A Hybrid Surrogate-Assisted Evolutionary Algorithm for Computationally Expensive Many-Objective Optimization

Kanzhen Wan[1,2], Cheng He[1], Auraham Camacho[1,3], Ke Shang[1], Ran Cheng[1], and Hisao Ishibuchi[1]
[1]Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.
[2]School of Computer Science and Engineering, Harbin Institute of Technology, Harbin 150001, China.
[3]CINVESTAV-Tamaulipas, Tamaulipas, Mexico.
wankanzhen@hotmail.com, chenghehust@gmail.com, acamacho@tamps.cinvestav.mx, and {chengr, hisao}@sustc.edu.cn

***Abstract*—Many real-world optimization problems are challenging because the evaluation of solutions is computationally expensive. As a result, the number of function evaluations is limited. Surrogate-assisted evolutionary algorithms are promising approaches to tackle this kind of problems. However, their performance highly depends on the number of objectives. Thus, they may not be suitable for many-objective optimization. This paper proposes a novel hybrid algorithm for computationally expensive many-objective optimization, called C-M-EA. The proposed approach combines two surrogate-assisted evolutionary algorithms during the search process. We compare the performance of the proposed approach with seven multi-objective evolutionary algorithms. Our experimental results show that our approach is competitive for solving computationally expensive many-objective optimization problems.**

***Index Terms*—Surrogate-assisted evolutionary optimization, Expensive many-objective optimization, Hybrid optimization**

I. Introduction

Many real-world applications can be formulated as multi-objective optimization problems (MOPs). An MOP has multiple objective functions to be optimized simultaneously. Generally, they are conflicting. That is, any improvement to one objective may lead to a deterioration in other objectives. Consequently, any single solution cannot simultaneously optimize all objectives. When more than three objectives are involved, an MOP is called a many-objective optimization problems (MaOP). An MOP is defined as follows [1]:

$$\begin{aligned} &\text{minimize}: \ \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), ..., f_m(\mathbf{x}))^T, \\ &\text{subject to}: \ \mathbf{x} \in \Omega, \end{aligned} \tag{1}$$

where Ω is the *decision space*, $\mathbf{F} : \Omega \to \mathbb{R}^m$ consists of m real-valued objective functions, and $\mathbb{R}^m$ is the *objective space*. The objectives $f_i(\mathbf{x})$, for $i = 1, 2, \ldots, m$ in (1) are often conflicting with each other. Hence, generally there is no solution that minimizes all the objectives simultaneously.

A common approach to compare solutions in multi-objective optimization is *Pareto Dominance*. Let $\mathbf{F^1}, \mathbf{F^2} \in \mathbb{R}^m$. $\mathbf{F^1}$ dominates $\mathbf{F^2}$ if and only if $f_i^1 \leq f_i^2$ for $\forall i \in \{1, \ldots, m\}$ and $f_j^1 < f_j^2$ for $\exists j \in \{1, \ldots, m\}$. Solution $\mathbf{x}^* \in \Omega$ is *Pareto optimal* if $\mathbf{x}^*$ is not dominated by any other solution $\mathbf{x}$ in Ω (i.e., $\mathbf{F}(\mathbf{x}^*)$ is not dominated by $\mathbf{F}(\mathbf{x})$). The set of all the Pareto optimal solutions is called the *Pareto set* (PS). The projection of PS to the objective space is the *Pareto front* (PF). For an MOP, the number of Pareto optimal vectors may be large, even infinite [1]. Although, for many real-world applications, it is enough to find a representative set (i.e., a good approximation) of PF [1], [2].

The objective functions of an MOP may be multi-modal or non-continuous. Consequently, traditional derivative-based optimization methods may have difficulties in solving such an optimization problem. On the other hand, multi-/many-objective evolutionary algorithms (MOEAs or MaOEAs) are population-based, gradient-free, black-box optimization algorithms [1]. Given these properties, MOEAs have attracted a lot of attention in the evolutionary computation community.

Many real-world optimization problems are challenging because they involve computationally expensive objective functions. As a result, the number of function evaluations (FEs) is limited in practical settings. For example, several hours are needed for a single FE in aerodynamic design optimization [3]. Typically 20,000 FEs are employed for solving this problem. Therefore, given its computational burden, it will take a considerable amount of time.

A popular approach to solve computationally expensive MOPs is to use surrogate models. A surrogate model provides a computationally efficient approximation of objective functions. MOEAs can employ surrogate models to reduce the amount of time and resources related to computationally expensive MOPs. In the literature, different surrogate models have been proposed, such as radial basis function networks [4], support vector machines [3], Gaussian process models [5], and extreme learning machines [6]. Also, a number of surrogate-

This work was supported by National Natural Science Foundation of China (Grant No. 61876075), 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). Corresponding Author: Hisao Ishibuchi.

assisted evolutionary algorithms (SAEAs) have been proposed, including the decomposition-based MOEA with a Gaussian process model (MOEA/D-EGO) [5], the radial-space-division-based MaOEA with a feedforward neural network (CSEA) [7], [8], and the reference-vector-guided MaOEA with a Gaussian process model (K-RVEA) [9].

