Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

An Adaptive Reference Vector Guided Evolutionary Algorithm Using Growing Neural Gas for Many-objective Optimization of Irregular Problems

Qiqi Liu, Yaochu Jin, Martin Heiderich, Tobias Rodemann, Guo Yu

2020

Preprint:

This is an accepted article published in IEEE Transactions on Cybernetics. The final authenticated version is available online at: https://doi.org/10.1109/TCYB.2020.3020630 Copyright 2020 IEEE

An Adaptive Reference Vector Guided Evolutionary Algorithm Using Growing Neural Gas for Many-objective Optimization of Irregular Problems

Qiqi Liu, Yaochu Jin, *Fellow, IEEE,* Martin Heiderich, Tobias Rodemann and Guo Yu

***Abstract*—Most reference vector based decomposition algorithms for solving multi-objective optimization problems may not be well suited for solving problems with irregular Pareto fronts because the distribution of predefined reference vectors may not match well with the distribution of the Pareto optimal solutions. Thus, adaptation of the reference vectors is an intuitive way for decomposition based algorithms to deal with irregular Pareto fronts. However, most existing methods frequently change the reference vectors based on the activeness of the reference vectors within specific generations, slowing down the convergence of the search process. To address this issue, we propose a new method to learn the distribution of the reference vectors using the growing neural gas network to achieve automatic yet stable adaptation. To this end, an improved growing neural gas is designed for learning the topology of the Pareto fronts with the solutions generated during a period of the search process as the training data. We use the individuals in the current population as well as those in previous generations to train the growing neural gas to strike a balance between exploration and exploitation. Comparative studies conducted on popular benchmark problems and a real-world hybrid vehicle controller design problem with complex and irregular Pareto fronts show that the proposed method is very competitive.**

***Index Terms*—Evolutionary many-objective optimization, reference vector, irregular Pareto front, growing neural gas**

I. Introduction

A lot of progress has been made over the past dacade on solving many-objective optimization problems (MaOPs) using multi-objective evolutionary algorithms (MOEAs). The methodologies reported in the literature in solving MaOPs can be categorized into four groups, namely, modification of the canonical dominance relationship [1]–[5], use of a performance indicator [6]–[8], decomposition of the MaOP into single-objective optimization problems [9]–[11], and introduction of a secondary selection criterion [12], [13]. However, less attention has been paid on solving MaOPs with irregular Pareto fronts (PFs), irregular problems for short. Irregular PFs do not cover the whole objective space, which can be discontinuous, inverted, or degenerate. Consequently, predefined evenly distributed weight or reference vectors typically used in decomposition based MOEAs may not be well suited for dealing with irregular MaOPs [14]–[16], since many reference vectors may be wasted because no solutions can be associated with them. An illustrative example is given in Fig. 1, where one set of predefined reference vectors and one set of adaptive reference vectors are adopted for guiding the search. The predefined set of reference vectors is evenly distributed in the objective space, but only a very small number of them covers the true Pareto front, resulting in a waste of many reference vectors. Moreover, unwanted solutions may be chosen by predefined reference vectors, even if they are far from the current PF, as shown in Fig. 1. As a result, the performance of decomposition based MOEAs may seriously deteriorate. Therefore, reference vector adaptation becomes imperative for decomposition based MOEAs to achieve good performance in solving irregular MaOPs.

One intuitive idea for decomposition based MOEAs to handle irregular problems is to adjust the reference vectors with which no solution is associated. For convenience, reference vectors with which no solution associated are called inactive reference vectors; otherwise, they are active. Note however, that adaptation of reference vectors is not trivial. The convergence speed will be slowed down and the search direction may be misled if the reference vectors are inappropriately adjusted. One reason for maladaptation of the reference vectors is that the population in the early search stage is usually far away from the true PFs, making it hard for the algorithm to figure out the right distribution of the Pareto optimal solutions. Taking this into consideration, the reference vectors in MOEA/D-AM2M, [15], for example, are adjusted after a predefined number of generations based on the distribution of solutions in that particular generation only. By contrast, MOEA/D-AWA [17] adds new reference vectors in regions where reference vectors are sparsely distributed and deletes reference vectors in the crowded region. Recently, an algorithm, termed MOEA/D-LTD, was proposed based on MOEA/D [18] by estimating the PFs and then sampling reference vectors according to the estimated PFs after a specific number of generations. Although adjusting the reference vectors after some predefined generations can avoid slowing down the convergence, the promising solutions may get lost in-between because no reference vectors can be associated with them or several promising solutions are associated with the same reference vector. Besides, it is not straightforward to determine how many generations we should

Qiqi Liu, Yaochu Jin and Guo Yu are with the Department of Computer Science, University of Surrey, Guildford, Surrey GU2 7XH, UK. e-mail: (qiqi.liu; yaochu.jin; guo.yu)@surrey.ac.uk. (*Corresponding author: Yaochu Jin*)

Martin Heiderich is with Honda R&D Europe (Deutschland) GmbH, Carl-Legien-Strasse 30, D-63073 Offenbach/Main, Germany. e-mail: martin_heiderich@de.hrdeu.com. Tobias Rodemann is with Honda Research Institute Europe,Carl-Legien-Strasse 30,D-63073 Offenbach/Main, Germany. e-mail: tobias.rodemann@honda-ri.de.

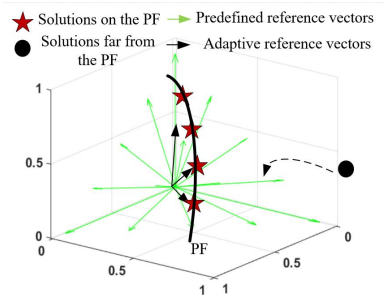


Fig. 1: The predefined reference vectors do not match well with the PF, resulting in that only a small number of reference vector have solutions associated with. Thus, most of the predefined reference vectors are wasted.

wait before adapting the reference vectors since different problems may have very different PF shapes and the complexity of each problem is also different.

Another natural way of adapting inactive reference vectors is to delete the inactive reference vectors and replace them with newly generated reference vectors [11], [14], [19], [20]. In [11], a variant of the reference vector guided evolutionary algorithm (RVEA), termed RVEA$*$ was proposed, where two sets of reference vectors are adopted for handling irregular problems. The inactive reference vectors in the set of reference vectors are replaced by randomly generated new vectors according to the range of objective values of all non-dominated solutions at each generation. While in [19], [20], additional reference points are added around the active reference points and those newly included reference points whose niche counts are more than one will be deleted to avoid wasting computational resources. However, deleting reference vectors that are inactive in one particular generation may be harmful as they may become active again in the next generation. That is, it is difficult to determine when and which reference vectors should be adjusted without apriori knowledge about the true Pareto front. To alleviate the above problem, at most one inactive reference vector is adjusted in every generation in [21] to avoid the loss of potentially useful inactive reference vectors and a secondary criterion is adopted to facilitate the angle penalized distance based environmental selection.

To address the above challenges, it is desirable to learn the location of reference vectors based on the knowledge acquired in the whole search history instead of based on the solutions in the current generation only. In [22], the active and inactive reference vectors after cascade clustering selection are used to train a classifier using incremental support vector machine to estimate the regions where the reference vectors should be located. In [23], self-organizing map (SOM) is adopted to learn the distribution of weight vectors using the neighbour information in the locations of the obtained solutions.

Apart from adapting reference vectors for decomposition based algorithms, clustering based methods have also been shown effective for solving MaOPs with irregular Pareto fronts. Clustering based methods have natural advantages in dealing with irregular problems since the reference vectors are generated based on the structure in the distribution of the solutions identified by a clustering algorithm rather than predefined. In [24], solutions in the current population are sorted at first based on the Pareto dominance relationship, and then k-means clustering is applied to group the solutions in the last non-dominated front to be selected based on the Euclidean distance of the solutions to the ideal point [24]. By contrast, a bottom-up hierarchical clustering algorithm is adopted [25], which can more precisely identify the center of a cluster to generate a corresponding reference vector. In [26], both partitional clustering and hierarchical clustering are adopted and the similarity of the solutions is measured by the acute angle between them. Note that the clustering criterion in the clustering based methods plays an important role in selecting promising solutions, which should be carefully designed to cluster solutions during the search process to capture the true geometry of the PFs.

There are also algorithms that do not belong to either reference vector adaptation or clustering, such as PICEA-w [27], AR-MOEA [28], and VaEA [29]. The reference vectors in PICEA-w [27] are co-evolved with the solutions rather than being replaced or deleted whenever they become inactive. AR-MOEA [28] adaptively adjusts the distribution of the reference points for calculating the IGD-NS indicator [30] and the solutions are selected based on their IGD-NS indicator value. In VaEA [29], the solutions in the last non-dominated front to be selected having the largest similarity to the solutions in the previous fronts will be chosen.

