Adapting Reference Vectors and Scalarizing Functions by Growing Neural Gas to Handle Irregular Pareto Fronts

Yiping Liu, *Member, IEEE*, Hisao Ishibuchi, *Fellow, IEEE*, Naoki Masuyama, *Member, IEEE*, and Yusuke Nojima, *Member, IEEE*

***Abstract*—The performance of decomposition-based multi-objective evolutionary algorithms often deteriorates clearly when solving multi-objective optimization problems with irregular Pareto fronts. The main reason is the improper settings of reference vectors and scalarizing functions. In this study, we propose a decomposition-based multi-objective evolutionary algorithm guided by a growing neural gas network, which learns the topological structure of the Pareto front. Both the reference vectors and the scalarizing functions are adapted based on the topological structure to enhance the evolutionary algorithm's search ability. The proposed algorithm is compared with eight state-of-the-art optimizers on 34 test problems. The experimental results demonstrate that the proposed method is competitive in handling irregular Pareto fronts.**

***Index Terms*—Decomposition-based multi-objective evolutionary optimization, growing neural gas, irregular Pareto front.**

I. Introduction

MULTI-objective optimization problems (MOPs) are commonly seen in a variety of disciplines. They involve multiple objectives to be optimized simultaneously. Without loss of generality, an MOP can be formulated as follows:

$$\begin{aligned} &\min \boldsymbol{f}(\boldsymbol{x}) = \min(f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})), \\ &\text{s.t. } \boldsymbol{x} \in S \subset \boldsymbol{R}^n, \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is an n-dimensional decision vector in the search space S, $f_m(\boldsymbol{x})$ is the m-th objective to be minimized ($m = 1, ..., M$), and M is the number of objectives. When $M > 3$, this problem is often called a many-objective optimization problem (MaOP). Due to the conflicting nature of objectives, usually an MOP has no single optimal solution which simultaneously optimizes all objectives. It has a set of trade-off solutions, known as the Pareto optimal solution set (PS). The image of the PS in the objective space is called the Pareto front (PF).

Over the past two decades, a number of Multi-Objective Evolutionary Algorithms (MOEAs) have been proposed, which have been proven successful in solving MOPs. Generally, MOEAs can be classified into three categories, i.e., Pareto-, indicator-, and decomposition-based MOEAs.

Among others, decomposition-based MOEAs are very competitive, where an MOP is decomposed into a number of single-objective problems based on a scalarizing function and a set of well distributed reference vectors. A population is guided to search towards the PF in the directions specified by the reference vectors. Decomposition-based MOEAs have distinct advantages due to the above-mentioned key feature. For example, they converge fast and have the capability of solving MaOPs owing to the strong selection pressure towards the PF [1]–[4]. They also have good exploitation ability by incorporating local search operators [5], [6]. In addition, they are able to maintain diversity in the decision space by working with niche methods [7], [8].

In decomposition-based MOEAs, the distribution of obtained solutions is determined by both the reference vectors and the scalarizing function. In most existing studies, the reference vectors are predefined and uniformly distributed in the entire objective space, and the same scalarizing function is used for all reference vectors. Under this specification, a decomposition-based MOEA works well on an MOP with a simple PF. For example, MOEA based on Dominance and Decomposition (MOEA/DD) [9] is very powerful in solving DTLZ1 to DTLZ4 [10]. We regard a PF like those of DTLZ1 to DTLZ4 as a regular PF, if (1) the geometric shape of the PF is simple, such as a hyperplane or concave hypersphere; (2) any positive vector emanated from the ideal point intersects with the PF; (3) the range of every objective on the PF is almost the same.

An MOP is unlikely to have a regular PF unless it is particularly man-made to meet the above conditions. Therefore, a real-world MOP usually has an irregular PF, which can be highly non-linear, disconnected, degenerated, incomplete, badly-scaled, and/or high-dimensional. For example, due to the complicated non-linear relationship between objectives, the PF of a carbon fiber drawing process optimization problem is disconnected into several parts [11]. Another example is a financial portfolio management problem [12] where the ranges of objective values on the PF are quite different, which makes

This work was supported by National Natural Science Foundation of China (Grant No. 61876075 and 61803192), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No. 2017ZT07X386), Shenzhen Peacock Plan (Grant No. KQTD2016112514355531), the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. ZDSYS201703031748284), the Program for University Key Laboratory of Guangdong Province (Grant No. 2017KSYS008), and JSPS KAKENHI (Grant No. 19K20358).

Y. Liu, N. Masuyama, Y. Nojima are with Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University, Sakai, Osaka 599-8531, Japan. (yiping0liu@gmail.com, masuyama@cs.osakafu-u.ac.jp, nojima@cs.osakafu-u.ac.jp)

H. Ishibuchi is with Shenzhen Key Laboratory of Computational Intelligence, University Key Laboratory of Evolving Intelligent Systems of Guangdong Province, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China. (hisao@sustech.edu.cn) (*Corresponding Author*)

the PF badly scaled. More real-world MOPs with irregular PFs were reported, such as a job shop scheduling problem [13] and a water resource management problem [14]. When solving such an MOP, the performance of a decomposition-based MOEA often deteriorates clearly [1]. One reason is the inconsistency between the distribution of the reference vectors and the PF shape. If the PF is irregular, some reference vectors may have no intersections with the PF. Consequently, the obtained solution by those reference vectors could be very close to each others (or close to some other reference vectors), which decreases the diversity of the obtained solution set. Another reason is the choice of an improper scalarizing function for each reference vector, which results in solutions far away from the PF (poor convergence) or the reference vectors (poor diversity).

To address these issues, in this study we propose a Decomposition-based multi-objective Evolutionary Algorithm guided by Growing Neural Gas (DEA-GNG). In the proposed method, we employ a GNG network [15] to learn the topological structure of the PF. GNG adaptively organizes a topological network by a set of nodes and edges. The edges connect the nodes and represent the relationships among them. Inputting the known solutions as signals, the network grows as the population evolves, and it reflects the topology of the PF more and more closely. A node in GNG indicates a position on the PF, while the edges emanated from the node provide the curvature information of the position. Thus we can use them to adapt the reference vectors and the scalarizing functions, where a different scalarizing function is used for each reference vector.

This study has the following contributions:

- A reference vector adaptation strategy based on GNG is proposed, where the nodes in GNG are adopted as reference vectors and combined with another set of uniformly distributed reference vectors to guide the evolution. Using these reference vectors, the evolutionary algorithm is capable of searching for Pareto optimal solutions according to the PF shape while having a good exploration ability in the entire objective space.
- A method to automatically choose a scalarizing function for each reference vector is proposed. The curvature information of the PF around each node (reference vector) from GNG can be estimated by the edges emanated from the node. Then, a scalarizing function that can well balance convergence and diversity is obtained according to the angels between the reference vector and the edges.
- A variant of GNG is proposed to solve some issues arising from multi-objective evolutionary optimization and to learn the topology of the PF more accurately.

The remainder of this paper is organized as below. In Section II, related studies on multi-objective evolutionary optimization are reviewed for the completeness of the presentation. The motivation of this work is also elaborated. The proposed DEA-GNG is described in detail in Section III. Section IV presents the experimental design and results. Section V concludes the paper and provides future research directions.

II. Preliminaries

A. Multi-Objective Evolutionary Optimization

A large number of MOEAs have been proposed and can be categorized into Pareto-, indicator-, and decomposition-based algorithms.

Pareto-based MOEAs, such as Non-dominated Sorting Genetic Algorithm II (NSGA-II) [16] and Strength Pareto Evolutionary Algorithm 2 (SPEA2) [17], adopt the Pareto dominance- and the density-based criteria to select non-dominated solutions with good diversity. However, they generally fail in solving MaOPs efficiently. The main reason is that the proportion of non-dominated solutions in the population considerably rises as the number of objectives increases, and then the density-based criterion plays a key role in the environmental and mating selection phases. However, the density-based criterion often select a solution far away from the PF due to its low density value in the high-dimensional objective space. To improve their ability in solving MaOPs, many attempts have been reported, e.g., dominance area control [18], Pareto partial dominance [19], ε-dominance [20], preference order ranking [21], and fuzzy Pareto dominance [22]. These methods actually introduce a kind of preference to Pareto dominance, which would lead the population converge to certain regions of the PF [23]. As a result, they may be not suitable to search for the entire PF.

Indicator-Based Evolutionary Algorithms (IBEAs) [24] adopt a performance indicator to evaluate the fitness of each solution. Solutions with the best indicator values are preferred. Hypervolume (HV) is one of the most widely-used indicators in IBEAs. To reduce computational complexity for calculating HV when solving MaOPs, HypE [25] uses a Monte Carlo simulation to estimate HV. However, the reference point for calculating HV is still difficult to specify for an irregular PF. Recently, the R2 indicator [26], [27] was proposed to estimate HV with much lower complexity. It requires a set of reference vectors as in decomposition-based algorithms.

MOEA/D [28] is one of the most famous decomposition-based algorithms. In MOEA/D, uniformly distributed reference vectors are first generated. Then, the weighted sum approach, the Tchebycheff approach, or the penalty-based boundary intersection (PBI) approach is used to decompose the MOP into a number of single-objective subproblems. Uniformly distributed reference vectors with the same scalarizing function would appreciably decrease the performance of decomposition-based algorithms on irregular PFs, which have been shown in a recent study [1]. Currently, there exist two research directions to address this issue.

One is to adapt scalarizing functions. There are only a few studies along this direction. In [29], the weighted sum approach and the Tchebycheff approach are automatically chosen according to the number of neighbors of a solution. Later, the same authors proposed to simultaneously use the two approaches in MOEA/D [30]. In [31], a Pareto adaptive scalarizing method named MOEA/D-PaS was proposed to approximate the optimal p value in the L_p scalarizing function. In this method, several candidate p values are pre-defined. Then, for each reference vector, the one that can select a

solution closest to the reference vector is chosen. Following the same idea, a method to adaptive parameter θ in the PBI function was proposed in [32]. MOEA/D using a learning-to-decompose method (MOEA/D-LTD) was proposed very recently in [33]. A Gaussian process (GP)-based model is employed to learn the PF shape and to adjust parameters in a modified PBI function.

The other is to adapt reference vectors. In MOEA/D with adaptive weight adjustment (MOEA/D-AWA) [34], the most crowded reference vectors are periodically removed while new reference vectors are generated in sparse regions. NSGA-III [2] is an improved version of NSGA-II, which employs uniformly distributed reference points (vectors) for diversity maintenance. In [35], an adaptive version of NSGA-III (A-NSGA-III) is presented, where additional reference vectors are generated around the one crowded by solutions. In a modified version of reference vector guided evolutionary algorithm (RVEA) named RVEA* [3], adaptive weight update is performed by randomly generating reference vectors according to obtained solutions. An MOEA based on hierarchical decomposition (MOEA/HD) was proposed in [36]. In MOEA/HD, the subproblems are layered into different hierarchies, and the search directions of lower-hierarchy subproblems are adaptively adjusted according to the higher-hierarchy search results. In [11], a clustering-based adaptive MOEA (CA-MOEA) was introduced, where candidate solutions are clustered in environmental selection. Each cluster center can be regarded as a reference point (vector). The solution closest to each reference point is selected. An approach to adapt the weights (reference vectors) (AdaW) was proposed in [37]. In AdaW, diverse non-dominated solutions are maintained in an archive to guide the addition and deletion of reference vectors. The results showed that AdaW is able to obtain good diversity on various Pareto front shapes. In the aforementioned MOEA/D-LTD [33], the GP-based model is also used to generate the reference vectors. In addition, some other methods to adapt reference vectors based on the current population were proposed [4], [38]–[40]. Particularly, MOEA/D with self organized map-based weight (reference) vectors (MOEA/D-SOM) was recently proposed in [41]. SOM [42] is a classic topological clustering algorithm. The nodes in the SOM network trained by the previous and current populations indicates a position on the PF. Thus they are adopted to adapt reference vectors in MOEA/D-SOM.

