingly, a shared weight vector is calculated for all subproblems to provide a stable search direction. Finally, the corner solution for each of  𝑁  least similar subproblems (  𝑁  is the objective number) is preserved to maintain diversity, while one solution having the best aggregated value on the shared weight vector is selected for each of the remaining subproblems to speed up convergence. When compared to several competitive MOEAs in solving a variety of test MOPs, the proposed algorithm shows some advantages at fitting their different PF shapes.

Index Terms—Evolutionary algorithm, fuzzy decomposition, multi/many-objective optimization.

I. INTRODUCTION

Evolutionary algorithms characterized by a population-based iterative search engine have been recognized as an effective approach for solving multi/many-objective optimization problems (MOPs), which are found in some real-life applications [1]–[3]. In this article, the unconstrained MOPs are considered, as modeled by

Minimize

\[
F(x) = (f_1(x), \ldots, f_m(x))
\]

Subject to: \( x \in \Omega \) (1)

where \( x = (x_1, \ldots, x_n) \) is an \( n \)-dimensional vector in the decision space \( \Omega \), and \( F(x) \) includes \( m \) (often conflicting) objectives to be optimized. The term multiobjective optimization problem is used when \( m = 2 \) or 3 and the many-objective optimization problem is used when \( m > 3 \). When considering all objectives, a set of equally optimal solutions called the Pareto set is often found, and the mapping of this set into the objective function space is called the Pareto front (PF) [8].

The main goal of solving MOPs is to find a set of solutions that can closely and evenly approximate their PFs [4]–[7].

In recent years, numerous multi/many-objective evolutionary algorithms (MOEAs) have been proposed to solve various MOPs, which can be classified into three main categories, that is: 1) the Pareto-based MOEAs [9], [10]; 2) indicator-based MOEAs [11]–[13]; and 3) decomposition-based MOEAs (MOEADs) [14], [15]. MOEAs have become very popular in recent years, mainly due to their promising performance and implementation efficiency [16]–[18]. With the decomposition approach, the target MOP is transformed into a set of subproblems, which are optimized in a collaborative way using an evolutionary search. However, the performance of MOEADs will be highly affected by their adopted weight vectors or aggregation methods. Some relevant works on this topic are briefly reviewed as follows.

For the weight vectors used in MOEADs, they are first initialized to have an even distribution in objective space, which helps to maintain the population’s diversity. In some early MOEADs [14]–[16], their weight vectors are uniformly sampled from the unit hyperplane \( f_1 + f_2 + \cdots + f_m = 1 \) by using a systematic extraction method, such as Das and Dennis’s method [19], Deb and Jain’s method [20], or mixture uniform design [21]. However, the performance of MOEADs strongly

Manuscript received February 3, 2020; revised April 14, 2020; accepted June 25, 2020. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61876110 and Grant 61836005; in part by the Joint Funds of the NSFC under Key Program Grant U1713212; in part by NSFC under Grant 61876162; in part by the Shenzhen Scientific Research and Development Fund and Development Fund Program under Grant JCZSY2018030712367294 and Grant JCZ2019080816421203. The work of V. Iros A. Coello Coello was supported in part by the CONACyT Project 1920 (Fronteras de la Ciencia) and in part by the SEP-Cinvestav 2018 Project (application no. 4). This article was recommended by Associate Editor H. Ishibuchi. (Corresponding author: Qiuzhen Lin.)

Songbai Liu is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, and also with the Department of Computer Science, City University of Hong Kong, Hong Kong.

Qiuzhen Lin is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: qiuzhlin@szu.edu.cn).

Kay Chen Tan is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the City University of Hong Kong Shenzhen Research Institute, Shenzhen, China (e-mail: kctan@cityu.edu.hk).

Maoguo Gong is with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, International Research Center for Intelligent Perception and Computation, Xidian University, Xi’an 710071, China.

Carlos A. Coello Coello is with the Department of Computer Science, CINVESTAV-IPN (Evolutionary Computation Group), Mexico City 07730, Mexico.

This article has supplementary downloadable material available at https://www.researchgate.net/profile/Liu_Songbai3, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2020.3008697

depends on the PF shapes, when the weight vectors with even distribution are used [22]. These MOEAs are good at solving MOPs with regular PFs but perform poorly on MOPs, whose PF shapes are degenerated, disconnected, inverted, or strongly convex/concave [28]. This is mainly due to the low matching degree of the evenly distributed weight vectors and the irregular PF shapes. To avoid this mismatching, three main kinds of improved approaches are used in some recent MOEAs, such as: 1) adaptive extraction of weight vectors [23]–[25]; 2) dynamic adjustment of weight vectors [26]–[32]; and 3) the use of multiple sets of weight vectors [33], [34], adapting the distribution of weight vectors to fit various PF shapes.

Regarding the aggregation methods, choosing an effective aggregation function to formulate scalar subproblems can help to balance convergence and diversity during the evolutionary search. Primitively, three aggregation functions, that is, weighted sum (WS), Tchebycheff (TCH), and penalty-based boundary intersection (PBI), are introduced in [14], which are also used in some recent MOEA/Ds [35], [36]. However, as discussed in [37] and [38], the search capabilities of these MOEAs are also highly impacted by their aggregation functions. For example, as shown in Fig. 1, the improvement region (IR) and the shared IR (SIR) for a subproblem decomposed from WS are always larger than that from TCH in the attainable objective space. Note that the detailed definitions of IR and SIR can be found in [40] and [41], where a larger size of IR implies a larger probability to find a solution associated to the subproblem with a better aggregated value, while a larger size of SIR indicates a larger probability to have the same better solutions for the subproblem and its neighbors (here, the neighbors of a subproblem are defined based on the distances between their relevant weight vectors). Thus, as observed in Fig. 1, the search capabilities of MOEAs using WS are generally better to speed up convergence and worse to maintain diversity when compared to that using TCH [40], [41], especially on MOPs with a large number of objectives [39]. In order to better balance convergence and diversity with aggregation functions in MOEAs, three main kinds of improved methods are proposed, such as embedding constraints into aggregation functions [42]–[46]; adaptive selection of aggregation functions [37], [47]; and new aggregation functions [48].

Although the above efforts have been conducted on weight vectors or aggregation methods to further enhance MOEAs, it is not always an easy task to select a suitable aggregation method adhering with a specific set of weight vectors, which is expected to conform the target MOP’s PF and characteristic for obtaining superior performance [41]. Recently, without using the preset weight vectors, solutions in some MOEAs [49]–[58] are treated as weight vectors, trying to guide the population to approximate the PF adaptively. Two metrics are often used to select these solutions as weight vectors, which can reflect their direction similarity. One is the direction angle of two solutions in the objective space, which is used in VaEA [49], MOEA/D-AM2M [51], MaOEACSS [50], PAEA [52], hpEA [53], MaOEAC [54], SPfAT [55], and PaRP/EA [56]. The other is the distance of two solutions’ projections on the unit hyperplane $f_1 + f_2 + \cdots + f_m = 1$ (m is the objective number), which is adopted in DDEA [57] and MaOEADDC [58]. A smaller value of the direction angle or the projections’ distances on the unit hyperplane means a higher direction similarity for two solutions, which is also used to define the neighborhoods of subproblems [14]. In Fig. 2, a simple example is plotted to show the measurement of direction similarity based on the direction angle (denoted by $\theta$) or the projections’ distances (denoted by $\text{dis}^3$, and $\text{dis}^2$) on three different unit hyperplanes $UH^p$ with $p = 0.5$, $1$, and $2$ (here, $UH^p$ indicates the unit hyperplane $f_1^p + f_2^p + \cdots + f_m^p = 1$, where $p$ is a positive parameter to determine the curvature of the hyperplane). Then, in these MOEAs [49]–[56], solutions with high direction similarity and poor convergence are removed, while those with low direction similarity are added into the population to compensate their diversity. Thus, the search directions are automatically guided by solutions, which have superior performance on solving various MOPs [49]–[58]. It is worth noting that the direction angle $\theta$ in Fig. 2 is actually equal to another special metric $\text{dis}^2$, as they will return the same comparison results on the direction similarity. Thus, the most commonly used metrics $\theta$ and $\text{dis}^2$ in these MOEAs [49]–[56] are actually two special cases in the family metrics $\text{dis}^p$ that represent the projections’ distance on $UH^p$ with $p = 1$ and $p = 2$. Obviously, as illustrated in Fig. 2, it is insufficient to use the above two metrics to measure the direction similarity of solutions [54], but an adaptive metric with variable $\text{dis}^p$ should be adopted when tackling MOPs with different PF shapes.