Motivated by various reference vector adaptation based methods, we hypothesize that it is necessary to take both the history information and neighbour information of the solutions in the current generation into consideration when we adapt the reference vectors. We argue that the reference vectors cannot be adjusted simply because they are inactive in single generations. Instead, we need to learn the distribution of the reference vectors based on the overall information of the solutions found during the search process. In this study, we propose to use an improved version of the original growing neural gas (GNG) [31], termed iGNG, to learn the distribution of reference vectors, in which the position of nodes of the iGNG is used to represent the reference vectors and the solutions generated during the search process are seen as the input to train the iGNG. We call the iGNG based reference vector adaptation for evolutionary optimization RVEA-iGNG. GNG is a self-organizing neural network inspired from SOM and it is able to gradually adjust the location of its nodes to the data, thereby incrementally learning the topology of the data. This property makes the GNG well suited for learning the distribution of reference vectors since it makes good sense that a reference vector and its neighbours are able to gradually move towards the input data (here the locations of the non-dominated solutions), if there are solutions always associated with this reference vector. By doing so, the topology of the Pareto front can be learned and the distribution of the reference vectors can be determined accordingly.

It is worth mentioning that we found a very recent work, called DEA-GNG, in which also a growing neural gas is

adopted to adapt the weight vectors [32] when we were preparing this paper. Compared with DEA-GNG, RVEA-iGNG proposed in the present work is able to quickly adjust the GNG network, when there is an abrupt change in the distribution of the population, which may frequently happen in evolutionary search. This is mainly attributed to the different methods for node deletion and addition used in DEA-GNG and RVEA-iGNG. First, node deletion in DEA-GNG is based on the solutions in two successive generations only, whereas node deletion in RVEA-iGNG is based on the solutions over a number of generations, making the iGNG less sensitive to temporary losses of solutions in the current population. Second, iGNG in RVEA-iGNG is trained by solutions in both the current population and the archive, and allows to add multiple new nodes in one iteration, while in DEA-GNG, only one new node can be inserted. It should be emphasized that being able to adding multiple new nodes and training the iGNG using solutions both in current population and in the archive enable RVEA-iGNG to quickly respond to large changes in the population, thereby reducing the risk of losing promising solutions.

Main contributions of this work are briefly summarized as follows:

- An improved growing neural gas (iGNG) is proposed to adapt the reference vectors of a decomposition based evolutionary algorithm for solving MOEAs with irregular Pareto fronts. The proposed iGNG deletes nodes using solutions in the current and a number of previous populations, and add new nodes based on solutions in the currently population and in an archive as well. This way, RVEA-iGNG is able to quickly yet steadily adapt the reference vectors to achieve a good balance between exploration and exploitation, thereby properly guiding the evolutionary search towards the Pareto fronts having an irregular shape.
- The position of the boundary nodes of the iGNG is extended according to the weights of their directly connected neighboring nodes to avoid the issue of shrinking boundary nodes, which effectively enhances the exploratory capability of the RVEA-iGNG.

The remainder of the paper is organized as follows. Section II introduces the basics of RVEA used as the search algorithm and the GNG neural network model for reference vector adaptation. Section III presents the proposed algorithm in detail, followed by experimental studies in Section IV. Finally, conclusions and discussions are given in Section V.

II. Background

A. Reference Vector Guided Evolutionary Algorithm

Decomposition based algorithms predefine a set of uniformly distributed reference or weight vectors to divide the objective space into several subspaces, thereby decomposing a multi-objective optimization problem into a number of single-objective optimization problems. In this sense, the reference vector guided evolutionary algorithm [11], termed as RVEA, can be considered as a decomposition based algorithm. Note however, that RVEA is also meant for incorporating user preferences [33], which can therefore also be seen as a preference driven MOEA. In the canonical RVEA, the uniformly distributed reference vectors are generated by the canonical simplex-lattice design method [34].

After generating a set of reference vectors, the solutions in the population will be associated to different reference vectors based on the angle between the reference vectors and the solutions. A solution will be assigned to a reference vector if the angle between the solution and the reference vector is the smallest among the angles between the solution and all reference vectors.

The angle penalized distance (APD) based selection mechanism in RVEA is designed to achieve a balance between convergence and diversity. Solutions with a larger APD value will be more likely to survive. The pressure towards convergence is controlled by the distance between the transformed objective value and the ideal point, while diversity is promoted by the angle between the solution and its associated reference vector. Different from RVEA, in [16], angle between solutions is used to main the diversity of population.

$$d_{t,i,j} = (1 + P(\theta_{t,i,j})) \cdot \left\| f'_{t,i} \right\| \tag{1}$$

where $\theta_{t,i,j}$ and $\left\| f'_{t,i} \right\|$ represents the angle and distance, respectively. $P(\theta_{t,i,j})$ is a penalty function related to $\theta_{t,i,j}$.

$$P(\theta_{t,i,j}) = M \cdot (\frac{t}{t_{max}})^{\alpha} \cdot \frac{\theta_{t,i,j}}{\gamma_{v_{t,j}}} \tag{2}$$

and

$$\gamma_{v_{t,j}} = \overset{min}{i \in \{1,\cdots,N\}} \langle v_{t,i}, v_{t,j} \rangle \tag{3}$$

where M is the number of objectives, $\gamma_{v_{t,j}}$ is the smallest angle between all pairs of reference vectors, which is used to normalize the angle between reference vectors. By using the APD selection criterion, both convergence and diversity can be promoted. In non-decomposition based algorithms, e.g., in [35], convergence and diversity are usually considered separately. Although convergence and diversity are both important for guiding the search process, convergence should usually be more focused on at the early stage and diversity should be more prioritized at late stage, which is controlled by α. Note that for handling problems with different objective scales, RVEA also adjusts the reference vectors according to the range of solutions after a fixed number of generations.

B. Growing Neural Gas Network

GNG was proposed by Fritzke [31] for learning the topology of data, belonging to the family of the self-organizing mappings (SOMs) [36]. Compared with the SOM, the GNG does not need to tune the parameters and the isolated nodes will be deleted over the time. GNG learns the topology of the data and similar inputs will be mapped to adjacent nodes in the

Algorithm 1: Growing neural gas.

Input : Growing neural gas parameters $params$, Training data $input$
Output : Growing neural gas net

Initialize: Start with two nodes at random position;
Generate an input data $input$;
Find the nearest node s_1 and the second nearest node s_2 to $input$;
Increase the age of all connections emanating from s_1;
Calculate the error of s_1 using the Euclidean distance between s_1 and $input$;
$\Delta error(s_1) = \|w_{s_1} - input\|$;
Move s_1 and its direct connected neighbours towards $input$ using the steps ε_b and ε_n, respectively;
$\Delta w_{s_1} = \varepsilon_b(input - w_{s_1})$;
$\Delta w_n = \varepsilon_n(input - w_n)$;
if s_1 *and* s_2 *are connected* **then**
    set age = 0;
else
    create a connection between s_1 and s_2;
end
Remove the connection if $age > a_{max}$ and remove the isolated nodes;
if $\sim$ *mod(the number of input data generated,*λ *)&* $size(w) < N_{max}$ **then**
    q $\leftarrow$ Find the node with the largest $\triangle error$;
    insert a node r between q and its neighbour f which has the largest error among its neighbours;
    $w_r = 0.5(w_q + w_f)$;
    create a connection between r and q, r and f;
    remove the connection between q and f;
    $\Delta error(q) = error(q) * \alpha$;
    $\Delta error(f) = error(f) * \alpha$;
    $\Delta error(r) = error(q)$;
end
Multiply all errors by d;

network. Both the SOM and GNG can be used for clustering data but the GNG has a number of advantages compared with other clustering methods. For instance, the number of clusters in the GNG does not need to be specified in advance compared to the k-means clustering [37]. It is well known that k-means can not cluster data with an irregular topology, while the GNG can learn the topology of the data automatically and works well for data with an irregular distribution. Although hierarchical clustering [38] can overcome the drawbacks of the k-means algorithm, the number of clusters in hierarchical clustering algorithms can not grow or decrease over the time in contrast to the GNG. The performance of the GNG on clustering is further verified in [39]–[42], in which the GNG is extended to handle non-stationary data streams. A major drawback of the improved GNG for handling non-stationary data is that several additional parameters are introduced and they are usually problems-specific. In this study, an improved GNG aiming to cluster non-stationary data is proposed, since