In this study, we propose to use GNG to adapt the reference vectors and the scalarizing functions, as both adaptations are important for handling irregular PFs. GNG is also a topological clustering algorithm. It has more advantages than SOM in handling irregular PFs which will be explained in the next subsection.

B. Motivation

GNG is able to learn the topological structure in a given set of input signals (vectors) adaptively. The network of GNG consists of (1) a set of nodes which accumulate input signals and (2) a set of edges among pairs of nodes which represent the relationships among connected nodes. In this study, they are denoted as R_{node} and V_{edge}, respectively. The main steps of GNG are given as follows [15]:

Step 0. (Initialization) Start with two connected nodes at random positions $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$. Note that the age of a new generated edge and the error variable of a new generated node are 0.

Step 1. Input a signal ξ.

Step 2. Find the nearest node $\boldsymbol{r}_a$ and the second-nearest node $\boldsymbol{r}_b$ to ξ.

Step 3. Increment the age of all edges emanating from $\boldsymbol{r}_a$.

Step 4. Increase the error variable of $\boldsymbol{r}_a$ by adding the squared distance of $\boldsymbol{r}_a$ and ξ:

$$\mathrm{error}(\boldsymbol{r}_a) = \mathrm{error}(\boldsymbol{r}_a) + \|\boldsymbol{r}_a - \xi\|^2. \tag{2}$$

Step 5. Move $\boldsymbol{r}_a$ and its direct topological neighbors, $\boldsymbol{r}_k^{nb}, k = 1, \ldots, K$, (i.e., the nodes connected with $\boldsymbol{r}_a$ by an edge) towards ξ by learning rates ϵ_a and ϵ_{nb}, respectively, of the total distance:

$$\begin{aligned} \boldsymbol{r}_a &= \boldsymbol{r}_a + \epsilon_a(\xi - \boldsymbol{r}_a), \\ \boldsymbol{r}_k^{nb} &= \boldsymbol{r}_k^{nb} + \epsilon_{nb}(\xi - \boldsymbol{r}_k^{nb}). \end{aligned} \tag{3}$$

Step 6. If $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$ are connected by an edge, set the age of this edge to zero. If such an edge does not exist, create it.

Step 7. Remove edges with an age larger than $\mathrm{age}_{\max}$. If this results in nodes having no emanating edges, remove them as well.

Step 8. If the number of signals input so far is a multiple of a parameter λ and the number of nodes has not reached to maximum, insert a new node as follows:

- Determine the node $\boldsymbol{r}_1^{\max}$ with the largest error variable.
- Insert a new node $\boldsymbol{r}^{\mathrm{new}}$ halfway between $\boldsymbol{r}_1^{\max}$ and its neighbor $\boldsymbol{r}_2^{\max}$ with the largest error variable:

$$\boldsymbol{r}^{\mathrm{new}} = 0.5(\boldsymbol{r}_1^{\max} + \boldsymbol{r}_2^{\max}) \tag{4}$$

- Insert edges connecting $\boldsymbol{r}^{\mathrm{new}}$ with $\boldsymbol{r}_1^{\max}$ and $\boldsymbol{r}_2^{\max}$, and remove the original edge between $\boldsymbol{r}_1^{\max}$ and $\boldsymbol{r}_2^{\max}$.
- Decrease the error variables of $\boldsymbol{r}_1^{\max}$ and $\boldsymbol{r}_2^{\max}$ by multiplying them by a constant α. Set the error variable of $\boldsymbol{r}^{\mathrm{new}}$ to be the same as that of $\boldsymbol{r}_1^{\max}$.

Step 9. Decrease all error variables by multiplying them by a constant δ.

Step 10. If a stopping criterion (e.g., reaching the maximum number of iterations) is not yet fulfilled, go to Step 1.

Please refer to [15] for details of GNG. Although GNG has been widely used in the information modeling due to its superior flexibility and adaptability [43], [44], there exist few studies on using GNG in MOEAs. For example, in [45], [46], a model-building GNG is proposed to generate offspring for multi-objective estimation of distribution algorithms.

In this study, we adopt GNG to guide the evolution in decomposition-based MOEAs for the first time. A population in an MOEA is expected to approximate the PF and be well scattered over the PF as the search goes on [47]. Then, we can assume the known solutions as input signals sampled from the PF with noise. A GNG network trained by these known solutions will provide the topological information of the PF

more and more closely as the search goes on, which can be employed to adapt the reference vectors and the scalarizing function of each reference vector.

A SOM network also consists of nodes and edges. Besides the aforementioned MOEA/D-SOM, it has been frequently applied to multi-objective evolutionary optimization. In [48], SOM is used as a distribution probability model to generate new solutions. A method to visualize Pareto optimal solutions based on SOM is proposed in [49]. A self-organizing MOEA is presented by detecting relationships between solutions in the decision space in [50]. Actually, GNG has more advantages than SOM for handling irregular PFs due to the following reasons.

One is that GNG has a much faster convergence to low distortion errors than SOM. This is important to reduce the computation cost, since we need to continuously update GNG or SOM as the population evolves. Fig. 1 shows the networks of GNG and SOM when using uniformly sampled solutions on the PF of 3-objective Minus-DTLZ2 [1] and DTLZ7 [10] as input signals, respectively, where the dots are the input signals, the circles are the nodes in GNG or SOM, and the lines are the edges among the nodes. We can see that GNG can find a topological structure that closely reflects the PF shape of Minus-DTLZ2 and DTLZ7 at the 30th iteration (i.e., inputting all the signals 30 times) in Figs. 1 (a) and (d), respectively, while SOM cannot in Figs. 1 (b) and (e). SOM converges at the 100th (200th) iteration in Fig. 1 (c (f)). One reason for the slow convergence of SOM is that all nodes of SOM are randomly generated in the initialization phase, which may be far away from the input signals. In contrast, nodes in GNG are gradually generated according to the positions of the input signals. Another reason is the predefined connections (edges) between nodes in SOM. Two connected nodes may be far away from each other. We can see a number of such pairs of nodes in Figs. 1 (b) and (e). Making them closer takes many iterations.

The predefined structure of SOM is also the other reason for its disadvantages in handling an irregular PF. In the SOM network, the number of dimensions, the number of nodes in each dimension, and the connections between nodes should be predetermined. However, the GNG network does not require such specifications. These specifications will reduce the algorithm's flexibility in handling irregular PFs:

- The dimension of an irregular PF is usually unknown. If the dimension of the SOM network is different from that of the PF, SOM cannot learn a good topological structure of the PF. On the contrary, the dimension of the GNG network is automatically determined according to the input signals.
- The setting of population size is not arbitrary for SOM. For example, if we set the number of nodes in each dimension in a 2-D SOM network to be a and b, the total number of nodes is $a \times b$. In a decomposition-based MOEA, the population size is usually set equal to the number of weight vectors to enhance diversity. Thus when we use all the nodes in SOM as the reference vectors, the population size is $a \times b$. For GNG, we can set the maximum number of nodes to an arbitrary value. The number of nodes usually reaches the maximum number in a well-trained GNG network.
- SOM tries to represent separate clusters by a single network, whereas GNG does not try to connect separate clusters into a single network. That is, there is no edge connecting separate clusters in GNG. This is very important for handling disconnected PFs. Let us check Fig. 1 (d), where the network in GNG is partitioned into four sub-networks[1] according to the PF shape of DTLZ7. SOM performs poorly even at the 100th iteration in Fig. 1 (e). It finally converges at the 200th iteration in Fig. 1 (f). However, we can observe that there are still some nodes not related to the input signals.

For these reasons, we propose to use GNG to guide the evolution by adapting the reference vectors and the scalarizing function for each eference vector in decomposition-based MOEAs. In the next section, we will introduce the proposed DEA-GNG by answering the following questions.

- How to get the input signals to train GNG? There are a huge number of solutions generated during the evolution. It is impractical and inefficient to input all of them to GNG as signals.
- Is there any flaw of GNG when using it in decomposition-based MOEAs?
- How to adapt the reference vectors based on the obtained GNG network? Can we directly adopt the nodes in GNG as the reference vectors?
- How to choose a proper scalarizing function for each reference vector based on the topological information from GNG?

III. Proposed Method

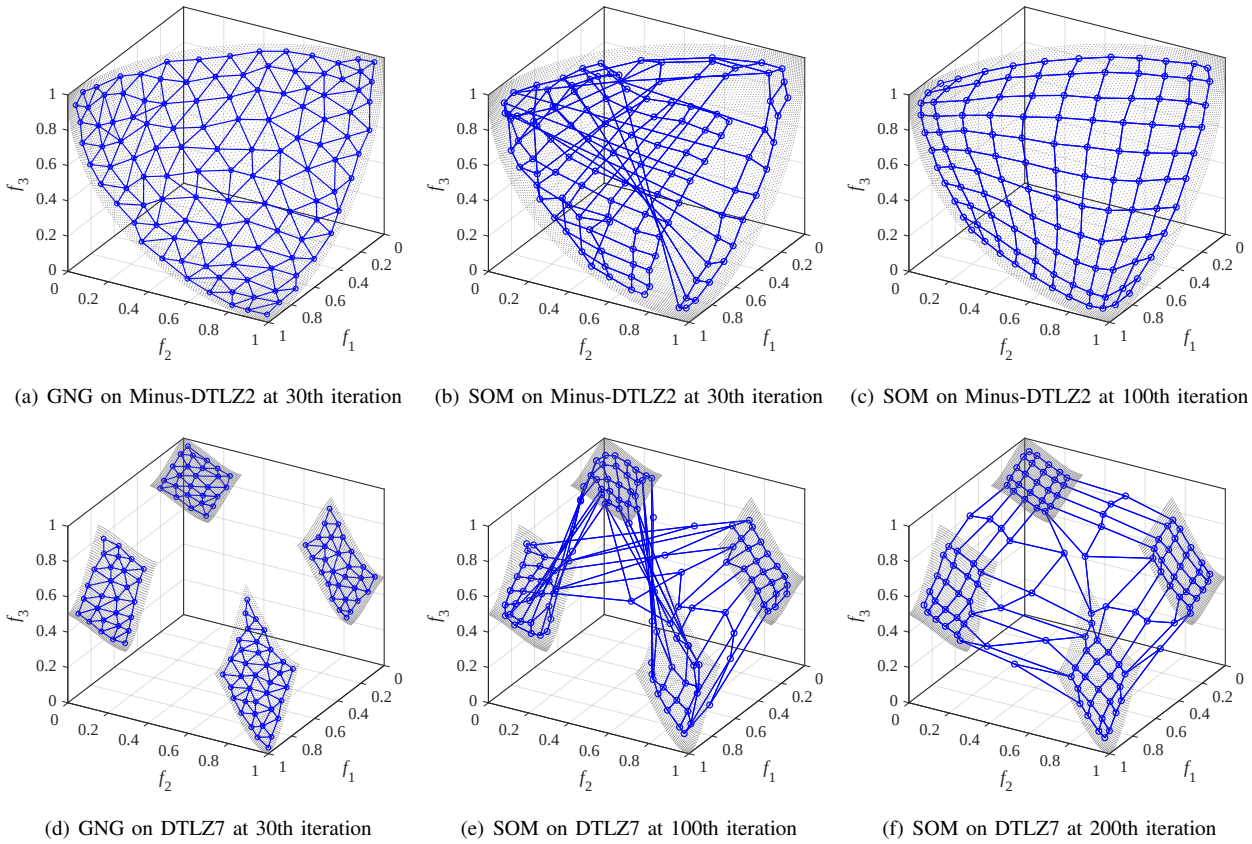
Our DEA-GNG can be divided into two parts. One is the DEA-based optimization model. The other is the GNG-based learning model. First, we present the general framework of DEA-GNG in Subsection III-A. Then, the key techniques in the two models are described in detail in Subsections III-B and III-C to III-F, respectively. Finally, in Subsection III-G, we discuss the similarities and differences between DEA-GNG and existing methods.