Given the above discussions, a fuzzy decomposition-based MOEA (called FDEA) is proposed in this article for solving MOPs with various PF shapes. In FDEA, a fuzzy decomposition is proposed to divide the target MOP as $N$ constrained fuzzy subproblems ($N$ is the population size), which includes two main components, that is, a fuzzy prediction and a weight...
vector extraction. Here, the fuzzy prediction is used to roughly predict a suitable $p$ for UH$^p$ that fuzzily fits the current non-dominated solution set. Then, the direction similarity between solutions can be appropriately measured based on their projections on UH$^p$. After that, the weight vector extraction is used to select $N$ solutions as weight vectors based on the direction similarity, which can keep a high matching degree with the target MOP’s PF. Furthermore, one shared weight vector can be obtained from the $N$ extracted weight vectors, which provides a stable search direction for all fuzzy subproblems. Finally, an elite selection strategy is run, which chooses one corner solution for each of the $m$ least similar subproblems ($m$ is the number of objectives) to maintain diversity and then selects one solution with the best aggregated fitness value for each of other subproblems to ensure convergence. When compared to some competitive MOEAs, our algorithm has shown some advantages in solving numerous test MOPs with various PF shapes.

The remainder of this article is organized as follows. Section II introduces our motivations to design FDEA. Section III presents the details of FDEA. The experimental results and some discussions are provided in Section IV. Finally, our conclusions and future work are presented in Section V.

II. MOTIVATIONS

In early studies of MOEADs [35], [36], their weight vectors are evenly sampled from the unit hyperplane, resulting in the fact that their performance is sensitive to the matching degree of weight vectors and the PF shapes [22]. To alleviate this problem, solutions are selected as weight vectors based on the direction similarity in some MOEAs [49]–[52], [54], [56], trying to automatically guide the evolutionary search toward different PFs. However, there still exist some open challenges for these MOEAs.

First, there are two special metrics for calculating direction similarity in these MOEAs, that is, $dis^1$ and $\theta$ (equal to $dis^2$) in Fig. 2, which actually represent the projections’ distances on UH$^p$ with $p = 1$ and $p = 2$. These two metrics cannot eliminate performance sensitivity when solving different MOPs with complicated PF shapes. For example, the angle $\theta$ between two solutions (equal to $dis^2$) is applied in VaEA [49], while $dis^1$ is implemented in DDEA [56]. When using VaEA and DDEA to solve the 3-objective MaF3 problem with a convex PF [59], their final solutions in a single run will be distributed mainly on the central area of the PF, while the PF boundaries are rarely covered, as illustrated in Fig. 3. This is reasonable since, as pointed out in MaOEA/C [54], the metric $\theta$ or $dis^2$ is not very appropriate to reflect the distribution of solutions near the boundaries of the convex PF, which can be observed in Fig. 4(a). Six solutions $x^1$–$x^6$ actually have a similar performance on convergence and diversity, but $\theta_1$ (the angle between $x^1$ and $x^2$) and $\theta_3$ (the angle between $x^5$ and $x^6$) are much smaller than $\theta_2$ (the angle between $x^3$ and $x^4$), which indicates that $x^3$ and $x^4$ show better distribution based on the metric $\theta$ when compared to other solutions near the boundaries. Similarly, if $dis^1$ is used for tackling convex MOPs, all solutions are projected onto the UH$^1$ to calculate their distances. However, this kind of unfair phenomenon still exists for the boundary solutions [see Fig. 4(a)]. Thus, an appropriate metric to exactly reflect the direction similarity between solutions is very important for improving the performance of these MOEAs. In PAEA [52] and PaRP/EA [56], the direction similarity is modified to compute the angles of two solutions on two adversarial directions, that is, one emanating from the ideal point and the other backward from the nadir point ($znad$), as plotted in Fig. 4(b). However, the angle oriented from $znad$ for two solutions is still a special case of their projections’ distances on a unit hypersurface.

Second, solutions are selected as weight vectors in some MOEAs [49]–[54], leading to the fact that a dynamic change of weight vectors occurs in each generation. As pointed out in [26]–[31], the frequent change of weight vectors may deteriorate the convergence speed, as the dynamically changed search directions in each generation may be confusing. To alleviate this problem, the weight vectors in MOEA-D-AM2M [51] are only changed once in each generation ($\tau$ is set to 50 in its experiments), while a fixed $m$-D weight vector $(1, 1, \ldots, 1)$ ($m$ is the objective number) is used with the WS method in VaEA [49], MaOEA/C [54], PAEA [52], and DAEA [57], which can provide a more stable search direction in the evolutionary process.

Based on the above discussions, in order to extract weight vectors that can be well adapted to various PF shapes, a natural idea is to fuzzily predict the target MOP’s PF shape during the evolutionary process, which can help to obtain an appropriate metric to exactly reflect the direction similarity. Then, weight vectors should be extracted to fit the PF shapes, and an elite selection strategy should be designed to balance convergence and diversity during the evolutionary process, under the case that the weight vectors are dynamically changed in each generation. Thus, to cover all the above-mentioned issues, FDEA is
Algorithm 1 FDEA(\(N, m, G_{\text{max}}\))
1: initialize \(P\) and set \(G = 1\)
2: while \(G \leq G_{\text{max}}\) do
3: generate the offspring population \(Q\)
4: \(U = P \cup Q\) and set \(P = \emptyset, Q = \emptyset\)
5: normalize all solutions in \(U\)
6: \((S_1, S_2, \ldots, S_N) = \text{Fuzzy-Decomposition} (U, N, m)\)
7: \(P = \text{Elite-Selection}(S_1, S_2, \ldots, S_N)\)
8: \(G++\)
9: end while
10: return \(P\)

proposed in this article with fuzzy decomposition and elite selection, which will be introduced in the following section.

III. PROPOSED ALGORITHM

The proposed FDEA algorithm includes two main components, that is, fuzzy decomposition and elite selection. First, after the offspring population is generated, fuzzy decomposition is executed, which includes a fuzzy prediction to estimate the population’s shape and a weight vector extraction to define \(N\) constrained fuzzy subproblems (\(N\) is the population size). After that, one shared weight vector is further obtained using the \(N\) extracted weight vectors, which can provide a stable search direction for all subproblems. Finally, an elite selection strategy is run to obtain the next population, which selects one corner solution for each of \(m\) least similar subproblems (\(m\) is the number of objectives) and one solution with the best aggregated value for each of the rest subproblems. In this way, our algorithm can properly balance convergence and diversity for solving MOPs with different PF shapes. To introduce FDEA, its main framework is first introduced in Section III-A, and then the details of fuzzy decomposition and elite selection are, respectively, given in Sections III-B and III-C to clarify the implementation of FDEA.