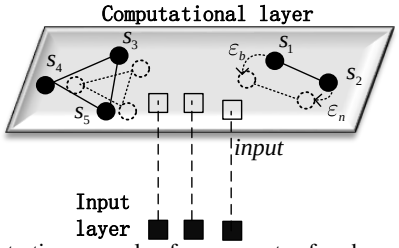


Fig. 2: An illustrative example of movements of nodes and connections in a growing neural gas network. The solid circles and lines represent the topology before new input data is presented. The dashed circles are the adjusted positions of the nodes when the network is trained with the new input.

non-dominated solutions are constantly generated during the search process. A GNG network, a single-layer artificial neural network, is composed of a number of nodes and connections. Each node has a weight vector, which corresponds to the position of nodes in the network. Fig. 2 shows an illustrative example of a GNG, where, e.g., the weight vector of node s_1 is w_{s1}. The nodes in the network may have connections with each other. The nodes and connections in the network, denoted by solid circles and lines, respectively, will adjust their positions (denoted by dashed circles and dashed lines, respectively) after new inputs are presented. In Fig. 2, the nearest node and the second nearest node of the new input $input$ are s_1 and s_2, respectively. We can find that the neighbour of node s_1 is node s_2 because they are connected. The pseudo code of the GNG is presented in Algorithm 1. Node $s1$ and its neighbours will move towards $input$ by a step ε_b and ε_n, respectively. In GNG, each connection has an age. Each time when a new input data is presented, the age of connections between the winner and its neighbours will increase by one and the connection whose age is larger than a predefined age threshold a_{max} will be deleted. Moreover, the error of a node is measured by the Euclidean distance between the node and the input data. A new node will be added between the node with the largest accumulated error and its neighbour, which has the largest accumulated error among all neighbours. We can find from the description that the GNG has the implicit property to adjust the positions of the nodes .

III. Proposed Algorithm

The main idea of the algorithm proposed in this work, termed RVEA-iGNG, is that the weights of the nodes in an iGNG can be used to learn the distribution of the Pareto front, thereby adaptively determining the reference vectors for RVEA instead of predefining a set of uniformly distributed reference vectors.

A. A General Framework of RVEA-iGNG

The main framework of RVEA-iGNG is described in Algorithm 2, which can be divided into five main steps as follows.

1) Initialization (Lines 1-3): A set of uniformly distributed reference vectors and an initial population are generated

Algorithm 2: General Framework of RVEA-iGNG.

Input : Population P, Offspring O, Archive A, Parameters of iGNG $params$, Maximum generations $MaxGen$, The frequency of adapting reference vectors f_r
Output : The final Population P

1 Initialize: reference vectors V_0, $scale$ = 1,f_r = 0.1;
2 $net \leftarrow$ InitilizeGrowingNeuralGas(V_0,P,$params$) ;
3 $A \leftarrow$ empty set; $A \leftarrow$ UpdateArchive(P,A);
4 **if** *gen<MaxGen* **then**
5 $O \leftarrow$ Reproduction(P);
6 $P \leftarrow P \cup O$;
7 $A \leftarrow$ UpdateArchive(P,A);
8 **if** *gen<0.9*MaxGen* **then**
9 $net \leftarrow$ TrainingImprovedGrowingNeuralGas(P,A,$params$);
10 $V \leftarrow$ the weights of nodes of net multiply $scale$;
11 **end**
12 **if** $((\frac{gen}{MaxGen} \mod f_r) == 0)$ **then**
13 $scale$= max($A.objs$) - min($A.objs$)
14 **end**
15 $P \leftarrow$ EnvironmentalSelection(P,A,V);
16 **end**

and the reference vectors that have solutions associated with them are used to train the initial growing neural gas network. The non-dominated solutions in the initial population are copied to archive A.

2) Update Archive (Lines 5-7): The offspring are generated by the simulated binary crossover [43] and polynomial mutation [44] operator using the solutions in the current population. In every generation, an archive is maintained to preserve more promising solutions, in which solutions are maintained by the unary epsilon indicator [6].
3) Training Improved Growing Neural Gas (Lines 8-10): To simultaneously promote exploitation and exploration, we utilize both the solutions in the archive and the solutions in the current population as the input data to train the growing neural gas network. Similar to [14], [32], we also keep the reference vectors in the last $0.1 * MaxGen$ generations unchanged.
4) Adjusting the Reference Vectors (Lines 12-14): At every generation, the weights of the nodes of an improved growing neural gas are used as the reference vectors to guide the search process. Similar to RVEA, we adapt the reference vectors after every $f_r * MaxGen$ generations for handling problems of different objective scales, in which f_r is the frequency of adapting reference vectors. $A.objs$ means the objective values of the solutions in the archive.
5) Environmental Selection (Line 15): The environmental selection is based on the reference vectors generated by the trained growing neural gas. The angle penalized distance is used as the selection criterion in environmental selection. Solutions in the archive also go through the environmental selection process in case that promising reference vectors are generated while promising solutions get lost.

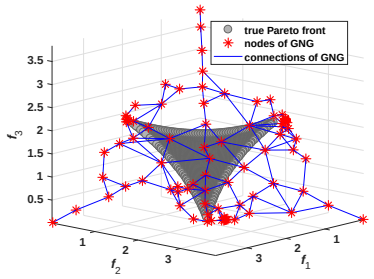


Fig. 3: An illustrative example of the drawbacks of the GNG in dealing with non-stationary data. The grey circles denote the true Pareto front. The nodes and connections of the GNG are denoted by stars and lines, respectively.

B. A New Strategy for Adjusting the Nodes in iGNG

The weight vectors of the nodes in the iGNG are used as the reference vectors to guide the search process. The solutions generated during the search process will be used to train the network. Each time when there is a new non-dominated solution, the node in the iGNG having the nearest Euclidean distance to the solution will move towards the solution. Most importantly, the neighbours of the node will also move towards the solution since they are connected with each other.

The canonical GNG is designed for learning the topology of stationary data. As a result, the network has no ability to quickly adjust itself if the input data suddenly changes. For example, as shown in Fig. 3, the GNG network is trained by using solutions at the early stage of the search process as the input data and the true Pareto front is denoted by circles. In this case, the GNG network does not match well with the true Pareto front. We can find that there is no isolated or dead nodes in this network and the number of nodes has already reached the predefined maximum number. Note that the number of nodes in the GNG has a maximum value since their weights are used to represent the reference vectors. In our study, the maximum number of nodes is set to 1.5 times of the population size. Thus, in this example, no nodes will be deleted if the original GNG network is adopted since only isolated nodes will be deleted in original GNG. As a result, the topology of the GNG network can hardly be adjusted to precisely match the distribution of the solutions in the population, especially when the distribution of the solutions has abrupt changes during the search process.

To solve this issue, in our study, we propose a new strategy to adjust the nodes in the iGNG, to deal with the non-stationary solutions during the search process. The main idea is to delete those nodes that have little contributions to the population and to add more nodes in promising regions. Bearing this in mind, we design a strategy for adjusting nodes in the following two steps:

1) The nodes of the growing neural gas can be seen as non-contributory if no enough solutions are associated with them over a period. We define a $flag$ to determine which nodes will be deleted. If the sum of the age of the connections emanating from a node in generation t is the same as that in the $t-1$ generation, then the $flag$ will increase by one. So we rank the $flag$ and then delete those nodes whose $flag$ ranks top over a period.
2) After deleting those non-contributory nodes, new nodes will be generated in the iGNG until the predefined maximum number of nodes is reached.

Algorithm 3: Training Improved Growing Neural Gas (iGNG).