A. General Framework

Algorithm 1 presents the overall framework of the proposed DEA-GNG.

Lines 1 to 5 in Algorithm 1 are the initialization phase. $g = 0$ is the count value of generation (line 1). The population P is randomly initialized in the decision space (line 2). Function *Ideal_Point* returns the minimum objective values as the estimated ideal point $\boldsymbol{z}^* = (z_1^*, \ldots, z_M^*)$ (line 3). R_u is a set of uniformly distributed reference vectors whose size is equal (or similar) to the population size N. R is the reference vectors to be used in selection (line 4). The input signal archive A_S with the maximum size N_S is initialized to be an empty set. (line 5).

[1]The term "sub-network" means that any node in a sub-network has no connection with any node in the other sub-networks.



(a) GNG on Minus-DTLZ2 at 30th iteration (b) SOM on Minus-DTLZ2 at 30th iteration (c) SOM on Minus-DTLZ2 at 100th iteration

(d) GNG on DTLZ7 at 30th iteration (e) SOM on DTLZ7 at 100th iteration (f) SOM on DTLZ7 at 200th iteration

Fig. 1. The networks of GNG and SOM using solutions uniformly sampled on the true PF of Minus-DTLZ2 and DTLZ7 as the input signals, respectively. The dots are the input signals, the circles are the nodes in GNG or SOM, and the lines are the edges among the nodes.

Algorithm 1 General Framework

Require: P (Population), N (Population Size), A_S (Input Signal Archive), N_S (Maximum Size of A_S), $G_{\max}$ (Maximum Generation)

1: $g = 0$;
2: P = *Initialize* (P);
3: $\boldsymbol{z}^*$ = *Ideal_Point* (P);
4: R = R_u = *Uniform_Reference_Vector_Generation* (N);
5: $A_S = \emptyset$;
6: **while** the stopping criterion is not met, i.e., $g < G_{\max}$ **do**
7: P' = *Mating_Selection* (P);
8: P'' = *Reproduction* (P');
9: $\boldsymbol{z}^*$ = *Ideal_Point* ($P'' \cup \{\boldsymbol{z}^*\}$);
10: **if** $g < (1-\alpha) \times G_{\max}$ **then**
11: A_S = *Input_Signal_Archive_Update* ($A_S \cup P'', R, N_S, \boldsymbol{z}^*$);
12: [R_{node}, V_{edge}]= *GNG_Update* (A_S);
13: R = *Reference_Vector_Adaptation* ($R_u, R_{\text{node}}, A_S$);
14: F^S = *Scalarizing_Function_Adaptation* (R, V_{edge});
15: **end if**
16: P = *Environmental_Selection* ($P \cup P'', R, F^S, N, \boldsymbol{z}^*$);
17: $g = g + 1$;
18: **end while**
19: **return** P

In the DEA-based optimization model, there are three operations iterated until the stopping criterion defined by the maximum number of generations $G_{\max}$ is satisfied (line 6): (1) parents P' are selected by mating selection (line 7); (2) offspring P'' are generated by a reproduction operator (line 8); and (3) new population is obtained by environmental selection (line 16). In addition, $\boldsymbol{z}^*$ is updated once the offspring are produced (line 9). We will describe environmental and mating selection in Subsection III-B.

There are four operators in the GNG-based learning model: (1) input signal archive update (line 11), (2) GNG update (line 12), (3) reference vector adaptation (line 13), and (4) scalarizing function adaptation (line 14). They will be explained in detail in Subsections III-C to III-F to answer the four questions in Subsection II-B, respectively. Note that these operators are not employed in the last $\alpha \times G_{\max}$ generations (line 10). This is because that we expect the evolution to focus on exploitation by keeping the reference vectors and the scalarizing functions unchanged. α is set to 0.1 in this study. Please refer to the supplementary material for the investigation on the effect of α.

B. Environmental and Mating Selection

The environmental selection in DEA-GNG is presented in Algorithm 2. It is similar to that in NSGA-III [2]. The main difference is that the reference vectors $R = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{|R|}\}$

Algorithm 2 Environmental Selection

Require: P (Population), N (Population Size), P'' (Offspring), $R = \{\boldsymbol{r}_1, \dots, \boldsymbol{r}_{|R|}\}$ (Reference Vectors), $F^S = \{f_1^S, \dots, f_{|R|}^S\}$ (Scalarizing functions), $\boldsymbol{z}^*$ (Ideal Point)

1: Use non-dominated sorting [16] to divide $P \cup P''$ into $\{F_1, \dots, F_i, \dots\}$, where the index i in F_i shows the minimum value of i such that $|F_1| + \cdots + |F_i| \geq N$;
2: $P = F_1 \cup \cdots \cup F_{i-1}$;
3: Set $\boldsymbol{z}'$ to be the maximum objective values in F_1;
4: Normalize every solution in $P \cup F_i$ by $\boldsymbol{f}'(\boldsymbol{x}) = \frac{\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{z}^*}{\boldsymbol{z}' - \boldsymbol{z}^*}$;
5: **for all** $\boldsymbol{x} \in P \cup F_i$ **do**
6: $\quad \boldsymbol{r}_{\min} = \arg\min_{\boldsymbol{r} \in R} \angle(\boldsymbol{f}'(\boldsymbol{x}), \boldsymbol{r})$;
7: $\quad$ Associate $\boldsymbol{x}$ to $\boldsymbol{r}_{\min}$;
8: **end for**
9: For each $\boldsymbol{r}_j \in R$ ($j = 1, \dots, |R|$), denote the number of associated solutions in P as c_j. Set $C = \{1, \dots, |R|\}$;
10: For each $\boldsymbol{r}_j \in R$ ($j = 1, \dots, |R|$), denote the associated solution subset in F_i as Δ_j;
11: **while** $|P| < N$ **do**
12: $\quad J = \arg\min_{j \in C} c_j$;
13: $\quad$ **if** $\Delta_J = \emptyset$ **then**
14: $\quad\quad$ Remove J from C;
15: $\quad$ **else**
16: $\quad\quad$ **if** $c_J = 0$ **then**
17: $\quad\quad\quad \boldsymbol{x}_{\min} = \arg\min_{\boldsymbol{x} \in \Delta_J} f_J^S(\boldsymbol{x})$;
18: $\quad\quad\quad$ Move $\boldsymbol{x}_{\min}$ from Δ_J to P;
19: $\quad\quad$ **else**
20: $\quad\quad\quad$ Randomly move a solution from Δ_J to P;
21: $\quad\quad$ **end if**
22: $\quad\quad c_J = c_J + 1$;
23: $\quad$ **end if**
24: **end while**
25: **return** P

and the scalarizing functions $F^S = \{f_1^S, \dots, f_{|R|}^S\}$ are both adapted based on the GNG network. Note that in line 4 (also in Algorithm 3, line 12), $\boldsymbol{v}^1./\boldsymbol{v}^2$ is the vector with elements v_m^1/v_m^2, where m denotes the mth dimension. In line 6, $\angle(\boldsymbol{f}'(\boldsymbol{x}), \boldsymbol{r})$ is the angle between $\boldsymbol{f}'(\boldsymbol{x})$ and $\boldsymbol{r}$.

We use the tournament selection as the mating selection, where two solutions are randomly chosen and compete under two selection criteria: The primary criterion is the non-dominated ranking and the secondary criterion is the c_j value of the corresponding reference vector. That is, the solution in the lower front with the smaller c_j is selected.

C. Input Signal Archive Update

A large number of solutions are generated during the evolution. Among them, non-dominated solutions are close to the true PF. Therefore, we only use these solutions as the input signals for training GNG. However, the number of non-dominated solutions is still huge especially when solving MaOPs. It is inefficient to use all of them as the input signals.

The input signal archive update method is similar to the environmental selection in Algorithm 2, where P and N are replaced by A_S and N_S, respectively. There are two differences. The first is that i in F_i is set to 1 in line 1. That is, we only reserve solutions in the first non-dominated front. The other is to modify lines 16 to 21 as follows:

1: **if** $c_J = 0$ **then**
2: $\quad \boldsymbol{x}_{\min} = \arg\min_{\boldsymbol{x} \in \Delta_J} \angle(\boldsymbol{f}'(\boldsymbol{x}), \boldsymbol{r}_J)$;
3: $\quad$ Move $\boldsymbol{x}_{\min}$ from Δ_J to P;
4: **else**
5: $\quad$ **if** $c_J < c_{\max}$ **then**
6: $\quad\quad \boldsymbol{x}_{\max} = \arg\max_{\boldsymbol{x} \in \Delta_J} \angle(\boldsymbol{f}'(\boldsymbol{x}), \boldsymbol{r}_J)$;
7: $\quad\quad$ Move $\boldsymbol{x}_{\max}$ from Δ_J to P;
8: $\quad$ **else**
9: $\quad\quad$ Randomly move a solution from Δ_J to P;
10: $\quad$ **end if**
11: **end if**

Note that here in line 5, $c_{\max}$ is a predefined integer. We will explain its setting later.

The reasons for the above modifications are as follows:

(1) We select the solution closest to the reference vector according to the angle between them to maintain the distribution according to the current reference vectors. (lines 1 to 3)
(2) We also select the farthest solutions (i.e., the solutions with the largest angles to the reference vector) to diversify A_S. (lines 5 to 7)
(3) Selecting too many farthest solutions may result in some solutions very close to each other. For example, when solving a bi-objective optimization problem, if we select more than one farthest solution for each reference vector, there could be some solutions close to each other. Therefore, we only select $M-1$ farthest solutions (M is the number of objectives). Considering the closest solution which has been selected in line 3, $c_{\max}$ is set to be M.
(4) If both the closest and farthest solutions have been selected, we randomly select solutions to further promote diversity. (line 9)

Under these modifications, we can diversify A_S while maintaining the distribution according to the current reference vectors. Although the modifications may decrease the uniformity of signals in A_S, the nodes of GNG are less dense and thus more uniform than the input signals.