A. Main Framework

Here, to give an overview of FDEA, its main framework is provided in Algorithm 1 with three inputs: 1) \(N\) (the population size); 2) \(m\) (the number of objectives); and 3) \(G_{\text{max}}\) (the maximum number of generations). In line 1, an initial population \(P\) is randomly generated to have \(N\) solutions in decision space \(\Omega\) for solving the target MOP, and the generation counter \(G\) is initialized as 1. Then, an offspring population \(Q\) is produced to have \(N\) new solutions in line 3, by running variation operations (i.e., simulated binary crossover (SBX) [60] and polynomial-based mutation (PM) [61]) on \(N\) parent pairs randomly selected from \(P\). Then, these two populations (\(P\) and \(Q\)) are merged together in line 4 to obtain a union population \(U\) including \(2N\) solutions. Next, an adaptive normalization procedure used in NSGA-III [20] is run on \(U\) to mitigate the impact of different scaled objectives in MOPs, as shown in line 5. Thus, the \(i\)th objective \(f_i(x)\) of each solution \(x\) in \(U\) is normalized as

\[
f_i'(x) = \frac{f_i(x) - z_i^m}{z_i^{\text{rad}} - z_i^m}\]  

where \(f_i'(x) (i = 1, 2, \ldots, m)\) indicate the \(i\)th normalized objective of \(x\), while \(z^m = (z_1^m, z_2^m, \ldots, z_m^m)\) is an ideal point and \(z^{\text{rad}} = (z_1^{\text{rad}}, z_2^{\text{rad}}, \ldots, z_m^{\text{rad}})\) is a nadir point, which are obtained according to all the nondominated solutions in \(U\) using the estimation method in [20] and [56]. After that, in line 6, our fuzzy decomposition method is run to decompose the target MOP into \(N\) constrained fuzzy subproblems, which will be introduced in Section III-B. Then, the union population \(U\) will be divided to obtain \(N\) subsets \((S_1, S_2, \ldots, S_N)\), respectively, associated to \(N\) constrained fuzzy subproblems, by gathering solutions with high direction similarity in each subset. Finally, the elite selection process is run in line 7 to select one solution from each of subsets \((S_1, S_2, \ldots, S_N)\) to compose the new population, which will be introduced in Section III-C. Then, \(G\) is increased by 1 in line 8. While \(G\) is smaller than \(G_{\text{max}}\), the above procedures in lines 3–8 will be run iteratively; otherwise, \(P\) is reported as the final approximate solutions for the target MOP in line 10.

To further clarify the running of FDEA, the details of fuzzy decomposition and elite selection are introduced in the following sections.

B. Fuzzy Decomposition

In our fuzzy decomposition, a fuzzy prediction is first run to estimate the PF shape and then a weight vector extraction is executed to obtain \(N\) weight vectors, which define \(N\) constrained fuzzy subproblems. To clarify the running of fuzzy decomposition, its pseudocode is given in Algorithm 2 with three inputs: 1) \(U\) (the union population); 2) \(N\) (the population size); and 3) \(m\) (the objective number). In line 1, \(S_A\) (collecting all nondominated solutions), \(S_R\) (collecting all candidate solutions), and \(N\) subsets \(S_i\) (collecting all solutions associated to the \(i\)th subproblem, \(i = 1, 2, \ldots, N\)) are all initialized as empty sets. After running the nondominated sorting [9] on \(U\) in line 2, \(U\) is divided into \(L\) solution subsets \((F_1, F_2, \ldots, F_L)\) based on their nondominated ranks in \(U\). Then, set \(S_A\) as \(F_1\), and \(S_R\) is constructed in lines 3–5 by including one subset each time, staring from \(F_1\), then \(F_2\), and so on, until the size

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on May 16,2021 at 02:26:16 UTC from IEEE Xplore. Restrictions apply.
of $S_B$ exceeds $N$ for the first time. Let $F_l$ be the last subset included into $S_B$, that is, $S_B = F_1 + F_2 + \cdots + F_l$. The remaining fronts of $U$ (i.e., $F_{l+1}$ to $F_L$) are not considered in this process. After that, our proposed fuzzy prediction is run on $S_A$ in line 6, aiming to find an appropriate unit hypersurface (UHP) that fuzzily fits the shape of the current nondominated solutions. The details of fuzzy prediction will be introduced in Section III-B1.

After the estimation of the PF shape, each solution $x \in S_B$ will obtain a projection $x'$ on UHP in line 7, which is an intersection of a direction ray (from the origin pointing to $x$) on UHP. Then, the $i$th normalized objective of $x'$ can be obtained as follows:

$$f'_i(x') = \frac{f'_i(x)}{\left[\sum_{k=1}^{m} f'_k(x)\right]^{1/p}}$$

where $f'_i(x)(i = 1, 2, \ldots, m)$ is the $i$th normalized objective of $x$ in (2). Thus, the direction similarity of two solutions in $U$ can be defined as the distance of their projections on UHP. Then, a weight vector extraction introduced in Section III-B2 is run in line 8 to select $N$ solutions from $S_B$ as weight vectors (saved in $S_W$), while the remaining solutions in $S_B$ are preserved in $S_R$. Consequently, $N$ constrained fuzzy subproblems are defined in lines 9–11, by treating the $i$th solution $x' \in S_W$ as the first candidate solution for the $i$th subproblem and adding $x'$ into $S_I$ ($i = 1, 2, \ldots, N$). Afterward, each solution in $S_R$ will be associated to the subproblem with the highest direction similarity. Specifically, the distances of the $j$th solution $y^j (j = 1, 2, \ldots, |S_R|)$ in $S_R$ to the $i$th weight vector $x' (i = 1, 2, \ldots, N)$ in $S_W$ are calculated in line 14, the weight vector $u$ with the highest direction similarity for $y^j$ is found in line 16, and then $y^j$ is added into the subset $S_u$ to associate with its subproblem in line 17. Finally, $N$ solution subsets ($S_1, S_2, \ldots, S_N$), respectively, associated to $N$ subproblems are returned in line 19.

To further clarify the running of this fuzzy decomposition, the details of fuzzy prediction and weight vector extraction are, respectively, introduced as follows.

1) Fuzzy Prediction: In this process, an appropriate coefficient $p$ on UHP is estimated to effectively guide the evolutionary search and our fuzzy prediction is limited in the unit hypercube as defined by the origin and the $m$-dimensional point $(1, 1, \ldots, 1)$. The pseudocode of fuzzy prediction on UHP is given in Algorithm 3 with the input: $S_A$ with all nondominated solutions. In line 1, solutions outside the unit hypercube are first removed from $S_A$. Then, in line 2, the coefficient $p$ is initialized as 1, while the average value ($E$) and standard deviation ($\sigma$) of all distances from the solutions in $S_A$ to the linear unit hypersurface UH1 are initialized as 0. After that, in lines 3–6, the distance of each solution $x \in S_A$ to UH1 [marked as $Dis^1(x)$] can be computed as

$$Dis^1(x) = \frac{\sum_{i=1}^{m} f'_i(x) - 1}{\sqrt{m}}$$

where $f'_i(x)$ is the $i$th normalized objective of $x$ in (2) and $m$ is the objective number. Note that the distance $Dis^1(x)$ in (4) can be positive or negative. Specifically, the solution $x$ below UH1 will have $Dis^1(x) < 0$, otherwise will obtain $Dis^1(x) > 0$. Different cases of $Dis^1(x)$ are shown in Fig. 5, where $Dis^1(x') < 0$ and $Dis^1(x^2) > 0$. Then, in lines 6 and 7, the values of $E$ and $\sigma$ can be, respectively, calculated by (5) and (6), as follows:

$$E = \frac{\sum_{x \in S_A} Dis^1(x)}{|S_A|}$$

$$\sigma = \sqrt{\frac{\sum_{x \in S_A} (Dis^1(x) - E)^2}{|S_A| - 1}}$$

where $|S_A|$ denotes the cardinality of $S_A$. Based on $E$ and $\sigma$, $p$ can be roughly predicted in lines 8–11 by the following steps.