Input : iGNG network net, iGNG parameters $params$, TrainingData TD, Generation number gen, Population size N
Output : iGNG network net

```
w = net.w;
for i=1:size(w) do
    ageSum(i) ← sum the age of all connections emanating from i;
    if ageSum(i,gen)==ageSum(i,gen − 1) then
        flag = flag + 1;
    end
end
if size(w)> 1.5*N then
    del ← Rank the flag in descending order and find the the top 0.5N nodes;
    flag(del) ← empty set;
    Delete the del and delete the connections emanating from del ;
    for i=1:size(TD) do
        net ← GrowingNeuralGas(TD(i), 1.5 ∗ N, params);
    end
end
```

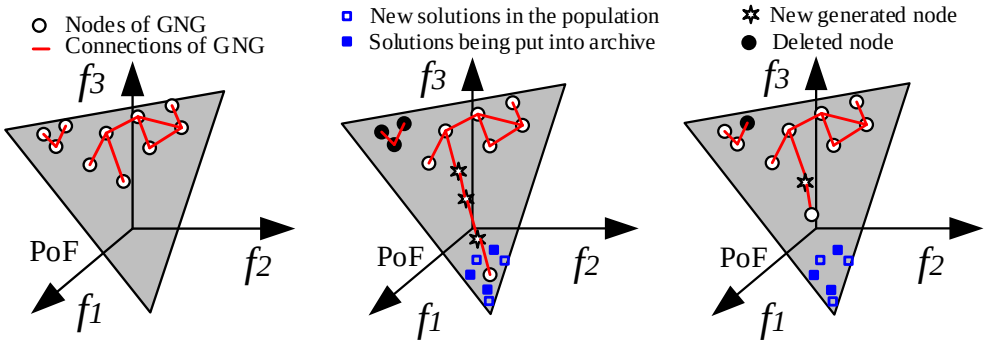
We know that adding a node only happens when the number of input data is an integer multiple of parameter λ, as shown in Line 16 of Algorithm 1. So it is very likely that there will be little chance to generate a new promising node when the number of promising input data is not enough, especially when the promising input data (a new solution) survives in one or a number of generations. In RVEA-iGNG, the maximum number of nodes in the iGNG is set to 1.5 times of the population size (N). The pseudo code for training iGNG is presented in Algorithm 3. We first rank the $1.5N$ nodes according to the age information learning over a period and delete the worst $0.5N$ nodes when the number of nodes reaches $1.5N$. If there is no input data near a node, then the age of connections emanating from the node will remain unchanged. From this point, we count the times of a node whose age keeps unchanged over the period that the number of nodes in the iGNG varies from N to $1.5N$. The nodes whose count ranks $0.5N$ top will be deleted. After that, the nodes of iGNG will keep increasing since the the maximum number of nodes of iGNG is set to $1.5N$. In this regard, more promising regions will be explored because there is enough time for the network to grow, even when the promising solutions may occur in just one or several generations.

Different from the method of adjusting the nodes in RVEA-iGNG, a hit point is set to control when a node is deleted in DEA-GNG. The maximum hit point threshold of each node is set to two times of the archive size. Therefore, if a node cannot be a winner in two successive generations with the archive as the training data, then the node will be deleted. This way, the node is deleted only based on the information of the archive in two successive generations.

To further understand the differences between DEA-GNG and RVEA-iGNG, Fig. 4 gives an example. Fig. 4 (a) plots the GNG trained by the solutions in the previous generations and Fig. 4 (b) and Fig. 4 (c) plot the retrained GNG with a set of promising solutions that are generated in the current population as the training data. In RVEA-iGNG, the current population and the archive are combined to train the iGNG, whereas only the solutions in the archive are used to train the GNG in DEA-GNG. For DEA-GNG, it is very likely that not enough solutions in the current population can be chosen to be stored in the archive. For example, in Fig. 4 (c), only the solutions denoted by the solid squares are used as the training data. Thus, the nodes in GNG cannot be quickly pulled to the region where new solutions are located. In addition, each time at most one node is deemed as dead when the hit point of this node becomes zero, thus, at most one new node can be generated and the number of nodes will directly reaches the predefined network size after the addition of one new node. A new node will be generated after every λ new input data provided that the number of nodes does not exceed the predefined network size. Thus, if a new promising region appears, the GNG may not be able to generate a new node in the promising region in DEA-GNG. For RVEA-iGNG, as shown in Fig. 4 (b), the nodes will be pulled more deeply towards the region where new solutions are located than that in DEA-GNG, since all new solutions are used to train the GNG. After $0.5N$ nodes, denoted as solid circles, are deleted, $0.5N$ new nodes can be generated successively until the number of nodes reaches $1.5N$. This way, enough promising nodes can be generated by RVEA-iGNG over a number of generations during which the number of the nodes grows from N to $1.5N$. Consequently, the distribution information of the solutions generated in the past generations and that of the solutions in the current population can be fully utilized.

C. Training improved GNG

An archive based on unary epsilon indicator [6] is adopted to preserve the non-dominated solutions generated during the search process. However, the non-dominated solutions in the previous generations may become dominated during the evolutionary search. The current population usually contains new information for searching for the true PFs. In this study, previous non-dominated solutions stored in the archive and



(a) trained GNG (b) adjusting nodes in RVEA-iGNG (c) adjusting nodes in DEA-GNG

Fig. 4: The mechanism of adjusting nodes in RVEA-iGNG and DEA-GNG

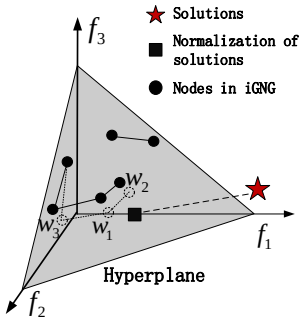


Fig. 5: An illustrative example of the normalization of solutions in the population to train the nodes in a iGNG. The position of nodes of the iGNG denoted by solid circles will move to the position denoted by dashed circles if a new input is given.

solutions in the current population are combined in every generation for training the iGNG.

The weight vectors of nodes are adjusted based on the Euclidean distance between the solutions and the nodes. However, the objective values of the solutions vary dramatically during the search process. So the solutions first need to be projected to a hyperplane so that the nodes and the solutions can exist on the same hyperplane. In this study, we use the following method to normalize the solutions in the population to make sure that the scale of solutions is within the range of $[0, 1]$.

$$f_j^{'} = (f_j - f_j^*)/(f_j^{max} - f_j^*), \tag{4}$$

where f_j is the j-th objective value, f_j^* is the minimum value of the j-th objective among all non-dominated solutions and f_j^{max} is the maximum value of j-th objective among all non-dominated solutions in the population.

Fig. 5 plots the normalization of solutions. After all solutions are normalized to the hyperplane, the normalization of solutions and the nodes of the iGNG will stay in the same hyperplane. This way, the solutions can be used to train the nodes during the search process in case solutions in different generations have different distances to the ideal point and the solutions are of different scales for different objectives. After deleting $0.5N$ non-contributory nodes in the iGNG, the combined training data containing the solutions both in the archive and in the current population will enable the iGNG to generate new promising nodes quickly. Consequently, if the current population is distributed in the same region of the archive, the diversity of solutions will be enhanced in this region because the nodes in the iGNG can further fine tune their location with more input data. Thus, more training data will further enhance the distribution of the nodes as the network tries to add new nodes near the node with the largest accumulated error, that is, the largest accumulated distance. On the contrary, new regions may be explored by deleting non-contributory nodes and new nodes can be generated in the promising regions according to the distribution of the current population if the solutions in the current population are not located in the same region as those in the archive.

In this study, the size of the archive is set to $2N$ regardless the number of objectives. We adopt the unary epsilon indicator to manage the archive and the pseudo code of updating archive is shown in Algorithm 4.

Algorithm 4: Update Archive.

Input : Population P, Archive A, Population size N
Output : Archive A

$A \leftarrow P \cup A$;
$A \leftarrow$ NonDominanceSorting(A);
$Fit, I, C \leftarrow$ Calculate the fitness of the solutions in Archive using epsilon indicator ;
for *i=1:size(A) - N* **do**
  $d \leftarrow$ find the solution with the minimum fitness Fit;
  $Fit(d) \leftarrow 0$;
  Update the fitness of other solutions j except those solutions with $Fit = 0$ using $Fit(j) = Fit(j) + exp(-I(d, j)/C(j) \cdot 0.05)$;
end
$A \leftarrow$ DeleteExtremeWorseSolutions(A);

D. Preserving the Boundary of Pareto fronts

The iGNG can reflect the topology of a set of solutions provided that the network is adequately trained. As shown in Fig. 6, the iGNG network is trained for 80 iterations using

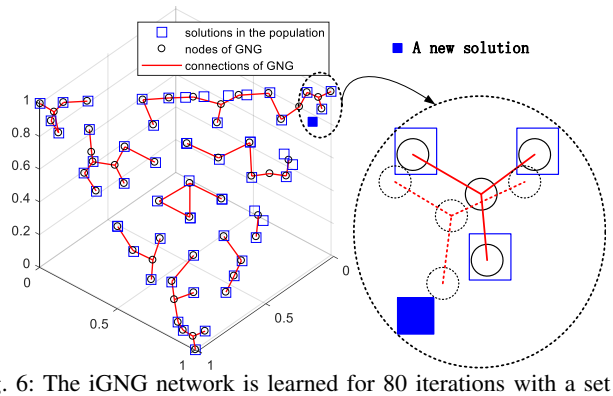


Fig. 6: The iGNG network is learned for 80 iterations with a set of solutions denoted by a square as the training data. Each time when there is a new solution generated near the boundary solutions, the nodes in the boundary will move a little towards the direction of the new solution.

the set of solutions denoted by squares as the training data. It can be found that the iGNG is able to accurately capture the topology of the distribution of the solutions. Assuming that the true Pareto front is an inverted triangular, solutions generated at the late stage of the search process will always locate inside the inverted triangular. In this case, the boundary of the Pareto front will be slightly inside the boundary of true Pareto front. For example, in Fig. 6, if a new solution denoted by a solid square is generated near the boundary solutions, the boundary of the Pareto front cannot be preserved. The reason is that the nodes in the iGNG always move towards the location of the solutions, as shown in Fig. 6. In our study, the weight vectors are used as the reference vectors, and therefore the reference vectors will locate slightly inside the boundary of the true PF.