It is interesting to note that our method shares some common ideas with the two-archive evolutionary algorithm (called C-TAEA) recently proposed in [51]. C-TAEA has the convergence- and diversity-oriented archives. The convergence-oriented archive and the population in DEA-GNG have similar selection criteria. Both the diversity-oriented archive and the input signal archive in DEA-GNG attempt to promote diversity while maintaining the current distribution of solutions. Their difference is that the diversity-oriented archive selects solutions with large constraint violation values to solve a constrained MOP, whereas the input signal archive diversifies solutions in the objective spaces to train the GNG network and thus to handle an irregular PF.

D. GNG Update

Once A_S is updated according to Subsection III-C, all the signals (solutions) in A_S will be input one time to update

GNG. Note that the signals are normalized into the range of $[0, 1]$ according to the maximum and minimum objective values in A_S before inputting. The maximum number of nodes is set to N in this study.

In the early stage of evolution, the input signals in A_S could be vastly different even in successive generations, whereas GNG will encounter issues when the input signals suddenly change. In this situation, some nodes could never be winners (closest to the signal) or neighbors of winners, since no signal close to them will be input in the next generations. Then, these nodes become useless and even misleading for adapting reference vectors in the later stage of evolution.

Moreover, GNG has a strategy to remove nodes having no emanating edges in Step 7 in Subsection II-B. When handling an irregular PF, such isolated nodes may certainly exist, since Pareto optimal solutions may locate there. Thus it is unnecessary to remove them.

Taking the above reasons into account, we propose a variant of GNG, where each node has a new property value called hit point (HP). The HP of a new generated node in Steps 0 and 8 in Subsection II-B is full, i.e., $\mathrm{HP}_{\max}$. We make the following modifications to the steps in Subsection II-B.

(1) In Step 2, after $\boldsymbol{r}_a$ (the winner) and $\boldsymbol{r}_b$ (the runner-up) are determined, the winner restores its HP value to $\mathrm{HP}_{\max}$, the HP of the runner-up is unchanged, and the other nodes decrease their HP values by one (i.e., $\mathrm{HP} = \mathrm{HP} - 1$).
(2) In Step 7, the isolated nodes are not removed. Instead, the dead nodes (i.e., the nodes with $\mathrm{HP} = 0$) and the edges emanated from them are removed.

With these improvements, GNG can learn the topological structure of the PF more adaptively and accurately.

In this study, we set $\mathrm{HP}_{\max}$ to $2|A_S|$. This indicates that even if a node loses all input signals (i.e., loses $|A_S|$ times) in one generation, it still has a chance to compete in the next generation. However, if it fails $|A_S|$ times again in the next generation, it will be eliminated.

E. Reference Vector Adaptation

As has been shown in Figs. 1 (a) and (d), the nodes (R_{node}) in GNG are well distributed according to the PF shape. However, we cannot directly use these nodes as the reference vectors in a decomposition-based MOEA due to the following two issues. One is the coverage of the boundary of the PF. Since GNG has a clustering nature, the nodes cannot cover the boundary of PF, which indicates that the solutions obtained by the guidance of the nodes also cannot. As a result, the population may locate in a smaller and smaller area as the evolution continues. The other is that the distribution of the nodes in the early stage of the evolution could be quite different from the true PF shape. If we use them as reference vectors, the exploration ability of the decomposition-based MOEA will decrease. Therefore, our reference vector adaptation method has two phases: expansion and combination, which are presented in Algorithms 3 and 4 to address these issues, respectively.

In Algorithm 3, we expand each sub-network R^q_{node} ($q = 1, \ldots, Q$) of GNG according to the corresponding signals.

Algorithm 3 Expansion

Require: R_{node} (Nodes in GNG), A_S (Input Signal Archive)
1: Divide the GNG network into several (Q) sub-networks according to the connections among nodes. The nodes belong to each sub-network are denoted as $R^1_{\mathrm{node}}, \ldots, R^Q_{\mathrm{node}}$, respectively;
2: $A^1_S = \cdots = A^Q_S = \emptyset$;
3: **for all** $\xi \in A_S$ **do**
4: $q_{\min} = \arg\min_{q=1,\ldots,Q} d(\xi, R^q_{\mathrm{node}})$; $\triangleright$ $d(\xi, R^q_{\mathrm{node}})$ is the shortest Euclidean distance between ξ and the nodes in R^q_{node}.
5: $A^{k_{\min}}_S = A^{k_{\min}}_S \cup \{\xi\}$;
6: **end for**
7: $R'_{\mathrm{node}} = \emptyset$;
8: **for** $q = 1, \ldots, Q$ **do**
9: find the minimum and maximum values of R^q_{node} in the objective space, denoted as $\boldsymbol{f}^q_{\min,\mathrm{node}}$ and $\boldsymbol{f}^q_{\max,\mathrm{node}}$, respectively;
10: find the minimum and maximum values of A^q_S in the objective space, denoted as $\boldsymbol{f}^q_{\min,A_S}$ and $\boldsymbol{f}^q_{\max,A_S}$, respectively;
11: **for all** $\boldsymbol{r}_{\mathrm{node}} \in R^q_{\mathrm{node}}$ **do**
12:

$$\boldsymbol{r}'_{\mathrm{node}} = (\boldsymbol{r}_{\mathrm{node}} - \boldsymbol{f}^q_{\min,\mathrm{node}}) ./ (\boldsymbol{f}^q_{\max,\mathrm{node}} - \boldsymbol{f}^q_{\min,\mathrm{node}}) \cdot (\boldsymbol{f}^q_{\max,A_S} - \boldsymbol{f}^q_{\min,A_S}) + \boldsymbol{f}^q_{\min,A_S}; \quad (5)$$

13: $R'_{\mathrm{node}} = R'_{\mathrm{node}} \cup \{\boldsymbol{r}'_{\mathrm{node}}\}$;
14: **end for**
15: **end for**
16: **return** R'_{node}

First, each signal in A_S is associated to its closest sub-network, and we get the corresponding subset of signals A^q_S ($q = 1, \ldots, Q$) (lines 1 to 6). Then, after finding the ranges of each R^q_{node} and A^q_S in the objective space (lines 8 to 10), all the nodes are transformed by Eq. (5) (line 12). Finally, R'_{node} is formed by uniting all the transformed nodes (line 13).

By such expansion, the nodes in R'_{node} can well cover the boundary of A_S. There could be some inconsistency between the distributions of R'_{node} and A_S when the PF shape is complex. However, the inconsistency can be neglected when the sizes of R'_{node} and A_S are large. Fig. 2 illustrates how the GNG network in Fig. 1 (d) is expanded. We can see that there are four sub-networks in Fig. 1 (d). Each of them corresponds to a disconnected region of the PF (i.e., a subset of the input signals). In Fig. 2, each sub-network is expanded to cover its corresponding input signal subset according to their maximum and minimum objective values.

In Algorithm 4, we combine R'_{node} with R_u by removing some reference vectors in R_u which are too close to R'_{node}. First, R'_{node} is mapped to a hyperplane specified by $f_1 + \cdots + f_M = 1$. The mapped R'_{node} is denoted as R_p (lines 1 to 5). Note that R_u is regularly generated on the hyperplane in this study. Next, we calculate the average Euclidean distance d_p between every pair of connected nodes in R_p (line 6). We also calculate the minimum Euclidean distance d_u among the

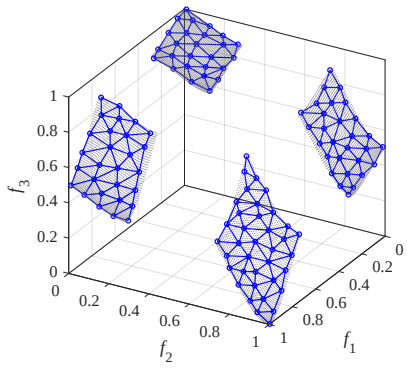


Fig. 2. Expanding each sub-network based on its corresponding input signals in Fig. 1 (d) by Eq. (5).

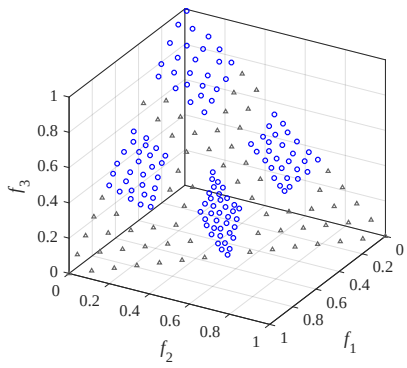


Fig. 3. Combining the nodes obtained by GNG in Fig. 2 and a set of uniformly distributed reference vectors. The triangles and the circles represents the reference vectors from R'_u (i.e., uniformly generated reference vectors) and R_p (i.e., nodes mapped on a hyperplane), respectively.

Algorithm 4 Combination

Require: R'_{node} (Transformed Nodes in GNG), R_u (Uniformly Distributed Reference Vectors)

1: $R_p = \emptyset$;
2: **for all** $\boldsymbol{r}'_{\text{node}} \in R'_{\text{node}}$ **do**
3: $\quad \boldsymbol{r}_p = \boldsymbol{r}'_{\text{node}} / \sum_{m=1}^{M} r'_{\text{node,m}}$; ▷ map the nodes in R'_{node} on a hyperplane specified by $f_1 + \cdots + f_M = 1$.
4: $\quad R_p = R_p \cup \{\boldsymbol{r}_p\}$;
5: **end for**
6: calculate the average Euclidean distance between every pair of connected nodes in R_p, denoted as d_p;
7: find the minimum Euclidean distance between each pair of the reference vectors in R_u, denoted as d_u;
8: $d_{\min} = \min(d_p, d_u)$;
9: $R'_u = R_u$;
10: **for all** $\boldsymbol{r}'_u \in R'_u$ **do**
11: $\quad$ **if** $d(\boldsymbol{r}'_u, R_p) < d_{\min}$ **then** ▷ $d(\boldsymbol{r}'_u, R_p)$ is the shortest Euclidean distance between $\boldsymbol{r}'_u$ and the nodes in R_p.
12: $\quad\quad R'_u = R'_u / \{\boldsymbol{r}'_u\}$;
13: $\quad$ **end if**
14: **end for**
15: **return** $R = R'_u \cup R'_{\text{node}}$

reference vectors in R_u (line 7). The threshold of removing reference vectors in R'_u (initialized to be equal to R_u) is the smaller value between d_p and d_u. If the Euclidean distance between a reference vector in R'_u and any node in R_p is smaller than the threshold, it is removed (lines 10 to 14). Finally, $R = R'_u \cup R'_{\text{node}}$ is returned as the adapted reference vectors for selection (line 15).

If d_p is much smaller than d_u, it implies that the most nodes in R'_{node} are crowded in some small regions. Then we use d_p as the threshold, so that the the reference vectors in R'_u which are very close to R'_{node} can be reserved, and the population could have a better chance to spread by the guidance of the reference vectors. On the other hand, a greater d_p value than d_u indicates that the number of nodes may be not enough (i.e., smaller than the population size). We adopt d_u as the threshold to improve the uniformity in the entire objective space. We show an example in Fig. 3 of combining the nodes in Fig. 2 with a set of uniformly distributed reference vectors, where the triangles and the circles are the reference vectors from R'_u and R_p, respectively.