**Step 1:** Preset two sample sets ($sp_1$ and $sp_2$) of $p'$, as follows:

$$p' = \begin{cases} 1 - 0.05i, & p' \in sp_1 \text{ and } i = 0, 1, \ldots, T_1 - 1 \\ 1 + 0.1j, & p' \in sp_2 \text{ and } j = 0, 1, \ldots, T_2 - 1 \end{cases}$$

where $p'$ is a sample value of $p$, while $sp_1$ and $sp_2$ contain $T_1$ and $T_2$ samples of $p'$, respectively. Here, $T_1 = 17$ and $T_2 = 51$. 

![Fig. 5. Distance between solutions and UH1 in the unit hypercube, where $Dis^1(x') < 0$.](image)

![Fig. 6. Example of an MOP’s PF with a linear shape but far away from UH1.](image)

**Algorithm 3 Fuzzy Prediction ($SA$)**

1: remove solutions outside the unit hypercube from $S_A$
2: initialize $E = 0$, $\sigma = 0$, $p = 1$
3: for each $x \in S_A$
4: compute $Dis^1(x)$ with (4)
5: end for
6: compute $E$ with (5)
7: compute $\sigma$ with (6)
8: preset two samples sets $sp_1$ and $sp_2$ of $p$ with (7) //step 1
9: calculate $Fit(p_{\sigma})$ by (8) for each sample $p_{\sigma}$ //step 2
10: preliminarily predict $p$ by (9) //step 3
11: fuzzily adjust $p$ by (10) //step 4
12: return $p$
Algorithm 4 Weight Vector Extraction ($S_B$, $N$, $m$)

1: initialize $S_W = \emptyset$ and $S_R = \emptyset$
2: while $|S_B| > N$
3: get two most similar individuals $(x^d, x^e) \in S_B$ with (13)
4: compute $DS(x^d, S_B)$ and $DS(x^e, S_B)$ with (12)
5: if $DS(x^d, S_B) < DS(x^e, S_B)$
6: add $x^d$ into $S_R$ and remove $x^e$ form $S_B$
7: else
8: add $x^e$ into $S_R$ and remove $x^e$ form $S_B$
9: end if
10: end while
11: find two least similar individuals $(x^q, x^p) \in S_B$ with (14)
12: add $x^q$ and $x^p$ into $S_W$, and remove them from $S_B$
13: while $|S_W| < m$
14: get $x \in S_B$ : arg max $DS(x', S_W)$
15: add $x$ into $S_W$, and remove $x$ from $S_B$
16: end while
17: $S_W = S_W \cup S_B$
18: return $(S_W, S_R)$

Step 2: Calculate the average fitting value of $S_A$ and UH$^p$
for each $p'$ from $sp_1$ and $sp_2$, denoted as $Fit(p')$, as follows:

\[ Fit(p') = \frac{\sum_{i=1}^{S_A} \left( \sum_{j=1}^{m} \left| f_j(x') \right| \right)^{1/p'}}{|S_A|}. \]  

Step 3: Preliminarily, predict $p$ as a suitable $p'$ with the closest fitting between UH$^p$ and $S_A$ as follows:

\[
p = \begin{cases} 
    p' \in sp_1: \arg \min \left| \text{Fit}(p') - 1 \right| & \text{if } E > 0 \\
    p' \in sp_2: \arg \min \left| \text{Fit}(p') - 1 \right| & \text{otherwise}.
\end{cases}
\]  

Thus, if $E > 0$, then $p \geq 1$, UH$^p$ tends to be concave; $E < 0$ or $E \to 0$, respectively, indicate UH$^p$ tends to be convex or linear.

Step 4: Fuzzily adjust the value of $p$ based on the coefficient of variation ($cv$) between $E$ and $\sigma$ as follows:

\[
p = \begin{cases} 
    1.0 + cv, & \text{if } |cv| < 0.1 \land r_1 < 0.9 \\
    p + r_2, & \text{if } |cv| > 0.1 \land r_2 > 0.9 \\
    p, & \text{otherwise}
\end{cases}
\]  

where $cv = \sigma/E$, $r_1$ and $r_2$ are two random numbers in [0, 1], and $r_3$ is a random small disturbance in $[-0.02, 0.02]$. The fuzzy adjustment of $p$ here is mainly used to fit the MOPs with linear PF shapes that may be far away from UH$^1$, as shown in Fig. 6. Thus, if $cv$ approaches to 0, the value of $p$ is predicted close to 1 with a high probability.

As discussed in Section II, in some angle-based MOEAs [49]–[56] and decomposition-based MOEAs [35], [36], the PFs of MOPs are only approximated by UH$^p$ or UH$^1$ to define the distance of solutions. Moreover, the PF shapes of MOPs are estimated as the model of UH$^p$ in pa$k$-MyDE [73] and RIB-EMOA [74], where a new optimization problem is defined to find the most suitable value of $p$. However, the estimation methods need true PF information to obtain the hypervolume (HV) [63], which are unpractical and inefficient on computational cost. Furthermore, other complicated models are studied by using some machine-learning methods to exactly estimate the PF shapes of MOPs, for example, the generic front model in [75] and the growing neural gas network in [68]. However, their performance is significantly affected by the qualities of data for training and testing. Especially, in the early and median evolutionary stages, the solutions’ qualities are relatively poor to match the PF shape, which may mislead the used machine-learning methods. In a recently proposed MDEA [76], the PFs of MOPs are estimated by predicting a $p$ value from a sample set $[0.1, 0.2, \ldots, 4]$ to calculate the Minkowski distance. Here, $m$ extreme solutions and $m + 1$ center solutions are identified at first, and then the fitting degrees between each UH$^p$ sample and each pair of (center solution and extreme solution) are calculated in order to find the $p$ value for the most frequently matched PF shape. When compared to the above-mentioned methods, our approach is easily implementable and efficient in terms of computational cost, as it only needs to fuzzily fit the shape of the current nondominated solutions to the model of UH$^p$ within some sample values of $p$.

2) Weight Vector Extraction: After the fuzzy prediction of UH$^p$ is given in Algorithm 3, a number of solutions are selected from $S_B$ based on the direction similarity of their projections on UH$^p$, which are then used as weight vectors to assist the fuzzy decomposition for the target MOP. Here, the pseudocode of weight vector extraction is introduced in Algorithm 4 with the inputs: $S_B$ (all candidate solutions), $N$ (the population size), and $m$ (the objective number). Here, two definitions of calculating the direction similarity are given as follows.

Definition 1: Given two different solutions $x, y \in S_B$, the direction similarity of $x$ and $y$ [called $DS(x, y)$] is measured by the distance of their projections (i.e., $x'$ and $y'$) on UH$^p$, which can be computed as

\[
DS(x, y) = \sqrt{\sum_{i=1}^{m} \left( f'_i(x') - f'_i(y') \right)^2} 
\]  

where $f'_i(x')$ and $f'_i(y')$ can be obtained by (3). A smaller value of $DS(x, y)$ means a higher direction similarity of $x$ and $y$.

Definition 2: Given a solution $x \in S_B$ and a subset $S_b \subseteq S_B$, the direction similarity between $x$ and $S_b$ [called $DS(x, S_b)$] is measured by finding a solution $y \in S_B$ with the minimum value of $DS(x, y)$ in $S_B$. Thus, $DS(x, S_b)$ is set the same with $DS(x, y)$, which can be computed as follows:

\[
DS(x, S_b) = \min_{y \in S_B, x \neq y} DS(x, y).
\]  

A smaller value of $DS(x, S_b)$ means a higher direction similarity of $x$ and $S_b$. Note that (12) is also used in the recently
proposed PMEA [77], which adaptively replaces each invalid weight vector by finding the solution vector with the lowest direction similarity to the remaining unselected solutions.