To address the issue of shrinking boundaries, we propose to extend the boundary of iGNG according to the directly connected nodes of each boundary node in the trained iGNG without using any additional information. Take the trained iGNG in Fig. 6 as an example, we can find that the directly connected neighbours of the boundary nodes in the iGNG are all located in one side of its neighbour nodes. In this study, the weight of a node that has either the maximum or minimum value in one dimension compared with its directed connected neighboring nodes will be chosen as boundary nodes. Thus, we propose to adjust the the weights of the boundary nodes by utilizing the information of the directly connected nodes of the neighbouring nodes.

$$\begin{aligned} w_{bn} &= w_{bn} - (w_{\mu} - w_{bn}), \\ w_{\mu} &= mean(w_{dn}), \end{aligned} \tag{5}$$

where w_{bn} is the weights of the boundary nodes and w_{dn} is the weights of the directly connected nodes of the boundary nodes.

E. Environmental Selection

The pseudo code for environmental selection is presented in Algorithm 4. The environmental selection contains three main parts. The first part is to integrate the population and the archive for environmental selection. The reason is that the history promising solutions in the archive may match well with the new generated reference vectors, while the solutions in the current population may not . In this way, promising solutions in the archive may have a chance to generate promising offspring if they can survive in the APD selection criterion. The second part is the APD selection which is the same as that in RVEA [11]. Here we only select the solutions in the first Pareto front for APD selection. Moreover, we delete those extreme worse solutions whose Euclidean distance to the ideal point is γ times larger than the average distance of the whole population to the ideal point. Since the number of nodes in the iGNG varies from N to $1.5 * N$ and the weights of nodes are used as the reference vectors, the number of solutions surviving in the APD selection criterion may be more than the predefined population size N. So we will delete the crowded solutions if the number of survived solutions exceeds the population size. The survived solutions are first normalized according to Eq. (4) and then the solution whose Euclidean distance to its neighbors is the smallest among all will be deleted one by one until the number of solutions equals the population size.

Algorithm 5: Environmental selection.

Input : Population P, Archive A, reference vectors V
Output : Population P

$P \leftarrow P \cup A$;
$P \leftarrow$ NonDominanceSorting(P);
$P \leftarrow$ DeleteExtremeWorseSolutions(P);
$P \leftarrow$ APDSelection(P,V);
while *size(P)*$>$ N **do**
  $P \leftarrow$ DeleteCrowdedSolutions(P);
end

IV. Empirical Results

This section describes the experimental setting and the empirical results. Seven state-of-the-art MOEAs are adopted to compare with the proposed RVEA-iGNG in dealing with irregular problems. All algorithms under comparison are implemented on the PlatEMO platform [45] with MATLAB 2018a on Intel Core i7-8700 (3.20GHz).

A. Test problems and Compared Algorithms

Three test suites, MaF [46], DPF [47] and IMOP [48] are adopted to examine the performance of RVEA-iGNG on solving irregular problems. We also test the performance of RVEA-iGNG on solving a real-world application, controller design of hybrid electric vehicles (HEVs) [33]. MaF [46] is composed of 15 widely used test functions. MaF1 is a modified version of the inverted DTLZ1 [19], MaF2 is DTLZ2BZ [49], MaF6 is a modified version of DTLZ5(I,M) [50], MaF7 is DTLZ7 [51], MaF8 is a Pareto-box problem [52] and MaF10 and MaF11 are WFG1 and WFG2 [53]. DPF1 to DPF5 [47] are

recently designed test functions with degenerate Pareto fronts. The properties of all test suites are summarized in Table SI in the supplementary material.

RVEA-iGNG is compared with eight state-of-the-art algorithms, namely, MOEA/D-AWA [17], AR-MOEA [28], AdaW [14], VaEA [29], A-NSGA-III [19], DEA-GNG [32], MOEA/D-SOM [26], and RVEA$*$ [11] for solving problems with irregular PFs. In the following a brief description of the compared algorithms is provided.

1) MOEA/D-AWA [17]: This algorithm is based on the framework of MOEA/D. The addition and deletion of the weight vectors in MOEA/D-AWA are based on the sparsity degree of solutions in the current population and the weight vectors are adjusted periodically.
2) AR-MOEA [28]: The enhanced inverted generational distance indicator is used to identify solutions that make no contribution. A set of reference points for calculating the contribution of the solutions to the performance indicator are adaptively updated. AR-MOEA performs well not only on problems with regular but also irregular PFs, which is very promising.
3) VaEA [29]: Different from the decomposition based algorithms, VaEA does not require any specific reference points or weight vectors. VaEA utilizes the information of the solutions in different non-dominated fronts in the current population. The angle between the solutions is used to enhance the diversity of the population and worse solutions in terms of Euclidean distance are deleted to ensure the convergence of the algorithm.
4) A-NSGA-III [19]: The reference points in A-NSGA-III are added near the active reference points and are deleted when the newly added reference points fall in a crowded region.
5) DEA-GNG [32]: Two sets of reference vectors are combined, one is the predefined reference vectors and the other is defined as the positions of the nodes of the GNG, which are adapted during the search process. In addition, the topology of the solutions learned by the GNG is used to adjust the parameters of the scalarizing functions.
6) MOEA/D-SOM [26]: SOM utilizes the neighbour function to learn the topology of the input data. This property is first used to learn the distribution of the weight vectors in decomposition based algorithms and the algorithm performs well on degenerate problems.
7) RVEA$*$ [11]: An additional set of references vectors except for the predefined references are adopted in RVEA$*$ to handle problems with irregular PFs. For the additional set of reference vectors, inactive reference vectors will be replaced by new generated vectors based on the range of objectives in every generation.
8) AdaW [14]: An archive is maintained to store the well-distributed and better converged solutions and the distribution of the solutions in the archive are used to add new reference vectors in undeveloped and promising regions and to delete reference vectors that contribute little. The diversity preserving method in AdaW can both deal with badly scaled problems and problems with concave or convex PFs well.

B. Performance Indicators

Two widely used performance indicators, inverted generational distance (IGD$^+$) [54] and hypervolume (HV) [55] are adopted in our experimental study to compare the performance of the proposed RVEA-iGNG with other seven state-of-the-art algorithms. IGD$^+$ and HV can both be used to account for convergence and diversity. For calculating the IGD$^+$ indicator, a set of uniformly distributed reference points sampled from the true PFs are needed, while the calculation of HV indicator only needs a predefined reference point. In our experiments, a solution set obtained by an algorithm is first normalized when calculating HV indicator and the reference point is set to $(1.1, 1.1, \ldots, 1.1)$.

C. Parameter Setting

We compare the performance of the algorithms on 2-,3-,5-,10- and 15-objective problems. The corresponding population size and the number of fitness evaluations are set to 210 230, 240 and 50000, 100000, 150000 for 5-,10-, 15-objective MaF and DPF test instances, respectively. The corresponding population size and the number of fitness evaluations are set to 40, 66 and 40000, 60000 for 2-,3- objective IMOP test instances, respectively. Since the population size of MOEAD-SOM can not be set arbitrarily, the population size of MOEAD-SOM are set to 49, 81, 144, 256 and 256 for a fair comparison with other algorithms. For DEA-GNG, ϵ is set to 0.05π for 2- and 3-objective instances, 0.15π for 5- problems, and 0.2π for problems with 10 or 15 objectives. The archive size is set to $M * N$ for DEA-GNG (M is the number of objectives and N is the population size).