Using the above combination, even if the nodes in R'_{node} are poorly distributed in the early stage, the reference vectors in R'_u will guide the population to search efficiently. One should note that if we set both the sizes of R_{node} and R_u equal or similar to the population size N, the size of $R'_{\text{node}} \cup R'_u$ will be greater than N in most cases. It suggests that in the environmental selection no solution will be assigned to some reference vectors in $R'_{\text{node}} \cup R'_u$, while every reference vector must be assigned to a solution in some decomposition-based MOEAs, such as MOEA/D. However, this creates no difficulty since the environmental selection in the proposed method is different from that in MOEA/D. If no Pareto optimal solution is close to a reference vector, then no solution will be assigned to the reference vector at the end of the evolution due to the association in Algorithm 2.

F. Scalarizing Function Adaptation

The importance of choosing a proper scalarizing function for each reference vector has been discussed in [31], [32]. A scalarizing function whose contour lines fit the PF shape can enhance the algorithm's search ability [52], [53]. Popular scalarizing functions includes PBI [28] and L_p [31], [54]. Both PBI and L_p have a parameter to generate different scalarizing functions. The parameter is $\theta \in [0, \infty]$ in PBI and $p \in [1, \infty]$ in L_p. Different settings of θ or p result in different preferences in selecting solutions. In general, a larger θ or p value prefers better diversity while a smaller one is for better convergence. The contour lines when $\theta \in [0, 1]$ in PBI are similar to those when $p \in [1, \infty]$ in L_p. Compared to L_p, PBI can further address diversity performance by setting $\theta \in (1, \infty]$.

Therefore, in this study, we propose to tune θ in PBI to specify an appropriate scalarizing function for each reference vector obtained from GNG, i.e., R'_{node}. The PBI scalarizing function is formulated as

$$f^S = d_1 + \theta \cdot d_2, \tag{6}$$

where θ is the aforementioned parameter. Let $\boldsymbol{y}$ be the projection of $\boldsymbol{f}(\boldsymbol{x})$ on the reference vector $\boldsymbol{r}$. d_1 is the

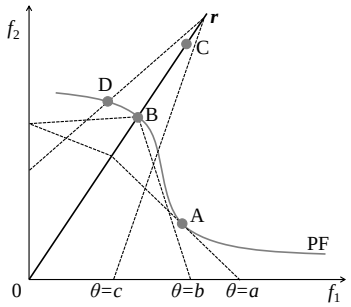


Fig. 4. Different settings of θ in PBI. $\boldsymbol{r}$ is the reference vector. The dash lines are the contour lines formed by PBI when θ is set to a, b, c, where $a < b < c$. A, B, and D are Pareto optimal solutions, and C is a solution far away from the PF.

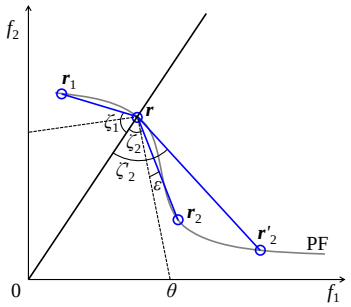


Fig. 5. Estimating θ in PBI by Eqs. (7) and (8). $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the topological neighbors of $\boldsymbol{r}$ in the GNG network. ζ_1 (ζ_2) is the angle between $\boldsymbol{r}$ and $\boldsymbol{r}_1$ ($\boldsymbol{r}_2$). $\boldsymbol{r}_2$ could be $\boldsymbol{r}'_2$ when the number of nodes in GNG is not enough.

distance between $\boldsymbol{z}^*$ (i.e., the ideal point which is estimated in Algorithm 1) and $\boldsymbol{y}$, and d_2 is the distance between $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{y}$.

Let us see an example of different settings of θ in Fig. 4, where $\boldsymbol{r}$ is the reference vector, the dash lines are the contour lines formed by PBI when θ is set to a, b, c, where $a < b < c$. A, B, and D are Pareto optimal solutions, and C is a solution far away from the PF. When $\theta = a$, A is most likely to be obtained since it has the best scalarizing function value. Actually, it is far from $\boldsymbol{r}$, which implies a poor diversity performance. If we set θ to c, C is evaluated as being better than D, which will reduce the algorithm's ability in converging. For the reference vector $\boldsymbol{r}$, $\theta = b$ is the best setting among them to search for B which locates at the intersection of $\boldsymbol{r}$ and the PF. We can see from this example that the appropriate setting of θ depends on the shape of the PF. The guidance of setting θ is to promote convergence under the guarantee of diversity.

Taking advantage of the topological information from GNG, we propose to estimate a proper θ value for each reference vector (node) $\boldsymbol{r}$ by the following equation:

$$\theta = \max(0, 1/\tan(\zeta')), \tag{7}$$

where

$$\begin{aligned} \zeta' &= \max(0, \zeta_{\min} - \epsilon), \\ \zeta_{\min} &= \min \zeta_k, k = 1, \ldots, K. \end{aligned} \tag{8}$$

In Eq. (8), ϵ is a small positive value, and ζ_k ($k = 1, \ldots, K$) is the angle between $\boldsymbol{r}$ and the kth edge $\boldsymbol{v}_{\text{edge},k}$ emanated from it. $\boldsymbol{v}_{\text{edge},k}$ can be obtained by $\boldsymbol{r}_k^{nb} - \boldsymbol{r}$, where $\boldsymbol{r}_k^{nb}$ is the kth topological neighbor of $\boldsymbol{r}$.

The reason of subtracting ϵ from $\zeta_{\min}$ is that we expect to slightly increase θ to counteract the distortion error and to guarantee diversity. The distortion error comes from the inconsistency between the GNG network and the true PF, since the GNG network cannot exactly reflect the true PF shape with a limited number of input signals and nodes in practice. ϵ can be set to a relatively large value when the number of nodes is not enough, especially when solving MaOPs. Please refer to the supplementary material for the investigations on setting ϵ.

Fig. 5 shows an example, where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the topological neighbors of $\boldsymbol{r}$. We can estimate θ by Eqs. (7) and (8) with the angles, ζ_1 and ζ_2. The angle between $\boldsymbol{r}$ and the contour lines formed by the scalarizing function is a bit smaller than ζ_1 and ζ_2 due to ϵ. However, one can imagine that we may have a neighbor node $\boldsymbol{r}'_2$ instead of $\boldsymbol{r}_2$. Then, we could obtain an improper θ value if ϵ is not large enough. An intuitive way to solve this issue is to increase the number of nodes. Consequently, the distances between nodes will decrease, and θ will be estimated more accurately.

For an isolated node in R'_{node}, θ is set to ∞ to guarantee diversity. In other words, the scalarizing function is d_2. Any other setting of θ will increase the chance of selecting solution far away from the isolated node. Then we may lose this isolated node and the solutions close to it in next generations. In addition, the scalarizing function is also set to d_2 for each reference vector in R'_u. There is no solution associated to each reference vector in R'_u in the previous generation. Using such a setting of θ can select solutions closest to the reference vectors in R'_u and thus enhance the algorithm's exploration ability in the entire objective space. Note that it is technically possible to tune p in L_p by the same manner of tuning θ by GNG, which is an interesting future work.

G. Discussions

To handle irregular PFs, most existing decomposition-based MOEAs adapt either the reference vectors or the scalarizing functions. Different from these methods, DEA-GNG simultaneously adapts the reference vectors and the scalarizing functions. Among these methods, MOEA/D-SOM [41] is the most related method to DEA-GNG, where both GNG and SOM can learn the topological structure of the PF using a network which consists of nodes and edges. In addition, MOEA/D-LTD [33] is another well-crafted method to adapt both the reference vectors and the scalarizing functions. In the following, we discuss the differences between DEA-GNG and MOEA/D-SOM and those between DEA-GNG and MOEA/D-LTD, respectively.

1) DEA-GNG vs. MOEA/D-SOM: In addition to the differences between GNG and SOM aforementioned in Subsection II-B, DEA-GNG and MOEA/D-SOM have the following four main differences:

(1) The input signals in MOEA/D-SOM are obtained by simply combining solutions in the last five generations. This may decrease the diversity of the population. DEA-GNG uses an archive to maintain good input signals.
(2) MOEA/D-SOM only uses the nodes as the reference vectors, while DEA-GNG combines the nodes with a set of uniformly distributed reference vectors as the reference vectors.
(3) MOEA/D-SOM does not adapt the scalarizing functions. One reason is that the edges in SOM are unable to correctly provide the curvature information of the PF.
(4) In MOEA/D-SOM, the SOM network requires 100 iterations for initialization and 10 iterations for update, which consumes a lot of computational resources. In DEA-GNG, the GNG network only requires one iteration for both initialization and update.

2) DEA-GNG vs. MOEA/D-LTD: The differences between DEA-GNG and MOEA/D-LTD fall into the following four aspects:

(1) MOEA/D-LTD uses a GP-based model to learn the mapping from $M-1$ objective functions to the remaining objective function (M is the number of objectives). A continuous hypersurface to approximate the PF can be obtained based on the model. However, DEA-GNG uses a GNG network to learn the topological structure of the PF.
(2) To obtain the reference vectors, a large number of points are sampled on the hypersurface in MOEA/D-LTD. Only a small number of points are selected from the sampled points based on their prediction variances of the GP model, Pareto dominance relations, and density values. In DEA-GNG, the reference vectors are obtained by expanding the nodes in the GNG network and combining them with a set of uniformly distributed reference vectors.
(3) In MOEA/D-LTD, the disconnection of the PF is not considered when adapting the scalarizing functions. In DEA-GNG, there is no edge connecting sub-networks, where each sub-network represents a disconnected region of PF. Thus the edge-based scalarizing function adaptation in a sub-network is not affected by the other sub-networks.
(4) According to [33], the time complexity of the GP-based learning model in one generation is $O(N^3)$ (N is the population size). Thus the total time complexity of MOEA/D-LTD in one generation is at least $O(N^3)$. In DEA-GNG, the time complexities of the GNG-based learning model [55] and the DEA-based optimization model in one generation are both $O(N^2)$. Thus the total time complexity of DEA-GNG in one generation is $O(N^2)$. This implies that DEA-GNG is computationally cheaper than MOEA/D-LTD. Please refer to the supplementary material for the complexity analysis of DEA-GNG.