In line 1 of Algorithm 4, $S_W$ (collecting the selected solutions as weight vectors) and $S_R$ (collecting the remaining solutions) are all initialized as empty sets. Then, solutions as well as weight vectors are collected into $S_R$ in lines 2–10, while the remaining solutions are collected into $S_W$ in lines 11–17. In lines 2–10, the solution with the highest direction similarity to $S_R$ is removed from $S_R$ and added into $S_R$ in lines 2–10, until the size of $S_R$ is equal to $N$. To be specific, two solutions $(x^i, x^j) \in S_R$ with the highest direction similarity are found in line 3, as follows:

$$
\left( x^i, x^j \right) = \arg \min_{(x^i, x^j) \in S_R, i \neq j} DS(x^i, x^j) \tag{13}
$$

where $(d, q) \in \{1, 2, \ldots, |S_B|\}$ and $DS(x^i, x^j)$ is computed by (11). Then, their direction similarity to $S_B$ is computed by (12) in line 4. One solution with the higher direction similarity to $S_B$ is removed from $S_B$ and added into $S_R$ in lines 5–8. After repeating the above process, $S_R$ is obtained and $S_B$ has $N$ selected solutions.

In lines 11–16, $m$ corner solutions [72] (i.e., the first $m$ members of $S_W$) are selected from $S_B$ to maintain the boundary information of population. First, two solutions $(x^i, x^j) \in S_B$ with the highest direction similarity in $S_B$ are found in line 11 as the first two corner solutions, as follows:

$$
\left( x^i, x^j \right) = \arg \max_{(x^i, x^j) \in S_B, i \neq j} DS(x^i, x^j) \tag{14}
$$

where $(i, j) \in \{1, 2, \ldots, |S_B|\}$ and $DS(x^i, x^j)$ is computed by (11). Then, $x^i$ and $x^j$ are added into $S_W$ and removed from $S_B$ in line 12. By this way, $S_W$ has two selected solutions. While the number of solutions in $S_W$ is smaller than $m$ (i.e., $|S_W| < m$) in line 13, one solution $x$ with the lowest direction similarity to $S_W$ will be found in line 14, which will be added into $S_W$ and removed from $S_B$ in line 15. Obviously, the iterative process of lines 14 and 15 will ultimately save $m$ corner solutions in $S_W$, and the remaining solutions in $S_B$ are also added into $S_W$ in line 17 in an ascending order. Finally, the final solution sets $S_W$ and $S_R$ are returned in line 18 to assist the fuzzy decomposition described in Algorithm 2.

C. Elite Selection

After the above fuzzy decomposition, $N$ subsets are obtained as $S_1, S_2, \ldots, S_N$, where solutions in each $S_i$ are associated to the $i$th subproblem ($i = 1, 2, \ldots, N$). Based on the output of Algorithm 2, the first member of $S_i$ is the weight vector for the $i$th subproblem, the first member in the first $m$ subsets is also a corner solution for its subproblem, and each solution in $S_i$ has a high direction similarity to the first member of $S_i$.

In this process, an elite selection method is run to obtain $N$ final solutions with balanceable convergence and diversity from $S_1, S_2, \ldots, S_N$, where one solution is selected from each subset. As shown in Section III-B, the population’s diversity has been well maintained by $N$ selected weight vectors, that is, the first solution in each $S_i$ ($i = 1, 2, \ldots, N$). Thus, this elite selection prefers to select one solution with good convergence from each $S_i$. In general, MOEADs will use an aggregated function (such as WS, TCH, or PBI) and the associated weight vectors to evaluate the solutions’ quality, aiming to balance convergence and diversity when a number of evenly distributed weight vectors are used during the evolutionary process. However, $N$ weight vectors extracted from Algorithm 4 in FDEA are dynamically changed in each generation, which will highly affect the search direction if the traditional decomposition methods in [14] and [15] are used.

Here, in order to provide a stable search direction for each subproblem, a shared weight vector $w$ and the WS aggregated function are used for all subproblems. The pseudocode of elite selection is given in Algorithm 5 with the inputs: $N$ subsets ($S_1, S_2, \ldots, S_N$). In line 1, the new population $P$ is initialized as an empty set and the shared weight vector $w = (w_1, w_2, \ldots, w_m)$ is initialized as $(1, 1, \ldots, 1)$. In lines 2–4, each member $x_k$ ($k = 1, 2, \ldots, m$) of $w$ is obtained by

$$
w_k = \sum_{l=1}^{N} x_k l \epsilon S_P(x_k) \tag{15}
$$

where $x_k$ represents the first member of $S_i$, $x_k'$ is the projection of $x_k$ on the predicted UHP, and $f_k'(x_k')$ can be computed by (3). Please note that the fixed weight vector $(1, 1, \ldots, 1)$ was used in some MOEAs [49]–[54] to guide the evolutionary search. However, an adaptive weight vector $w$ applied here is more suitable, as it can follow the distribution of $N$ extracted weight vectors, which can be more effective to guide the evolutionary search, especially for some imbalance cases in MOPs [27]. Thus, the $j$th constrained fuzzy subproblem can be mathematically defined as follows:

$$
\text{Minimize } g^{ws}(x|p^j) = w_1f'_1(x) + \cdots + w_mf'_m(x) \\
\text{subject to: } x \in S_j 
$$

where the weight vector $p^j$ is the first solution in $S_j$, $f_k'(x)$ ($k = 1, 2, \ldots, m$) is the $k$th normalized objective of $x$ in (2), the shared weight vector $w$ is obtained by (15), and $j = 1, 2, \ldots, N$. With the assistance of the fuzzily predicted UHP, the weight vector $p^j$ of the $j$th subproblem is obtained by Algorithm 4, which is stored, respectively, as the $j$th member of $S_W$ and the first member of $S_j$. Specifically, the weight vectors $p^1, p^2, \ldots, p^N$ are adopted to define the constrained attainable objective subspace of each subproblem, obtain the associated solution set $S_j$ in lines 9–18 of Algorithm 2, and obtain the shared $w$ in (15). Next, $m$ corner solutions are added into $P$ by collecting the first member of $S_i$ ($i = 1, 2, \ldots, m$) in lines 5–7, which maintains the boundary information of population. Then, the current best solution for each subproblem can be selected from its corresponding candidate solution set $S_j$ ($j = m + 1, m + 2, \ldots, N$) using (16). Finally, the new population $P$ having $N$ currently optimal solutions for these $N$ constrained fuzzy subproblems is returned in line 12 for the next generation.

D. Discussion

In the above sections, the general framework and main components of FDEA have been introduced in detail. Here,
Fig. 7. Simple example to show the process of FDEA (a) for the fuzzy prediction, (b)–(g) for the weight vector extraction, and (h)–(j) for the elite selection.

A simple example is plotted in Fig. 7, which shows the process of FDEA on solving a minimization problem with two objectives ($f_1$ and $f_2$). In Fig. 7(a), assume that the union population $U$ has three parents and three children ($x^1$–$x^6$) that are nondominated with each other. By using the fuzzy prediction in Algorithm 2, the predicted curve $U^p$ can be obtained in Fig. 7(a), and each solution correspondingly obtains a projection on $U^p$. Then, the direction similarity of any two different solutions is measured based on their projections’ distances using (3). Three weight vectors for decomposition in (16) are extracted from $U$ by iteratively dividing $U$ into two subpopulations $S_W$ and $S_R$, which are shown in Fig. 7(b)–(g). Concretely, the most similar solution pair ($x^1$, $x^2$) is identified in Fig. 7(a) using (13), followed by adding $x^4$ to $S_R$ as $DS(x^4, U)$ is larger than $DS(x^3, U)$, as shown in Fig. 7(b) and (c). By the way, the same way, the most similar solution pairs ($x^1$, $x^5$) in Fig. 7(b) and ($x^3$, $x^5$) in Fig. 7(d) are, respectively, found. Then, $x^2$ and $x^3$ are, respectively, added into $S_R$ in Fig. 7(e) and (g). Thereafter, two least similar solutions $x^1$ and $x^5$ in Fig. 7(f) are treated as two corner solutions using (14), which are also the first two solutions of $S_W$. Thus, three solutions are saved in $S_W$, that is, $x^1$, $x^5$, and $x^3$ in Fig. 7(f), which are treated as weight vectors to define three constrained attainable objective subspaces in Fig. 7(f) and to obtain a shared weight vector $w$ in Fig. 7(h). Furthermore, three constrained fuzzy subproblems formulated by (16) are associated to three candidate solution sets, respectively, that is, {$x^1$, $x^2$}, {$x^5$}, and {$x^3$, $x^4$, $x^5$} in Fig. 7(i). Finally, one best solution for each subproblem is selected from its candidate solution set by the elite selection in Algorithm 5. As shown in Fig. 7(j), the selected three optimal solutions are $x^1$, $x^5$, and $x^4$.