The parameters of the iGNG are set to $\varepsilon_b = 0.2, \varepsilon_n = 0.006, \alpha = 0.5, \alpha_{max} = 50, d = 0.995$, $\lambda = 50$. In RVEA-iGNG, the frequency of adapting reference vectors is set to $f_r = 0.1$;

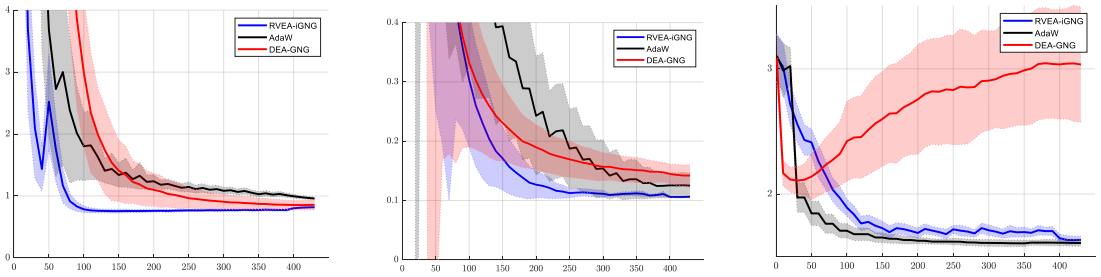
In this study, simulated binary crossover [43] and polynomial mutation [44] are adopted to generate offspring. The distribution index of crossover and the distribution index of mutation are set to $n_c = 20$ and $n_m = 20$, respectively. The crossover probability and the mutation probability are set to $p_c = 1$ and $p_m = \frac{1}{D}$ (D is the number of decision variables), respectively.

D. Results and Discussions

Table I presents the comparisons of RVEA-iGNG with other eight state-of-the-art algorithms in terms of IGD$^+$ and HV indicators in solving irregular problems and regular problems with complex PFs. The significance of the differences in performance is evaluated using Wilcoxons rank sum test [56]. From the table, we can find that RVEA-iGNG ranks the best overall compared with the eight algorithms, both in terms of the IGD$^+$ value and the HV value.

TABLE I: IGD$^+$ COMPARISONS AND HV COMPARISONS ON DEGENERATE, DISCONTINUOUS, INVERTED, COMPLEX, AND REGULAR TEST INSTANCES."+", "=", AND "−" DENOTE RVEA-iGNG PERFORMS SIGNIFICANTLY BETTER THAN, EQUIVALENT TO AND WORSE THAN THE COMPARED EIGHT ALGORITHMS, RESPECTIVELY.

RVEA-iGNG vs.	MOEA/D-SOM	DEA-GNG	A-NSGA-III	AR-MOEA	AdaW	RVEA*	MOEA/D-AWA	VaEA
				IGD$^+$ (+/=/-)				
degenerate	20/0/1	18/0/3	19/1/1	16/2/3	9/6/6	18/0/3	20/0/1	19/0/2
discontinuous	6/0/0	4/1/1	6/0/0	6/0/0	6/0/0	4/1/1	6/0/0	6/0/0
inverted	5/0/1	2/0/4	5/0/1	3/0/3	0/1/5	5/0/1	6/0/0	4/0/2
complex or regular	24/0/2	21/1/4	26/0/0	21/2/3	15/4/7	24/1/1	25/0/1	21/2/3
total	55/0/4	45/2/12	56/1/2	46/4/9	30/11/18	51/2/6	57/0/2	50/2/7
				HV (+/=/-)				
degenerate	18/1/2	16/3/2	17/1/3	16/1/4	14/5/2	17/3/1	19/0/2	15/1/5
discontinuous	4/0/2	3/2/1	1/3/2	4/1/1	3/0/3	5/1/0	4/1/1	5/0/1
inverted	5/0/1	3/0/3	3/0/3	5/1/0	4/1/1	5/0/1	5/0/1	5/1/0
complex or regular	26/0/0	23/1/2	19/1/6	16/3/7	16/4/6	23/2/1	21/1/4	20/1/5
total	53/1/5	45/6/8	40/5/14	41/6/12	37/10/12	50/6/3	49/2/8	45/3/11



(a) MaF7 (b) MaF8 (c) MaF2

Fig. 7: Real-time IGD$^+$ data as mean curves, with confidence intervals. The abscissae are the number of generations.

E. Comparisons on MaF Test Suite

Table SII presents the IGD$^+$ value of solutions obtained by RVEA-iGNG and other eight popular algorithms on solving MaF test suits. MaF is composed of 15 test problems of various properties, e.g., the true PFs of MaF1 and MaF4 are inverted, the true PFs of MaF7 and MaF11 are disconnected. Overall, RVEA-iGNG has achieved the best performance among the eight algorithms under comparison in terms of IGD$^+$ indicators. RVEA-iGNG wins on 18 instances in terms of IGD$^+$ out of 39 test instances in total. By contrast, MOEA/D-SOM, DEA-GNG, A-NSGA-III, AR-MOEA,AdaW, RVEA$*$, MOEA/D-AWA and VaEA win on one, two, zero, two, nine, three, zero, and four test instances in terms of IGD$^+$ out of 39 test instances, respectively. We find that MOEA/D-SOM and DEA-GNG are outperformed by RVEA-iGNG on most test instances, although they all adjust the reference or weight vectors by means of learning the topology of input data. We can also find that AdaW performs better than RVEA-iGNG on solving MaF2, MaF4 and MaF6 in terms of IGD$^+$, the reason is that the diversity preserving mechanism in AdaW is pretty good and also the PF boundary of these test problems is easily to be found. VaEA performs relatively well on both regular and irregular problems. However, the results obtained by VaEA are worse than RVEA-iGNG on most test instances. As previously discussed, adjusting the reference or weight vectors by deleting, adding or replacing the inactive reference vectors does not work well in many cases, which is confirmed by the inferior results obtained by A-NSGA-III, MOEA/D-AWA and RVEA$*$ presented in Tables SII.

Fig. S2 plots the decision variables of the non-dominated solutions obtained by RVEA-iGNG, and the eight compared algorithms on solving MaF8. MaF8 is a Pareto-box problem, and the Pareto optimal region of MaF8 in the decision space is always $2D$ manifold. The true Pareto set of MaF8 locates inside the polyhedron, as shown in Fig. S2. Therefore, the convergence and coverage of the solutions obtained by the compared algorithms can be directly visualized according to the distribution the decision variables of the obtained non-dominated solutions. The PF of MaF8 are found gradually during the search process and the distribution of solutions in the population changes their position frequently. We can find both DEA-GNG and AdaW fails to find the whole PF in limited time. This demonstrates RVEA-iGNG do have an advantage in dealing with problems which the search process has an abrupt change. It can also be found that the non-dominated solutions achieved by AR-MOEA and VaEA exhibit slightly worse performance in diversity, compared with RVEA-iGNG.

Fig. 7 plots the IGD$^+$ convergence curve of RVEA-iGNG, DEA-GNG and AdaW on solving 10-objective MaF7, MaF8 and MaF2 averaged over 30 independent runs. We can find that RVEA-iGNG converges much faster in on MaF7 (discontinuous) and MaF8 (degenerate) than DEA-GNG and AdaW. RVEA-iGNG converges slightly more slowly than AdaW on MaF2, but still much faster than DEA-GNG. This conforms that iGNG can more quickly adapt to abrupt changes in the population or when the PFs are found gradually during the

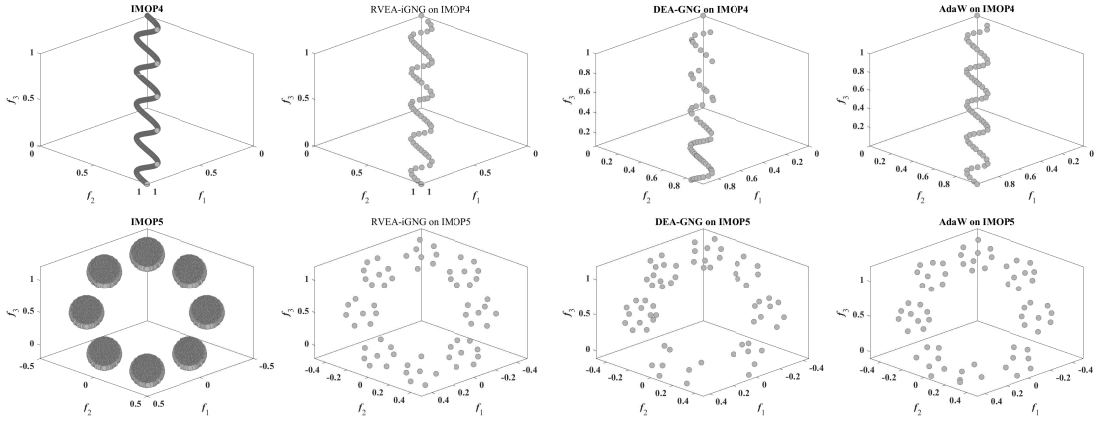


Fig. 8: The non-dominated solution set obtained by the eight compared algorithms on solving IMOP4 and IMOP5 with 3 objectives. The set with the median IGD$^+$ value out of 30 independent runs is plotted.

search process.