TABLE I
THE PF PROPERTIES OF TEST PROBLEMS

Reference	Problems	Geometric Shape	Badly Scaled	Incomplete	Disconnected	Degenerate	High-Dimensional
[2]	Scaled DTLZ2-2	Concave	Yes				
	Scaled DTLZ2-3	Concave	Yes				
	Scaled DTLZ2-5	Concave	Yes				Yes
	Scaled DTLZ2-8	Concave	Yes				Yes
[2]	Convex DTLZ2-2	Convex					
	Convex DTLZ2-3	Convex					
	Convex DTLZ2-5	Convex					Yes
	Convex DTLZ2-8	Convex					Yes
[1]	Minus-DTLZ2-2	Convex		Yes			
	Minus-DTLZ2-3	Convex		Yes			
	Minus-DTLZ2-5	Convex		Yes			Yes
	Minus-DTLZ2-8	Convex		Yes			Yes
[56]	DTLZ2BZ-2	Concave		Yes			
	DTLZ2BZ-3	Concave		Yes			
	DTLZ2BZ-5	Concave		Yes			Yes
	DTLZ2BZ-8	Concave		Yes			Yes
[10]	DTLZ5-3	Concave		Yes		Yes	
	DTLZ5-5	Concave		Yes		Yes	
	DTLZ5-8	Concave		Yes		Yes	
	DTLZ7-2	Mixed	Yes	Yes	Yes		
	DTLZ7-3	Mixed	Yes	Yes	Yes		
	DTLZ7-5	Mixed	Yes	Yes	Yes		Yes
	DTLZ7-8	Mixed	Yes	Yes	Yes		Yes
[57]	WFG1-2	Mixed	Yes				
	WFG1-3	Mixed	Yes				
	WFG1-5	Mixed	Yes				Yes
	WFG1-8	Mixed	Yes				Yes
	WFG2-2	Mixed	Yes	Yes	Yes		
	WFG2-3	Mixed	Yes	Yes	Yes		
	WFG2-5	Mixed	Yes	Yes	Yes		Yes
	WFG2-8	Mixed	Yes	Yes	Yes		Yes
[58] [59]	Polygon-based Problem-3	Convex		Yes			
	Polygon-based Problem-5	Convex		Yes			Yes
	Polygon-based Problem-8	Convex		Yes			Yes

IV. EXPERIMENTS

A. Settings of Experiments

In this study, we choose 34 test problems with irregular PFs [1], [2], [10], [56]–[59]. Their references and PF properties are listed in Table I. We consider these test problems with 2, 3, 5, and 8 objectives except for DTLZ5 and polygon-based problems. DTLZ5 and polygon-based problems do not have the bi-objective versions. Therefore, we consider DTLZ5 and polygon-based problems with 3, 5, and 8 objectives. Note that Problem$X$-M means Problem$X$ with M objectives. For DTLZ7 and polygon-based problems, the numbers of decision variables are M+19 and 2, respectively. For the other test problems, the number of decision variables is M+9.

We choose eight state-of-the-art decomposition-based MOEAs as the comparative algorithms in this study. They are A-NSGA-III [35], RVEA* [3], AdaW [37], MOEA/D-LTD [33], MOEA/D-AWA [34], MOEA/D-SOM [41], MOEA/D-PaS [31], and MOEA/D [28], which have been introduced in Section II. The code of MOEA/D-LTD is provided by the original authors. All the other algorithms are implemented by PlatEMO [60]. In addition, we include a variant of DEA-GNG which does not have the adaptation mechanism of the scalarizing function, denoted as DEA-GNG*. In DEA-GNG*, the scalarizing function for every reference vector is d_2. We compare DEA-GNG* with its original version to verify the effectiveness of the scalarizing function adaptation.

The following parameter settings are adopted by all the comparative algorithms. Simulated binary crossover (except in MOEA/D-PaS, where differential evolution operator with $F = 0.5$ and $CR = 0.9$ is adopted according to the

original study [31]) and polynomial mutation are applied as the crossover and mutation operators, respectively, where both distribution indexes are set to 20. The crossover and mutation probabilities are 1.0 and $1/n$, respectively, where n is the number of decision variables. The termination criterion is that the population has evolved for the predefined maximum number of generations. For WFG1, the maximum number of generations is set to 1,000, while 300 is used for the other test problems. For 2-, 3-, and 5-objective test problems, the systematic approach [61] is used to generate a set of uniformly distributed reference vectors. For 8-objective test problems, the two-layers approach [2] is used. The number of reference vectors generated by these approaches is non-arbitrary. We set the number of reference vectors to be 100 ($M = 2$), 120 ($M = 3$), 126 ($M = 5$), and 156 ($M = 8$) for all the comparative algorithms except for MOEA/D-SOM. Then, the population size N in these algorithms is set equal to the number of reference vectors. In MOEA/D-SOM, we use a 2-D SOM network where the number of nodes in each dimension is the same as suggested in [41]. N in MOEA/D-SOM is set equal to the number of nodes, which is also non-arbitrary. For a fair comparison, N is set to be 100 ($M = 2$), 121 ($M = 3$), 144 ($M = 5$), and 169 ($M = 8$) which are similar to those in the other algorithms.

In DEA-GNG, the size of input signal archive is set to MN. ϵ is set to 0.05π for 2- and 3-objective test problems and 0.15π for 5- and 8-objective test problems. For GNG, $\mathrm{HP}_{\max}$, $\mathrm{age}_{\max}$, λ, ϵ_a, ϵ_{nb}, α, and δ are set to $2|A_S|$, N, $0.2N$, 0.2, 0.01, 0.5, and 0.9, respectively. These parameters in GNG are set empirically. Although they can be further tuned for each test problem, we use a fixed setting for all the test problems to highlight the robustness and generality of our method for solving various MOPs.

The other parameters of the comparative algorithms are set in the same manner as in the original studies. In MOEA/D and MOEA/D-SOM, the PBI approach with $\theta = 5$ and the Tchebycheff approach are adopted, respectively. In RVEA*, the rate of changing the penalty function and the frequency to conduct the reference vector adaptation are set to 2 and 0.1, respectively. In MOEA/D-AWA, the ratio of updated weight vectors, the ratio of iterations to evolve with only MOEA/D, and the iteration interval of utilizing AWA are set to 0.05, 0.8, and 100, respectively. In AdaW, the interval of updating the reference vectors and the time of not allowing the update are every 5% of the total generations and the last 10% generations, respectively. The maximum archive size is set to $2N$. In MOEA/D-LTD, the begin, end, and interval of performing LTD procedure is 50%, 80%, and 20, respectively. The rate of the number of samples to the population size is 40. The coefficient of the variance threshold is 1.4.

In order to compare different algorithms on these test problems, we adopt the Inverted Generational Distance Plus (IGD^+) [62] as a performance metric which gives a comprehensive quantification of both the convergence and diversity of a solution set. It is weakly Pareto compliant and thus more accurate on evaluation than its original version, IGD. The smaller the value of IGD^+ of an algorithm, the better the performance of the algorithm. IGD^+ requires a reference set, whose points are typically uniformly distributed on the true PF of a test problem. In this study, over 5,000 reference points are evenly sampled from the true PF of each test problem. Note that solutions and the reference points will be normalized based on the true PF, when calculating IGD^+. In addition, we also use the HV [25] indicator to compare these algorithms. Please refer to the supplementary material for the results of HV. Note that there are some inconsistencies of evaluation results between IGD^+ and HV. One reason is that the evaluation by HV depends on the setting of the reference point when the PF shape is irregular [63]. Another reason is that HV and IGD^+ have different preferences for evaluation. In other words, different decision makers may have different preference to the obtained solution sets, which is quite natural. Please refer to [64] for a detailed discussion.

All the experimental results in this paper are obtained by executing 30 independent runs of each algorithm on each optimization problem. The Wilcoxon's rank sum test is employed to determine whether one algorithm shows a statistically significant difference with the other, and the null hypothesis is rejected at a significant level of 0.05.

B. Comparison with State of the Art

In this subsection, we applied DEA-GNG, DEA-GNG*, A-NSGA-III, RVEA*, AdaW, MOEA/D-LTD, MOEA/D-AWA, MOEA/D-SOM, MOEA/D-PaS, and MOEA/D to 34 test problems listed in Table I. The average IGD^+ value and the corresponding performance score [25] of each algorithm on each test problem are given in Table II. A darker tone corresponds to a larger performance score. For each test problem, the performance score of an algorithm is the number of the comparative algorithms which perform significantly worse than it according to IGD^+. Moreover, we give the average performance score of each algorithms over all the test problems at the bottom of Table II.

We can see from Table II that DEA-GNG achieved the highest average performance score, followed by AdaW, DEA-GNG*, and the others. This means that DEA-GNG has the best general performance on these test problems, and DEA-GNG* is the third best. DEA-GNG outperformed DEA-GNG* on most test problems, which demonstrates the effectiveness of the proposed scalarizing function adaptation. However, DEA-GNG obtained worse results on Convex DTLZ2 with more than two objectives than DEA-GNG*. The reason is that an extremely large value of θ is needed to find solutions around the boundary of the PFs of these test problems (due to the shape of the PFs). It is difficult for the proposed adaptation method to find such an extreme value based on the limited number of nodes. Both DEA-GNG and DEA-GNG* failed on 8-objective Convex DTLZ2, while MOEA/D is the best. This implies that the proposed reference vector adaptation could have a negative effect for this problem.

DEA-GNG (and DEA-GNG*) did not work well on 5- and 8-objective DTLZ5. This is owing to a large number of non-dominated solutions far away from the PF, which misleads the structure of GNG when solving this problem. Both DEA-GNG and DEA-GNG* did not work well on 2-objective WFG1,