Obviously, the proposed FDEA is a decomposition-based MOEA, where $N$ weight vectors are extracted from the combined population to formulate $N$ constrained fuzzy subproblems ($N$ is the population size). Compared with the traditional decomposition-based MOEAs, FDEA is characterized by the following three features.

1) FDEA involves a fuzzy prediction procedure before the decomposition of an MOP, which aims to estimate the shape of the current nondominated solutions using the model $U^p$, as plotted in Fig. 7(a). Consequently, FDEA can handle MOPs with different curvatures in their PFs, for example, convexity, linearity, or concavity.

2) FDEA does not require a set of predefined weight vectors, but fuzzily extracts them from the population, which aims to decompose an MOP automatically, as plotted in Fig. 7(b)–(g). Thus, FDEA can handle MOPs with irregular shapes of PFs, for example, degenerated, inverted, and disconnected PFs.

3) FDEA decomposes an MOP into $N$ constrained fuzzy subproblems by (16), where each subproblem has a candidate solution set in its constrained attainable objective subspace [as plotted in Fig. 7(i)] and all subproblems share the same aggregation function [i.e., the WS function with the shared $w$ in (15)] in order to provide a stable evolutionary direction, as plotted in Fig. 7(h). Thus, by collaboratively optimizing the shared WS function and obtaining the optimal solution for each subproblem from its candidate solution set, FDEA can balance the population’s diversity and convergence well.

IV. EXPERIMENTAL STUDIES

A. Benchmark Problems and Performance Metrics

To investigate the effectiveness of our proposed FDEA, especially on problems with irregular PF shapes, a total of 30 test problems with different PF shapes are selected from the WFG [62], WFG4x [27], and MaF [59] test suites, including WFG1–WFG9, WFG41–WFG48, and MaF1–MaF13. In this study, the objective number $m$ is set from 2 to 15, that is, $m \in \{2, 3, 5, 8, 10, 15\}$. The number of decision variables $n$ in each problem is set as follows. For WFG1–WFG9 and WFG41–WFG48, the decision variables have $k$ position-related parameters and $l$ distance-related parameters, that is, $n = k + l$, where $k$ and $l$ are, respectively, set to $2 \times (m - 1)$ and 20 as suggested in [54]; for MaF1–MaF7, $n$ is set by $n = m + k - 1$, where $k$ is set to 10 for MaF1–MaF6 and set to 20 for MaF7 as suggested in [59]; $n = 2$ and $n = 5$ are, respectively, used for MaF8 and MaF9, and MaF13. Due to page limitations, the main characteristics of these test problems are summarized in Table A.1 of the supplementary material.
In this article, the well-known HV [63] and inverted generational distance (IGD) [64] are used as the performance indicators. Both HV and IGD are able to reflect the convergence and diversity of the final solution set produced by the algorithms. A larger HV value or a smaller IGD value indicates a better approximation to the true PF. When computing HV, a reference point dominated by the nadir point of the true PF is carefully specified for various problems. To ensure a fair comparison, the reference points of all test problems in our experiments are set as suggested in [65] and [66]. All objective values in the final solution set are first normalized by using $1.1 \times (f_{min}^{0} + f_{min}^{1} + \ldots + f_{min}^{m})$, where $f_{k}$ is the $k$th member of the nadir point in the true PF ($k=1,2,\ldots,m$), and then the reference point is set to the $m$-D point (1,0,1,0, ... ,1,0).

Here, the recently proposed walking fish group algorithm [67] is used to compute the exact HV values for problems with $m \leq 10$, and the Monte Carlo simulation [13] using $10^5$ sampling points is used to calculate HV for problems with $m = 15$. To calculate IGD, a large set of points that are evenly sampled from the true PF is required. In particular, 5000, 10000, and 20000 points are uniformly sampled from the true PF to calculate the IGD values of 2-objective, 3-objective, and 5-objective to 15-objective problems, respectively. Due to page limitations, refer to [13], [64], and [67] for details of computing HV and IGD.

B. Compared Algorithms and Parameters Settings

In this study, eight competitive MOEAs, that is, NSGA-III [20], $\theta$-DEA [69], VaEA [49], MaOEA/C [54], MOEA/D-LTD [70], PaRP/EA [56], MOEA/AD [33], and DDEA [57], are included for performance comparison. In NSGA-III and $\theta$-DEA, $N$ weight vectors evenly extracted on $\text{UH}^1$ are used in their optimization process, whereas in MOEA/D-LTD, the weight vectors and aggregation methods are adaptively set by learning the characteristics of the estimated PFs for various problems. Moreover, in VaEA, MaOEA/C, and DDEA, solutions are treated as weight vectors. The angles between solutions are used to reflect their direction similarity as done in VaEA and MaOEA/C, while the distances of solutions' projections on $\text{UH}^1$ are employed in DDEA. Furthermore, two adversarial directions are considered in MOEA/AD by using two sets of weight vectors, and the direction similarity of solutions is adaptively computed by using two adversarial directions in PaRP/EA. The parameter settings of these MOEAs are provided in Table A.II of the supplementary material, as suggested in their references. Particularly, FDEA and all compared MOEAs use the same evolutionary operators, that is, SBX and PM.

The settings of population size for different numbers of objectives are listed in Table A.III of the supplementary material. For test problems with 2, 3, 5, 8, 10, and 15 objectives, the numbers of weight vectors are, respectively, set to 100, 120, 210, 240, 275, and 240, using the two-layer generation method with the simplex-lattice design factor $H$ in [44]. According to [54], the population size in MaOEA/C should be set as a multiple of $m$. Thus, its population size on 10-objective test problems was set to 280. For other compared MOEAs, their population sizes are set the same as the number of weight vectors. All the compared MOEAs are run 30 times independently on each test problem. The mean HV and IGD values and their standard deviations (included in brackets after the mean HV or IGD results) from 30 runs are collected for comparison. All the compared MOEAs are terminated when a predefined maximum number of generations $G_{\text{max}}$ is reached. The values of $G_{\text{max}}$ are set to 300, 500, 600, 800, 1000, and 1500, respectively, for 2-, 3-, 5-, 8-, 10-, and 15-objective test problems. Their maximum number of function evaluations (MFE) can be easily obtained by computing $\text{MFE} = N \times G_{\text{max}}$.

C. Comparison Results on WFG and WFG4x Problems

In this experiment, FDEA was compared with respect to NSGA-III, $\theta$-DEA, VaEA, MaOEA/C, MOEA/D-LTD, PaRP/EA, and DDEA on WFG1–WFG3, WFG41–WFG48, and WFG9 with different numbers of objectives. Here, a Wilcoxon rank-sum test with a 0.05 significance level and a Wilcoxon signed ranks test from the tool KEEL [71] are used to ensure a statistically sound conclusion, which can show the statistically significant differences on the performance results. In the following tables, the symbols “+,” “-,” and “~” indicate that the comparison results of the corresponding algorithm are significantly better than, worse than, and similar to FDEA on tackling each problem with different objectives. “+/−/~” collects the corresponding numbers of the above statistical results, and “Avg. rank” indicates the average performance ranks of FDEA and other compared MOEAs by Friedman’s test from KEEL.