F. Comparisons on the DPF Test Suite

DTLZ5(I,M) [50] is widely used for testing the performance of the algorithms designated for tackling degenerate MaOPs. However, DTLZ5(I,M) is easy to solve and may not reflect the true properties of the real-world problems. Thus, the DPF test functions, proposed by Zhen et al. [47], aim to provide hard degenerate test problems. The hardnesses of the test functions include: 1) the correlation between the objectives can be nonlinear, and 2) the redundant objectives can be implicit, or the correlation of objectives are different in different parts of the objective space. These properties pose great challenges to many algorithms for dealing with degenerate problems. Table SIII presents the mean IGD$^+$ values of the solution sets obtained by RVEA-iGNG and other eight compared algorithms in 30 independent runs. It can be found that RVEA-iGNG has achieved the overall best performance in solving DPF test functions: It wins on 9 out of 15 test instances in terms of the IGD$^+$ values. RVEA-iGNG performs the best on most test instances of DPF test instances, showing the effectiveness of topology learning of using the improved GNG network. Of the nine algorithms, however, MOEA/D-SOM, DEA-GNG, A-NSGA-III, RVEA$*$, MOEA/D-AWA and VaEA do not perform well when solving DPF1 and DPF4. Although RVEA-iGNG achieves better performance in solving DPF2 and DPF3 with 10, and 15 objectives, five other compared algorithms, DEA-GNG, A-NSGA-III,AR-MOEA, RVEA$*$, and VaEA perform better than RVEA-iGNG on DPF3 with 5 objectives. We can also find that RVEA$*$ perform well on partial degenerate problem MaF5 and AdaW achieves the best on solving DPF4 with 10 and 15 objectives.

G. Comparisons on the IMOP Test Suite

We also test the proposed RVEA-iGNG on solving IMOP with complex Pareto fronts [48], which are proposed recently. Table SIV shows the mean IGD$^+$ values of the solution set obtained by nine algorithms in 30 independent runs. One property of IMOP is that the Pareto fronts are found gradually during the search process like MaF10, and another important property of IMOP is that the true Pareto fronts are quite irregular, which further poses great challenges for traditional methods. As shown in Fig. 8, it can be found the solution set obtained DEA-GNG performs worse than RVEA-iGNG and DEA-GNG fails to find the whole Pareto fronts on solving IMOP4 and IMOP5. For instance, the PF of IMOP4 are found gradually from bottom to top, the solutions obtained by DEA-GNG in the bottom is much denser than those in the top, meaning that the GNG network constructed in the early stage cannot be quickly adjusted when new promising regions occur. We can also find that AdaW performly slightly worse than RVEA-iGNG.

H. More Comparisons in Terms of HV and Sensibility Analysis

The HV values of solutions obtained by RVEA-iGNG and other compared algorithms on solving DPF, MaF and IMOP test functions are presented in the supplementary material, respectively. Additionally, the analyse of RVEA-iGNG in solving regular problems are also given in the Section III in the supplementary material. We also analyse the sensibility of parameters of the iGNG network to see whether the parameters is sensitive to different problems, which is given in Section IV in the supplementary material. In addition, the improved GNG network is also included into other decomposition based framework and the details are given in Section V in the supplementary material.

I. Comparisons between DEA-GNG and REVA-iGNG

Although DEA-GNG and RVEA-iGNG both adopt the GNG to adjust the reference vectors embedded in decomposition based algorithms, they have the following three main differences.

1) DEA-GNG uses two sets of weight vectors, one is adapted by the GNG, and the other is predefined and uniformly distributed weight vectors. These two sets of weight

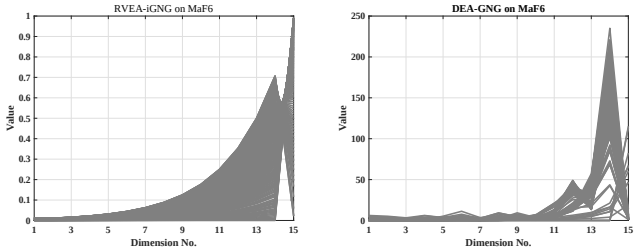


Fig. 9: The non-dominated solution set obtained by the RVEA-iGNG and DEA-GNG on solving MaF6 with 15 objectives. The set with the median IGD$^+$ value among 30 independent runs is chosen in the plots.

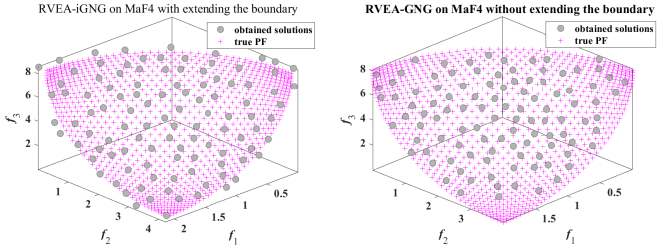


Fig. 10: The non-dominated solution set obtained by RVEA-iGNG with and without the strategy of extending the boundary on solving MaF4 with 3 objectives are given. The set with the median IGD$^+$ value among 30 independent runs is plotted.

vectors are combined to guide the search. By contrast, RVEA-iGNG uses only one set of reference vectors to guide the search, which is automatically adjusted by training the iGNG using the solutions in the current generation as well as those in the previous generations.A set of predefined reference vectors will help the search process when the PFs are regular, that is, the PFs cover the whole objective space. However, the predefined reference vectors may introduce unwanted solutions in unpromising regions since the unpromising solutions may be associated with the predefined reference vectors, which may mislead the search process. This can be further verified on the high-dimensional degenerate problems, such as MaF6. In Fig. 9, the solution set obtained by DEA-GNG is far away from the true PF, compared with the solution set obtained by RVEA-iGNG. This is caused by the introduction of predefined reference vectors. The combination of predefined reference vectors and adapted reference vectors shall boost the performance of DEA-GNG on dealing with regular problems, like MaF5, MaF10 and MaF12. However, as shown in Table SII in the Supplementary material, RVEA-iGNG can achieve better or comparable results on solving regular problems even with only one set of adapted reference vectors, which further demonstrates the effectiveness of the reference vectors trained with the iGNG.

2) The mechanisms of adjusting the nodes in the GNG are different. Node deletion in DEA-GNG is based on the hit point property defined by the authors. Every time when the hit point of a node decreases to zero, the node will be treated as dead and will be deleted. Actually, the hit point of a node will become zero quickly if it can not win in two successive generations. This mechanism does not have much difference compared with the traditional method of adapting reference vectors, since it does not utilize sufficient history information about the distribution of the solutions generated in the past generations. Different from DEA-GNG, RVEA-iGNG first increases the maximum number of the nodes in the iGNG to $1.5N$ and the nodes are ranked by the aggregated times that the age of the connections of the nodes is the same as that in the previous generations. Then $0.5N$ nodes are deleted based on the rank.

3) The framework of DEA-GNG is based on MOEA/D and the parameter θ of the scalarization function, PBI is learned the relationship between nodes. The framework of RVEA-iGNG is based on RVEA and the environmental selection is based on the APD selection criterion.

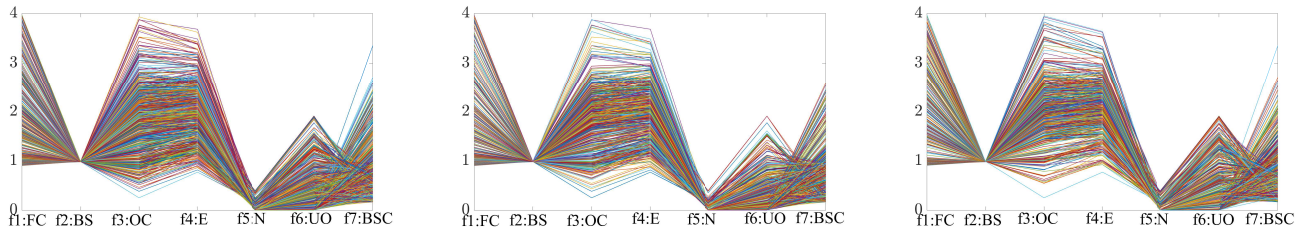
J. Effectiveness of the Boundary of Solutions

As analyzed in Section III.D, the location of the boundary nodes in GNG is usually slightly inside the true PF. Thus, we propose to extend the position of the boundary nodes according to the information of the connection between the boundary nodes and their neighbour nodes. To better show the effectiveness of our strategy for extending the boundary, we plot the solution set obtained by RVEA-iGNG with and without extending the boundary, respectively. As shown in Fig. 10, the solution set obtained by RVEA-iGNG with extending the boundary can better preserve the boundary solutions of the PFs.