TABLE II
RESULTS OF IGD^+

IGD^+	DEA-GNG	DEA-GNG*	A-NSGA-III	RVEA*	AdaW	MOEA/D-LTD	MOEA/D-AWA	MOEA/D-SOM	MOEA/D-PaS	MOEA/D
Scaled DTLZ2-2	1.852E-3 8	1.910E-3 6	2.717E-3 1	1.964E-3 4	1.862E-3 8	1.886E-3 7	1.985E-3 4	7.569E-3 0	2.204E-3 2	2.160E-3 2
Scaled DTLZ2-3	2.198E-2 6	2.321E-2 5	2.498E-2 2	2.508E-2 1	2.022E-2 8	1.945E-2 9	2.165E-2 7	5.167E-2 0	2.549E-2 1	2.494E-2 2
Scaled DTLZ2-5	1.033E-1 4	1.072E-1 3	7.370E-2 8	1.112E-1 3	8.374E-2 7	7.180E-2 9	1.131E-1 2	1.842E-1 1	1.136E-1 2	3.384E-1 0
Scaled DTLZ2-8	2.057E-1 4	2.132E-1 4	1.666E-1 8	3.048E-1 2	1.882E-1 6	1.483E-1 8	1.908E-1 6	2.698E-1 3	4.329E-1 1	5.665E-1 0
Convex DTLZ2-2	2.554E-3 6	2.606E-3 6	3.860E-3 1	3.503E-3 1	2.409E-3 8	2.804E-3 5	2.455E-3 8	5.827E-3 0	3.454E-3 2	3.024E-3 4
Convex DTLZ2-3	1.734E-2 6	1.655E-2 7	2.156E-2 1	2.337E-2 0	1.493E-2 8	2.238E-2 1	1.886E-2 4	2.026E-2 3	1.827E-2 5	1.512E-2 8
Convex DTLZ2-5	2.988E-2 7	2.919E-2 7	3.914E-2 5	3.717E-2 5	4.828E-2 4	5.424E-2 2	5.607E-2 1	5.472E-2 1	6.323E-2 0	2.276E-2 9
Convex DTLZ2-8	2.614E-2 3	2.570E-2 3	4.575E-2 2	1.982E-2 8	8.485E-2 1	2.141E-2 5	2.158E-2 5	2.118E-2 5	1.228E-1 0	1.884E-2 9
Minus-DTLZ2-2	1.908E-3 7	1.908E-3 7	4.553E-3 2	2.379E-3 5	1.898E-3 7	3.007E-3 4	2.067E-3 6	2.579E-2 0	3.301E-3 3	5.216E-3 1
Minus-DTLZ2-3	2.510E-2 7	2.842E-2 6	3.625E-2 2	2.562E-2 7	2.482E-2 8	4.192E-2 1	3.151E-2 4	5.934E-2 0	3.343E-2 3	3.095E-2 4
Minus-DTLZ2-5	1.055E-1 9	1.089E-1 7	1.569E-1 4	1.187E-1 6	1.081E-1 7	2.365E-1 1	1.909E-1 3	3.111E-1 0	2.337E-1 2	1.331E-1 5
Minus-DTLZ2-8	2.147E-1 9	2.294E-1 6	2.803E-1 4	2.332E-1 6	2.227E-1 6	3.116E-1 2	3.413E-1 0	2.480E-1 5	2.986E-1 3	3.324E-1 0
DTLZ2BZ-2	2.584E-3 5	2.905E-3 3	3.854E-3 1	2.494E-3 6	2.503E-3 6	2.448E-3 9	2.474E-3 8	8.798E-3 0	2.913E-3 3	3.687E-3 1
DTLZ2BZ-3	2.538E-2 9	2.891E-2 5	2.934E-2 4	2.647E-2 7	2.714E-2 6	3.204E-2 3	2.802E-2 6	7.136E-2 0	4.469E-2 1	4.173E-2 2
DTLZ2BZ-5	9.900E-2 9	1.252E-1 6	1.291E-1 5	1.004E-1 8	1.229E-1 6	1.494E-1 2	1.514E-1 2	2.122E-1 1	2.426E-1 0	1.327E-1 4
DTLZ2BZ-8	2.019E-1 7	2.046E-1 7	2.508E-1 3	2.149E-1 6	1.987E-1 7	2.840E-1 1	2.245E-1 5	2.811E-1 1	7.941E-1 0	2.597E-1 3
DTLZ5-3	1.800E-3 9	1.900E-3 7	5.186E-3 4	2.583E-3 6	1.853E-3 7	8.700E-3 2	4.437E-3 5	7.977E-3 3	9.392E-3 1	1.523E-2 0
DTLZ5-5	1.002E-1 3	2.353E-1 0	2.655E-1 0	3.852E-2 7	1.233E-1 2	1.043E-1 3	6.825E-2 6	6.390E-3 9	1.152E-1 2	1.270E-2 8
DTLZ5-8	1.765E-1 4	4.863E-1 0	5.771E-1 0	8.516E-2 6	2.010E-1 3	1.497E-1 4	6.226E-2 7	8.403E-3 9	4.399E-1 1	1.708E-2 8
DTLZ7-2	1.933E-3 7	2.038E-3 7	1.848E-3 9	1.253E-2 4	2.161E-3 6	2.334E-2 3	2.358E-3 5	1.575E-1 1	3.143E-1 0	8.928E-2 1
DTLZ7-3	1.936E-2 8	2.311E-2 6	2.827E-2 5	2.266E-2 7	1.973E-2 7	2.909E-2 4	4.096E-2 3	1.140E-1 0	2.283E-1 0	4.841E-2 2
DTLZ7-5	8.756E-2 9	1.118E-1 6	1.300E-1 4	9.910E-2 8	1.165E-1 5	1.041E-1 7	1.775E-1 3	2.240E-1 0	3.369E-1 0	3.496E-1 0
DTLZ7-8	1.851E-1 8	2.032E-1 5	1.996E-1 5	2.627E-1 3	1.994E-1 5	2.189E-1 4	2.061E-1 5	4.292E-1 1	3.058E-1 2	1.042E+0 0
WFG1-2	2.052E-2 3	2.087E-2 3	1.121E-2 7	4.281E-2 1	1.634E-2 3	2.855E-2 2	5.248E-3 9	1.126E-2 7	1.552E-1 0	1.378E-2 5
WFG1-3	1.889E-2 7	1.929E-2 6	1.943E-2 6	2.563E-2 3	1.959E-2 6	2.540E-2 4	2.818E-2 2	8.361E-2 1	1.435E-1 0	2.639E-2 3
WFG1-5	3.448E-2 9	7.384E-2 4	5.032E-2 6	9.635E-2 1	4.663E-2 8	5.734E-2 5	9.533E-2 1	1.143E-1 0	1.165E-1 0	5.092E-2 6
WFG1-8	3.507E-2 8	6.658E-2 3	5.058E-2 4	1.204E-1 1	5.322E-2 4	3.715E-2 8	4.432E-2 7	6.192E-2 3	2.888E-1 0	9.984E-2 2
WFG2-2	2.000E-3 9	2.889E-3 4	2.517E-3 4	2.918E-3 3	2.279E-3 7	2.727E-3 4	2.682E-3 4	6.611E-3 1	4.263E-3 2	1.346E-2 0
WFG2-3	1.451E-2 9	1.801E-2 6	2.160E-2 5	3.032E-2 1	1.734E-2 6	2.526E-2 3	2.740E-2 2	2.947E-2 1	1.696E-2 7	6.055E-2 0
WFG2-5	3.454E-2 8	4.070E-2 7	5.161E-2 5	7.251E-2 1	3.289E-2 8	5.936E-2 3	6.369E-2 2	5.308E-2 5	7.848E-2 1	1.283E-1 0
WFG2-8	3.491E-2 8	4.096E-2 5	9.518E-2 1	7.352E-2 3	5.900E-2 4	2.645E-2 9	4.255E-2 5	4.444E-2 5	3.836E-1 0	1.031E-1 1
Polygon-based Problem-3	2.229E-2 8	2.370E-2 6	2.666E-2 5	2.230E-2 8	2.331E-2 6	3.951E-2 3	3.107E-2 4	4.419E-2 1	5.164E-2 0	4.331E-2 1
Polygon-based Problem-5	3.383E-2 9	3.476E-2 7	4.553E-2 6	7.714E-2 3	3.849E-2 6	8.130E-2 2	6.739E-2 3	8.646E-2 1	1.169E-1 0	5.586E-2 5
Polygon-based Problem-8	3.934E-2 9	4.018E-2 7	5.105E-2 6	1.154E-1 1	4.233E-2 6	1.110E-1 3	1.175E-1 1	1.037E-1 4	1.017E-1 4	1.578E-1 0
Average Performance Score	7.088	5.206	3.971	4.176	5.941	4.176	4.265	2.118	1.412	2.824

either. We observed that when solving WFG1, the obtained PF usually spread from a very small area to the whole PF, which resulted in most input signals as well as the nodes located in the starting area. However, this feature of WFG1 seems have a larger negative effect on the other algorithms than DEA-GNG when the number of objectives is 3, 5, and 8.

A-NSGA-III achieved acceptable comprehensive performance according its average performance score. It got exciting results on Scaled DTLZ2-5, Scaled DTLZ2-8, and DTLZ7-2. However, it performed poorly on 2- and 3-objective Scaled DTLZ2, Convex DTLZ2, Minus DTLZ2, DTLZ2BZ, 5- and 8-objective Scaled DTLZ5, and 8-objective WFG2. Since A-NSGA-III regularly generates new reference vectors around uniformly distributed ones, these reference vectors cannot perfectly fit the irregular PF shape in such cases.

The general performance of RVEA* is moderate. It randomly generate new reference vectors where a Pareto optimal solution may locate, which has a positive effect on the test problems whose PFs do not cover the entire objective space, such as Minus-DTLZ2, DTLZ2BZ, DTLZ5, and 3-objective Polygon-based Problem.

AdaW won the second place according to the average performance score and generally performed better than DEA-GNG*. This shows the strength of the archive maintenance method in AdaW in promoting diversity. It is very interesting to use this archive maintenance method to maintain input signals in DEA-GNG in the future. However, this archive maintenance method seems to result in relatively poor performance on some high-dimensional problems, such as Convex DTLZ2-5, Convex DTLZ2-8, DTLZ5-5, DTLZ5-8, WFG1-8, and WFG2-8.

MOEA/D-LTD is generally better than A-NSGA-III. It is powerful in solving Scaled DTLZ2. This indicates that the GP-based learning model is quite good at handling the PF with a relatively simple shape. MOEA/D-LTD also achieved outstanding results on DTLZ2BZ-2, WFG1-8, and WFG2-8.

MOEA/D-AWA performed the fourth best among the examined algorithms on the average. It worked well on most problems especially Scaled DTLZ2-3, Scaled DTLZ2-8, Convex DTLZ2-2, DTLZ2BZ-2, WFG1-2, and WFG1-8. However, it seems that MOEA/D-AWA cannot handle an inverted triangle PF in the high-dimensional space according to the results on 5- and 8-objective Minus-DTLZ2 and Polygon-based Problems.

MOEA/D-SOM performed poorly on most test problems due to two aspects. One is that MOEA/D-SOM lacks a strategy to reserve the good input signals. The other is the drawbacks of SOM as aforementioned in Subsection II-B. However, MOEA/D-SOM significantly outperformed the others on 5- and 8-objective DTLZ5. This is because the 2-D SOM network has a natural advantage in learning a low-dimensional PF in

the high-dimensional space.

MOEA/D-PaS achieved the worst performance among these algorithms mainly due to the lack of reference vector adaptation, which implies that reference vector adaptation may be generally more important than scalarizing function adaptation for handling irregular PFs. However, MOEA/D-PaS still obtained some satisfying results, such as on 3-objective Convex DTLZ2 and WFG2.

MOEA/D is a classic decomposition-based algorithm with no adaptation mechanism of reference vectors or scalarizing functions. Although its general performance is not good, it surprisingly outperformed most other algorithms on Convex DTLZ2 with 3, 5, and 8 objectives and DTLZ5 with 5 and 8 objectives. The reason is that the misleading information obtained during the evolution results in inappropriate adaptation in the other algorithms.

V. Conclusion

In this paper, we have proposed a novel decomposition-based multi-objective evolutionary algorithm guided by growing neural gas, termed DEA-GNG. In the proposed method, the topological structure of the PF is learned by a modified GNG model where the known solutions are contentiously input as signals. The reference vectors are adapted by combining the nodes in the GNG network and a set of uniformly distributed reference vectors. The scalarizing function of each reference vector is adapted based on the angles between the corresponding node and the edges emanated from it. The search ability of the evolutionary algorithm is enhanced by these two adaptation mechanisms.

We compared DEA-GNG with A-NSGA-III, RVEA*, AdaW, MOEA/D-LTD, MOEA/D-AWA, MOEA/D-SOM, MOEA/D-PaS, and MOEA/D by applying them to 34 test problems. The experimental results showed that DEA-GNG is better than the compared algorithms on average. Moreover, by comparing DEA-GNG with its variant DEA-GNG* without adapting θ, the effectiveness of the proposed scalarizing function adaptation was verified. In addition, please refer to the supplementary material for the investigation of the effects of the three parameters, N_S, ϵ, and α on the behavior of DEA-GNG.