1) IGD Results on Problems With 2-Objective and 3-Objective: The average IGD results of FDEA and its seven competitors on WFG1–WFG3, WFG41–WFG48, and WFG9 with 2-objective and 3-objective are presented in Table I. As shown in the second last row of Table I, FDEA obtains the best results in 13 out of 24 problems, while NSGA-III, $\theta$-DEA, VaEA, MaOEA/C, DDEA, MOEA/D-LTD, and PaRP/EA perform best in 2, 2, 0, 1, 3, 2, and 1 problems, respectively. According to the Wilcoxon rank-sum test, when compared to NSGA-III, $\theta$-DEA, VaEA, MaOEA/C, DDEA, MOEA/D-LTD, and PaRP/EA, FDEA is, respectively, worse in 2, 3, 0, 1, 5, 3, and 1 out of 24 cases, while it is, respectively, better in 17, 18, 21, 16, 15, 18, and 18 cases, which validate the superiority of FDEA for tackling these 2-objective and 3-objective WFG and WFG4x problems. Particularly, FDEA outperforms all competitors on WFG3, WFG42, WFG44, and WFG9, whereas it performs worse than NSGA-III and $\theta$-DEA on WFG41 and WFG45, and worse than DDEA and MOEA/D-LTD on WFG1.

To visually show the performance, the final solution sets with the median IGD values obtained by FDEA and its seven competitors on these considered WFG and WFG4x problems with 2 and 3 objectives are plotted in Figs. A1–A24, which are provided in the supplementary material due to page limitations. As observed from these figures, the test problems have different PF shapes. For WFG1 with a mixed and biased PF in Figs. A1 and A2, all algorithms perform poorly on convergence, while only DDEA and MOEA/D-LTD perform better.
Based on the above analysis, NSGA-III and $\theta$-DEA can only solve the WFG and WFG4x problems with regular PFs (such as WFG41 and WFG46), as they use the fixed and evenly distributed weight vectors. MOEA/D-LTD can adaptively adjust the weight vectors and aggregation method by a learning model, which enhances the performance in some irregular problems (such as WFG1 and WFG43), but it still faces some challenges on problems with discontinuous PFs, such as WFG2, WFG47, and WFG48, especially in their 3-objective cases. Moreover, although solutions are treated as weight vectors in MaOEA/C, VaEA, and DDEA, they still fail to solve problems with convex PFs (such as WFG42 and WFG44) or with a sharp convex segment on PFs (such as WFG2, WFG47, and WFG48). Furthermore, PaRP/EA cannot deal with problems having discontinuous, extremely convex, or convex PFs (such as WFG43, WFG44, and WFG47), even when using two adversarial directions. Considering our FDEA, as the fuzzy decomposition includes a fuzzy prediction to estimate the PF shape and a weight vector extraction to obtain $N$ constrained subproblems, it can properly solve different problems by adaptively fitting their complex PF shapes. Based on the experimental studies on WFG and WFG4x problems, it is reasonable to conclude that FDEA achieves a superior performance on all of these selected problems except for WFG1.
on the 5-objective case, where MOEA/D-LTD is better than FDEA in 6 out of 12 problems and has the best performance rank (1.75), while FDEA is ranked with 2.9176. According to the statistical results for all cases in the last row of Table II, FDEA is advantageous on solving the WFG and WFG4x problems as it has the best rank (2.3403) in terms of HV.

D. Comparison of Results on the MaF Problems

As discussed above, FDEA is highly competitive or significantly superior to its competitors on the WFG and WFG4x problems with discontinuous, convex, or sharp concave PFs. In this section, the performance of FDEA is further studied on solving the problems with other irregular PF shapes, for example, the inverted and degenerated cases. Here, FDEA is compared with NSGA-III, \( \theta \)-DEA, VaEA, MaOEA/C, MOEA/AD, PaRP/EA, and DDEA on solving MaF1–MaF9 and MaF13 with different objectives \( m \in \{3, 5, 8, 10, 15\} \). As MOEA/AD adopts two adversarial sets of weight vectors to obtain two solution sets correspondingly, only the solution set with the better IGD/HV performance is reported as its final solution set.

1) IGD Results on Problems With Three Objectives: The average IGD results of FDEA and its seven competitors on MaF1–MaF7 problems with three objectives are shown in Figs. A25–A31, which are provided in the supplementary material due to page limitations. From these figures, it can be observed that each MaF problem has a different and irregular PF shape. For MaF1 with an inverted linear PF (in Fig. A25), MOEA/AD and FDEA obtain the best solution sets in view of distribution, while NSGA-III and \( \theta \)-DEA generate poorly distributed solutions. On MaF2 with a partially mild concave PF (in Fig. A26), FDEA obtains the best solution set in terms of convergence and distribution. Considering MaF3 with a convex PF (in Fig. A27), only FDEA can obtain an evenly distributed solution set, while the other competitors can only find solutions concentrating on the central part of the PF. Regarding MaF4 with an inverted concave PF (in Fig. A28), FDEA performs slightly better than the other algorithms, although they actually cannot obtain an evenly distributed solution set. For MaF5 with a badly scaled concave PF (in Fig. A29), FDEA only obtains the second-best result, while PaRP/EA performs best. On MaF6 with a degenerated PF (in Fig. A30), NSGA-III, \( \theta \)-DEA, and MOEA/AD cannot find an evenly distributed solution set, while other algorithms can almost cover the whole PF completely. Regarding MaF7 with four discontinuous PF segments (in Fig. A31), FDEA performs best as it can find solutions spread well on all segments of the PF.

Through the above experimental studies on the MaF problems, some conclusions can be drawn as follows. As the fixed weight vectors in NSGA-III and \( \theta \)-DEA cannot properly match the target problems’ PFs, such as inverted PFs in MaF1 and MaF4, degenerated PFs in MaF6, and discontinuous PFs in MaF7, they perform poorly on these problems. Although two adversarial sets of weight vectors are considered as MOEA/AD, \( \theta \)-DEA, and MOEA/AD cannot find an evenly distributed solution set, while other algorithms can almost cover the whole PF completely. Regarding MaF7 with four discontinuous PF segments (in Fig. A31), FDEA performs best as it can find solutions spread well on all segments of the PF.
TABLE IV
SUMMARY OF SIGNIFICANCE TEST BETWEEN FDEA AND SEVEN MOEAS ON MAF1–MAF7 PROBLEMS WITH HV

<table>
<thead>
<tr>
<th>Comparisons on</th>
<th>NSGA-II</th>
<th>$\theta$-DEA</th>
<th>VaLEA</th>
<th>MaOEA/C</th>
<th>MOEA/AD</th>
<th>DDEA/NS</th>
<th>PaRP/EA</th>
<th>FDEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m = 3$</td>
<td>0/6/1</td>
<td>5.7143</td>
<td>0/6/1</td>
<td>5.2143</td>
<td>0/5/2</td>
<td>4.2145</td>
<td>0/4/3</td>
<td>4.7143</td>
</tr>
<tr>
<td>$m = 5$</td>
<td>1/6/0</td>
<td>4.7143</td>
<td>1/6/0</td>
<td>5.8571</td>
<td>0/6/1</td>
<td>4.7143</td>
<td>0/5/2</td>
<td>4.1429</td>
</tr>
<tr>
<td>$m = 8$</td>
<td>0/6/1</td>
<td>5.00</td>
<td>0/7/0</td>
<td>6.00</td>
<td>0/7/0</td>
<td>5.1429</td>
<td>3/3/2</td>
<td>3.00</td>
</tr>
<tr>
<td>$m = 10$</td>
<td>1/4/2</td>
<td>3.6429</td>
<td>2/5/0</td>
<td>4.8571</td>
<td>1/6/0</td>
<td>5.50</td>
<td>3/3/2</td>
<td>3.2286</td>
</tr>
<tr>
<td>$m = 15$</td>
<td>2/4/1</td>
<td>5.00</td>
<td>2/4/1</td>
<td>4.5714</td>
<td>2/4/1</td>
<td>4.7857</td>
<td>3/2/2</td>
<td>3.50</td>
</tr>
<tr>
<td>All</td>
<td>4/26/5</td>
<td>4.8143</td>
<td>5/28/2</td>
<td>5.3</td>
<td>3/28/4</td>
<td>4.8714</td>
<td>10/17/11</td>
<td>3.7571</td>
</tr>
</tbody>
</table>

Average Ranks based on HV results

Fig. 8. Average ranks for all considered MOEAs by the Friedman test based on all HV results for all MaF, WFG, and WFG4X problems with 3–15 objectives.