K. Hybrid Electric Vehicle Controller Design

To further demonstrate the performance of the proposed RVEA-iGNG, we also test our algorithm on a real-world application, i.e., design of the controller of hybrid electric vehicles (HEVs). A detailed description of HEVs can be found in [33]. The design of HEV controller contains seven objectives and 11 design variables. The objectives are fuel consumption (FC), battery stress (BS), ICE operation changes (OC), ICE emissions (E), perceived ICE noise (N), urban operation (UO), average battery state of charge level (BSC).

We compare RVEA-iGNG and DEA-GNG for the optimization of the HEV controller design problems. The population size is set to 112 and all experiments are run for 17 times independently. As suggested in [33], the reference point for calculating HV is set to 4 for each objective. The mean HV value of solutions obtained by RVEA-iGNG and DEA-GNG are 0.3045 and 0.2921, respectively. Even though the result of RVEA-iGNG is not significantly different from that of DEA-GNG at a 0.05 level by the Wilcoxon's rank sum test, we find that RVEA-iGNG can explore more objective space than DEA-GNG and it is worth mentioning that DEA-GNG fails



(a) Approximated PF (b) solution set obtained by RVEA-iGNG (c) solution set obtained by DEA-GNG

Fig. 11: All non-dominated solutions obtained by RVEA-iGNG and DEA-GNG are adopted to approximate the PF of HEVs. All non-dominated solutions obtained by RVEA-iGNG and DEA-GNG among 17 independent runs are plotted.

once to approach the PF among 17 independent runs. All non-dominated solutions obtained by these two algorithms are combined to approximate the true PF, as shown in Fig. 11. From the figure, we can see that the solution set obtained by RVEA-iGNG cover more space than that obtained by DEA-GNG. This confirms that REVA-iGNG performs slightly better than DEA-GNG on the controller design problem.

V. Conclusion

In this paper, an adaptive reference vector guided evolutionary algorithm using an improved growing neural gas, termed as RVEA-iGNG, is proposed for many-objective optimization of irregular problems. In RVEA-iGNG, only one single set of reference vectors that are trained by an improved GNG is adopted to guide the search. When new data (new solutions) are presented to the improved GNG, not only the node that has the nearest Euclidean distance to the data (the winner) will be adjusted towards the new data, but also the direct neighbours of the winner will move towards the new data. Different from the traditional reference vector adaptation method, in which the inactive reference vectors will be replaced or deleted, the nodes in growing neural gas can be adjusted to the position of the new solutions only if there is enough number of solutions to keep the nodes moving towards the new position. By using the improved growing neural gas, the adaptation of the reference vectors can be based both on the information of the solutions in the past and in the current population. Moreover, a new effective method of adjusting the nodes in the iGNG is designed to make it suited for handling incremental data, since new non-dominated solutions are constantly generated during the search process. Experiments comparing RVEA-iGNG with eight state-of-the-art algorithms on solving MaF and DPF test functions as well as a real-world hybrid electric vehicle controller design problem demonstrate the effectiveness of the new GNG based reference vector adaptation method.

Our future work will focus on designing more effective methods for adjusting the nodes in the GNG to ensure a good balance between exploitation and exploration. Since it becomes indispensable to incorporate user preferences in solving MaOPs, new effective learning algorithms should be developed for training growing neural gas when user preferences are available. Finally, most existing work focuses on adaptation of the reference vectors without paying much attention to adapting the reproduction operators, which, however, shall also play an important role in enhancing the efficiency of MOEAs in solving irregular problems.

Acknowledgment

The authors would like to thank Fangqing Gu for providing the code of MOEA/D-SOM.

References

[1] 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.

[2] K. Ikeda, H. Kita, and S. Kobayashi, "Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal?" in *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, 2001, pp. 957–962.

[3] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 721–736, 2013.

[4] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 16–37, 2016.

[5] Y. Tian, R. Cheng, X. Zhang, Y. Su, and Y. Jin, "A strengthened dominance relation considering convergence and diversity for evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2018.

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

[7] 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.

[8] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[9] 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.

[10] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems." *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 450–455, 2014.

[11] 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.

[12] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 761–776, 2015.

[13] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 348–365, 2014.

[14] 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.

[15] 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, 2017.

[16] Y. Zhou, Y. Xiang, Z. Chen, J. He, and J. Wang, "A scalar projection and angle-based evolutionary algorithm for many-objective optimization problems," *IEEE transactions on cybernetics*, vol. 49, no. 6, pp. 2073–2084, 2018.

[17] 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.

[18] 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.

[19] 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 Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2014.

[20] ——, "An improved adaptive approach for elitist nondominated sorting genetic algorithm for many-objective optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2013, pp. 307–321.

[21] Q. Liu, Y. Jin, M. Heiderich, and T. Rodemann, "Adaptation of reference vectors for evolutionary many-objective optimization of problems with irregular Pareto fronts," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 1726–1733.

[22] H. Ge, M. Zhao, L. Sun, Z. Wang, G. Tan, Q. Zhang, and C. P. Chen, "A many-objective evolutionary algorithm with two interacting processes: Cascade clustering and reference point incremental learning," *IEEE Transactions on Evolutionary Computation*, 2018.

[23] 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, 2017.

[24] R. Denysiuk, L. Costa, and I. E. Santo, "Clustering-based selection for evolutionary many-objective optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2014, pp. 538–547.

[25] Y. Hua, Y. Jin, and K. Hao, "A clustering-based adaptive evolutionary algorithm for multiobjective optimization with irregular Pareto fronts," *IEEE Transactions on Cybernetics*, no. 99, pp. 1–13, 2018.

[26] Q. Lin, S. Liu, K.-C. Wong, M. Gong, C. A. C. Coello, J. Chen, and J. Zhang, "A clustering-based evolutionary algorithm for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, 2018.

[27] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired co-evolutionary algorithms using weight vectors," *European Journal of Operational Research*, vol. 243, no. 2, pp. 423–441, 2015.

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

[29] 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.

[30] Y. Tian, X. Zhang, R. Cheng, and Y. Jin, "A multi-objective evolutionary algorithm based on an enhanced inverted generational distance metric," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 5222–5229.

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

[32] Y. Liu, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Adapting reference vectors and scalarizing functions by growing neural gas to handle irregular Pareto fronts," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2019.

[33] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and Y. Jin, "Evolutionary many-objective optimization of hybrid electric vehicle control: from general optimization to preference articulation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, no. 2, pp. 97–111, 2017.

[34] J. A. Cornell, *Experiments with mixtures: designs, models, and the analysis of mixture data*. John Wiley & Sons, 2011, vol. 403.

[35] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 524–541, 2015.

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

[37] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[38] R. Xu and D. C. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, pp. 645–678, 2005.

[39] B. Fritzke, "A self-organizing network that can follow non-stationary distributions," in *International conference on artificial neural networks*. Springer, 1997, pp. 613–618.

[40] I. J. Sledge and J. M. Keller, "Growing neural gas for temporal clustering," in *2008 19th International Conference on Pattern Recognition*. IEEE, 2008, pp. 1–4.

[41] H. Frezza-Buet, "Following non-stationary distributions by controlling the vector quantization accuracy of a growing neural gas network," *Neurocomputing*, vol. 71, no. 7-9, pp. 1191–1202, 2008.

[42] M. Ghesmoune, M. Lebbah, and H. Azzag, "A new growing neural gas for clustering data streams," *Neural Networks*, vol. 78, pp. 36–50, 2016.

[43] R. B. Agrawal, K. Deb, and R. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[44] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," *Computer Science and Informatics*, vol. 26, pp. 30–45, 1996.

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

[46] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao, "A benchmark test suite for evolutionary many-objective optimization," *Complex & Intelligent Systems*, vol. 3, no. 1, pp. 67–81, 2017.

[47] L. Zhen, M. Li, R. Cheng, D. Peng, and X. Yao, "Multiobjective test problems with degenerate Pareto fronts," *arXiv preprint arXiv:1806.02706*, 2018.

[48] Y. Tian, C. He, R. Cheng, and X. Zhang, "A multistage evolutionary algorithm for better diversity preservation in multiobjective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.

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

[50] K. Deb and D. Saxena, "Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems," in *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, 2006, pp. 3352–3360.

[51] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary multiobjective optimization*. Springer, 2005, pp. 105–145.

[52] M. Köppen and K. Yoshida, "Substitute distance assignments in NSGA-II for handling many-objective optimization problems," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 727–741.

[53] 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.

[54] 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.

[55] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.

[56] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.