One future direction is to improve the input signal archive update for uniformity. One possible way is to incorporate some existing diversity maintenance methods. Another future research is to design a new GNG model that can provide accurate information of adapting scalarizing functions even if the number of nodes is not enough.

* The code of DEA-GNG is available on https://github.com/yiping0liu.

References

[1] H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, "Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 169–190, 2017.

[2] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2013.

[3] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, 2016.

[4] Y. Xiang, Y. Zhou, M. Li, and Z. Chen, "A vector angle-based evolutionary algorithm for unconstrained many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 131–152, 2017.

[5] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 204–223, 2003.

[6] B. Derbel, A. Liefooghe, Q. Zhang, H. Aguirre, and K. Tanaka, "Multi-objective local search based on decomposition," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 431–441.

[7] Y. Liu, G. G. Yen, and D. Gong, "A multi-modal multi-objective evolutionary algorithm using two-archive and recombination strategies," *IEEE Transactions on Evolutionary Computation*, 2018. [Online]. Available: http://dx.doi.org/10.1109/TEVC.2018.2879406

[8] R. Tanabe and H. Ishibuchi, "A decomposition-based evolutionary algorithm for multi-modal multi-objective optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2018, pp. 249–261.

[9] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.

[10] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, *Scalable Test Problems for Evolutionary Multiobjective Optimization*. Springer, 2005.

[11] Y. Hua, Y. Jin, and K. Hao, "A clustering-based adaptive evolutionary algorithm for multiobjective optimization with irregular Pareto fronts," *IEEE Transactions on Cybernetics*, vol. 49, no. 7, pp. 2758–2770, 2019.

[12] R. Qi and G. G. Yen, "Hybrid bi-objective portfolio optimization with pre-selection strategy," *Information Sciences*, vol. 417, pp. 401–419, 2017.

[13] Y. Han, J.-Q. Li, D. Gong, and H. Sang, "Multi-objective migrating birds optimization algorithm for stochastic lot-streaming flow shop scheduling with blocking," *IEEE Access*, vol. 7, pp. 5946–5962, 2019.

[14] J. Z. Salazar, P. M. Reed, J. D. Quinn, M. Giuliani, and A. Castelletti, "Balancing exploration, uncertainty and computational demands in many objective reservoir optimization," *Advances in Water Resources*, vol. 109, pp. 196–210, 2017.

[15] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in neural information processing systems*, 1995, pp. 625–632.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[17] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), Tech. Rep., 2001.

[18] H. Sato, H. Aguirre, and K. Tanaka, "Improved S-CDAs using crossover controlling the number of crossed genes for many-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Deblin, Ireland: ACM, 2011, pp. 753–760.

[19] A. L. Jaimes, H. Aguirre, K. Tanaka, and C. A. C. Coello, "Objective space partitioning using conflict information for many-objective optimization," in *International Conference on Parallel Problem Solving from Nature*. Krakw, Poland: Springer, 2010, pp. 657–666.

[20] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[21] F. di Pierro, S.-T. Khu, and D. A. Savic, "An investigation on preference order ranking scheme for multiobjective evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 1, pp. 17–45, 2007.

[22] Z. He, G. G. Yen, and J. Zhang, "Fuzzy-based Pareto optimality for many-objective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 269–285, 2014.

[23] A. López Jaimes and C. A. Coello Coello, "Study of preference relations in many-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Montreal, Canada: ACM, 2009, pp. 611–618.

[24] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *International Conference on Parallel Problem Solving from Nature*. Birmingham, UK: Springer, 2004, pp. 832–842.

[25] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011.

[26] R. Hernández Gómez and C. A. Coello Coello, "Improved metaheuristic based on the R2 indicator for many-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Madrid, Spain: ACM, 2015, pp. 679–686.

[27] K. Shang, H. Ishibuchi, M.-L. Zhang, and Y. Liu, "A new R2 indicator for better hypervolume approximation," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2018, pp. 745–752.

[28] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[29] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Adaptation of scalarizing functions in MOEA/D: An adaptive scalarizing function-based multiobjective evolutionary algorithm," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2009, pp. 438–452.

[30] ——, "Simultaneous use of different scalarizing functions in MOEA/D," in *Proceedings of the Conference on Genetic and Evolutionary Computation*. ACM, 2010, pp. 519–526.

[31] R. Wang, Q. Zhang, and T. Zhang, "Decomposition-based algorithms using pareto adaptive scalarizing methods," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 821–837, 2016.

[32] M. Ming, R. Wang, Y. Zha, and T. Zhang, "Pareto adaptive penalty-based boundary intersection method for multi-objective optimization," *Information Sciences*, vol. 414, pp. 158–174, 2017.

[33] M. Wu, K. Li, S. Kwong, Q. Zhang, and J. Zhang, "Learning to decompose: a paradigm for decomposition-based multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 376–390, 2018.

[34] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu, "MOEA/D with adaptive weight adjustment," *Evolutionary Computation*, vol. 22, no. 2, pp. 231–264, 2014.

[35] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach." *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2014.

[36] H. Xu, W. Zeng, D. Zhang, and X. Zeng, "MOEA/HD: A multiobjective evolutionary algorithm based on hierarchical decomposition," *IEEE transactions on Cybernetics*, vol. 49, no. 2, pp. 517–526, 2019.

[37] M. Li and X. Yao, "What weights work for you? adapting weights for any pareto front shape in decomposition-based evolutionary multi-objective optimisation," *arXiv preprint arXiv:1709.02679*, 2017.

[38] Y. Liu, D. Gong, X. Sun, and Y. Zhang, "Many-objective evolutionary optimization based on reference points," *Applied Soft Computing*, vol. 50, no. 1, pp. 344–355, 2017.

[39] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 609–622, 2017.

[40] H.-L. Liu, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 433–448, 2018.

[41] F. Gu and Y.-M. Cheung, "Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 211–225, 2018.

[42] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[43] N. Masuyama and C. K. Loo, "Growing neural gas with correntropy induced metric," in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2016, pp. 1–7.

[44] N. Masuyama, C. K. Loo, and S. Wermter, "A kernel bayesian adaptive resonance theory with a topological structure," *International Journal of Neural Systems*, 2018.

[45] L. Martí, J. García, A. Berlanga, and J. M. Molina, "Introducing MONEDA: Scalable multiobjective optimization with a neural estimation of distribution algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2008, pp. 689–696.

[46] L. Martí, J. GarcíA, A. Berlanga, C. A. C. Coello, and J. M. Molina, "MB-GNG: Addressing drawbacks in multi-objective optimization estimation of distribution algorithms," *Operations Research Letters*, vol. 39, no. 2, pp. 150–154, 2011.

[47] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 41–63, 2008.

[48] D. Büche, M. Milano, and P. Koumoutsakos, "Self-organizing maps for multi-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 152–155.

[49] S. Obayashi and D. Sasaki, "Visualization and data mining of pareto solutions using self-organizing map," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2003, pp. 796–809.

[50] H. Zhang, A. Zhou, S. Song, Q. Zhang, X.-Z. Gao, and J. Zhang, "A self-organizing multiobjective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 792–806, 2016.

[51] K. Li, R. Chen, G. Fu, and X. Yao, "Two-archive evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 303–315, 2019.

[52] Y. Liu, D. Gong, J. Sun, and Y. Jin, "A many-objective evolutionary algorithm using a one-by-one selection strategy," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2689–2702, 2017.

[53] Y. Liu, H. Ishibuchi, Y. Nojima, N. Masuyama, and K. Shang, "Improving 1by1ea to handle various shapes of pareto fronts," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2018, pp. 311–322.

[54] D. Gong, Y. Liu, and G. G. Yen, "A meta-objective approach for many-objective evolutionary optimization," *Evolutionary Computation*, 2018. [Online]. Available: http://dx.doi.org/10.1162/evco_a_00243

[55] Y. Quintana-Pacheco, D. Ruiz-Fernández, and A. Magrans-Rico, "Growing neural gas approach for obtaining homogeneous maps by restricting the insertion of new nodes," *Neural Networks*, vol. 54, pp. 95–102, 2014.

[56] D. Brockhoff and E. Zitzler, "Objective reduction in evolutionary multiobjective optimization: Theory and applications," *Evolutionary computation*, vol. 17, no. 2, pp. 135–166, 2009.

[57] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[58] M. Li, C. Grosan, S. Yang, X. Liu, and X. Yao, "Multiline distance minimization: A visualized many-objective test problem suite," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 61–78, 2018.

[59] Y. Liu, H. Ishibuchi, Y. Nojima, N. Masuyama, and K. Shang, "A double-niched evolutionary algorithm and its behavior on polygon-based problems," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2018, pp. 262–273.

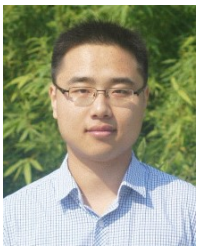
[60] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[61] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[62] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2015, pp. 110–125.

[63] H. Ishibuchi, R. Imada, N. Masuyama, and Y. Nojima, "Comparison of hypervolume, IGD and IGD+ from the viewpoint of optimal distributions of solutions," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2019, pp. 332–345.

[64] M. Li and X. Yao, "Quality evaluation of solution sets in multiobjective optimisation: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, p. 26, 2019.



Yiping Liu (M'18) received the Ph.D. degree in control theory and control engineering from China University of Mining and Technology, China in 2017. During 2016-2017, he was a visiting scholar in the School of Electrical and Computer Engineering, Oklahoma State University, USA. He is currently a Research Assistant Professor in the Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan. His research interests include evolutionary computation, multi-objective optimization, and machine learning.

Hisao Ishibuchi (M'93–SM'10–F'14) received the B.S. and M.S. degrees in precision mechanics from Kyoto University, Kyoto, Japan, in 1985 and 1987, respectively, and the Ph.D. degree in computer science from Osaka Prefecture University, Sakai, Osaka, Japan, in 1992. Since 1987, he has been with Osaka Prefecture University. Since 2017, he is a Chair Professor at Southern University of Science and Technology, China. His research interests include fuzzy rule-based classifiers, evolutionary multi-objective and many-objective optimization, memetic algorithms, and evolutionary games.

Dr. Ishibuchi was the IEEE Computational Intelligence Society (CIS) Vice-President for Technical Activities in 2010-2013. Currently he is an AdCom member of the IEEE CIS (2014-2019), and the Editor-in-Chief of the IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE (2014-2019).

Naoki Masuyama (S'12–M'16) graduated from Nihon University in 2010, received the M.E. degree from Tokyo Metropolitan University in 2012, and he obtained the Ph.D. degree from Faculty of Computer Science and Information Technology, University of Malaya, Malaysia in 2016. He is currently an Assistant Professor in the Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan. His research interests include data mining and machine learning.

Yusuke Nojima (S'99–M'04) received the B.S. and M.S. Degrees in mechanical engineering from Osaka Institute of Technology, Osaka, Japan, in 1999 and 2001, respectively, and the Ph.D. degree in system function science from Kobe University, Hyogo, Japan, in 2004. Since 2004, he has been with Osaka Prefecture University, Osaka, Japan, where he was a Research Associate and is currently an Associate Professor in the Department of Computer Science and Intelligent Systems. His research interests include evolutionary machine learning and evolutionary multi-objective optimization.