Average Ranks based on IGD results

Fig. 9. Average ranks for all considered MOEAs by the Friedman test based on all IGD results for all MaF and WFG problems with 3–15 objectives.

FDEA that regard solutions as weight vectors, FDEA is the best to solve the MaF problems, as the used fuzzy prediction can estimate the PF shapes, which helps to run the weight vector extraction in fuzzy decomposition.

2) HV Results on Problems With 3–15 Objectives: Due to page limitations, the median HV and IGD values of all the compared algorithms on MaF1–MaF13 with 3–15 objectives are given in Tables A.VII and A.VIII of the supplementary material, whereas the statistical test results on MaF1–MaF7 with HV are summarized in Table IV. Obviously, FDEA is significantly superior to its seven competitors on MaF1–MaF7, as it has the best rank (2.2286) in terms of HV.

E. Comparison Results on More Algorithms

To further validate the performance of FDEA, another eight MOEAs, including SPEA2 [10], MOEA/D-PaS [37], BCE-MOEA/D [6], RVEA [44], KnEA [5], hpaEA [53], MaOEA-IGD [11], and GFM-MOEA [75], are considered here for the experimental studies. For a fair comparison, the parameter settings of these MOEAs are provided in Table A.II of the supplementary material, as suggested in their references, and other parameters are set the same with that described in Sections IV-A and IV-B. Moreover, the HV and IGD results of these MOEAs on WFG1–WFG9, WFG41–WFG48, and MaF1–MaF13 test problems with 3–15 objectives are provided in Tables A.IX–A.XI of the supplementary material. Moreover, two overall average ranks of each MOEA considered in this article are obtained by the Friedman test on KEEL [71], respectively, based on these IGD and HV results, which are plotted in Figs. 8 and 9. From these summarized
results, FDEA obtains the best ranks on both HV-based and IGD-based rankings, which validate its superior performance when compared to these MOEAs.

F. More Discussions About the Shared Weight Vector and the Predicted Values for $p$

Here, two variants of FDEA are designed to study the influence of the shared $w$, with one using a fixed $w = (1, 1, \ldots, 1)$ and the other one without using the shared $w$ for the subproblems. Moreover, four other variants of FDEA are designed to study the influence of $p$, where $p$ is not fuzzily predicted but fixed as $p = 0.5$, $p = 1.0$, and $p = 2.0$, respectively, for first three of these variants, and $p$ is obtained by (9) instead of (10) for the last variant. Furthermore, the observations of the predicted value of $p$ on all considered MOPs during the evolutionary process are also studied. Due to page limitations, details for the comparison results of these six variants with FDEA and the observations of $p$ are provided in the supplementary file.

V. CONCLUSION AND FUTURE WORK

In this article, a fuzzy decomposition-based MOEA has been proposed for tackling various MOPs, that is, FDEA. This algorithm can fuzzily decompose an MOP into a set of constrained subproblems. To do this, fuzzy decomposition was run by using the fuzzy prediction to estimate the PF shapes and employing weight vector extraction to select solutions. Note that the fuzzy prediction will finally estimate a UHP, which helps to define a more precise metric for computing the direction similarity between solutions in the weight vectors extraction. This way, the decomposed subproblems can well fit the target problem’s PF. Finally, only $m$ corner solutions were selected to maintain diversity and other solutions were chosen for each of the remaining subproblems. In this case, the best convergence can be achieved by using the WS aggregated function and the shared weight vector from all the extracted weight vectors. When compared to eight competitive MOEAs (NSGA-III, $\theta$-DEA, VaEA, MaOEAC, DDEA, MOEA/D-LTD, MOEA/AD, and PaRP/E), FDEA had shown to be better in most cases, especially on problems with irregular PFs.

In our future work, the prediction of the population’s shape with more complicated models and the measurement of direction similarity between solutions will be further studied on more complicated MOPs. The application of FDEA on the real-world problems will be also conducted in our future work.

REFERENCES


Songbai Liu received the B.S. degree from Changsha University, Changsha, China, in 2012, and the M.S. degree from Shenzhen University, Shenzhen, China, in 2018. He is currently pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong, Hong Kong. He worked with Shenzhen TVT Digital Technology Company, Ltd., Shenzhen, as a Software QA Engineer from 2013 to 2015, and he worked with Shenzhen University as a Research Assistant from 2018 to 2019. His current research interests include nature-inspired computation, evolutionary many-objective optimization, and machine learning.

Qiuzhen Lin (Member, IEEE) received the B.S. degree from Zhaoqing University, Zhaoqing, China, in 2007, the M.S. degree from Shenzhen University, Shenzhen, China, in 2010, and the Ph.D. degree from the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 2014. He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University. He has published over twenty research papers since 2008. His current research interests include artificial immune system, multiobjective optimization, and dynamic system.

Kay Chen Tan (Fellow, IEEE) received the B.Eng. degree (First Class Hons.) in electronics and electrical engineering and the Ph.D. degree in evolutionary computation and control systems from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively. He is a Full Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. He has published over 200 refereed articles and six books. Prof. Tan is the Editor-in-Chief of the IEEE Transactions on Evolutionary Computation, was the Editor-in-Chief of the IEEE Computational Intelligence Magazine from 2010 to 2013, and currently serves as the Editorial Board Member of over ten journals. He was an Elected Member of IEEE CIS AdCom from 2017 to 2019.

Carlos A. Coello Coello (Fellow, IEEE) received the Ph.D. degree in computer science from Tulane University, New Orleans, LA, USA, in 1996. He is a Professor (CINVESTAV-IPN Researcher) with the Department of Computer Science of CINVESTAV-IPN, Mexico City, Mexico. He has authored and coauthored over 450 technical papers and book chapters. He has also coauthored the book Evolutionary Algorithms for Solving Multi-Objective Problems (Second Edition, Springer, 2007). His publications currently report over 48 900 citations in Google Scholar (his H-index is 80). His research interests include evolutionary multiobjective optimization and constraint-handling techniques for evolutionary algorithms.

Maoguo Gong (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electronic science and technology from Xidian University, Xi’an, China, in 2003 and 2009, respectively. Since 2006, he has been a Teacher with Xidian University. In 2008 and 2010, he was promoted as an Associate Professor and a Full Professor, respectively, both with exceptive admission. His current research interests include computational intelligence with applications to optimization, learning, data mining, and image understanding.

Dr. Gong was a recipient of the Prestigious National Program for the support of Top-Notch Young Professionals from the Central Organization of China, the Excellent Young Scientist Foundation from the National Natural Science Foundation of China, and the New Century Excellent Talent in University from the Ministry of Education of China. He is the Vice Chair of the IEEE Computational Intelligence Society Task Force on Memetic Computing, an Executive Committee Member of the Chinese Association for Artificial Intelligence, and a Senior Member of the Chinese Computer Federation. He is also an Associate Editor of the IEEE Transactions on Evolutionary Computation.

Prof. Coello Coello was a recipient of the 2007 National Research Award from the Mexican Academy of Sciences in the area of Exact Sciences, the 2013 IEEE Kiyo Tomiyasu Award, and the 2012 National Medal of Science and Arts in the area of Physical, Mathematical and Natural Sciences. He is an Associate Editor of the IEEE Transactions on Evolutionary Computation and serves on the editorial board of 12 other international journals. He is a member of the Association for Computing Machinery, Sigma Xi, and the Mexican Academy of Science.