Despite their effectiveness, SAEAs face challenges. First, regarding the surrogate model, it is important to decide which surrogate model should be used and when to use it. Second, the surrogate model must be updated during the search process to achieve a better approximation of the objective functions. However, how to perform such an update is not straightforward. Third, it is critical to decide the role of the surrogate model in the search process. Commonly, a surrogate model is used to approximate the objective functions [9], whereas other approaches use it to identify good solutions from the whole population [8].

A number of different ideas have been proposed to design efficient SAEAs [3]. For example, three different SAEAs (named MOEA/D-EGO [5], MOEA/D-RBF [4], and EL-MOEA/D [6]) have been proposed based on different surrogate models in the same algorithm framework (i.e. MOEA/D-DE [10]). However, it is likely that different surrogate models may need different algorithm frameworks. Moreover, different SAEAs can be combined as a single hybrid algorithm to handle computationally expensive MOPs.

Various kinds of hybrid algorithms have been proposed for multi-objective optimization in the literature. For example, Ishibuchi and Murata [11] proposed a hybrid algorithm that combines a genetic algorithm with a modified local search algorithm, called MOGLS, for multi-objective combinatorial optimization. Ishibuchi *et al.* [12] then proposed a new version of MOGLS by modifying the selection strategy for the local search procedure. Similar to Ishibuchi's works, many hybrid MOEAs use a local search in an MOEA, also called memetic MOEAs [13], [14]. Sindhya *et al.* [15] proposed the modular hybrid algorithm framework for multi-objective optimization following the memetic approach. Many hybrid EAs have been proposed for solving conventional multi-/many-objective problems. However, only a few ideas have been proposed for solving expensive many-objective optimization. Lim *et al.* [16] proposed a surrogate-assisted memetic algorithm that uses surrogate models in the local search procedure. Pilát and Neruda [17] used different surrogate models in the local search procedure. To summarise, the previous studies mainly focus on the surrogate-assisted memetic approach that replaces the FEs in the local search procedure by the surrogate model.

In this paper, we propose a novel hybrid algorithm for computationally expensive many-objective optimization, called C-M-EA, that combines the classification-based SAEA (CSEA) and the decomposition-based SAEA (MOEA/D-EGO). Unlike the conventional combination of global search and surrogate-assisted local search, we combine two different SAEAs in a sequential way. To evaluate its behavior, we test its performance on a number of benchmark problems and two real-world problems. The rest of this paper is organized as follows.

Algorithm 1 CSEA

```
Input: $K$ (the number of reference solutions), $FE_{max}$, $N$
    (population size)
Output: $Arc$ (Archive of all evaluated solutions)
    /* Initialization */
 1: Generate the initial population $P$ of $N$ solutions by the
    Latin hypercube sampling
 2: $FE \leftarrow N$
 3: Initialize the feedforward neural network $FNN$
 4: $Arc \leftarrow P$
 5: while $FE \leq FE_{max}$ do
      /* select reference solutions $RF$ */
 6:   $RF \leftarrow$ ReferenceSelection($P, K$)
      /* train the FNN classifier */
 7:   $[Data_{train}, Data_{test}] \leftarrow$ DataDivision($P$)
 8:   $FNN \leftarrow$ Train($FNN, Data_{train}$)
 9:   $error \leftarrow$ Validation($FNN, Data_{test}$)
      /* select new solutions ($New$) to evaluate */
10:   $New \leftarrow$ SurrogateSelection($FNN, RF, P, error$)
11:   $Arc \leftarrow Arc \cup New$
      /* select the next population */
12:   $P \leftarrow$ ReferenceSelection($P \cup New, N$)
13:   $FE \leftarrow FE + |New|$
14: end while
```

In Section II, we give a brief introduction to the two adopted algorithms, CSEA and MOEA/D-EGO. In Section III, the details of the proposed algorithm are given. In Section IV, its search ability is examined through computational experiments. Finally, Section V concludes the paper.

II. Background

A. CSEA

CSEA [8] is a recently proposed SAEA for computationally expensive many-objective optimization. It selects a few reference solutions (RSs) to train a feedforward neural network (FNN) as a classifier. Then, this classifier is employed to identify solutions that are dominated by RSs. CSEA can effectively accelerate the convergence of solutions, but cannot maintain their good diversity sometimes on expensive MOPs [8]. The framework of CSEA is given in Algorithm 1. Its main components are described as follows:

1) Selection of Reference Solutions: CSEA uses a radial-projection-based environmental selection to identify the reference solutions and to construct the next generation of solutions. An archive (Arc) is used to store all the evaluated solutions. The solutions in Arc are projected into a 2-dimensional radial space. After the projection, a prefixed number of solutions are selected considering both convergence and diversity.

2) Construction of the Surrogate Model: After selecting the RSs, the solutions in Arc can be divided into two classes, i.e., the dominated class (negative) and the nondominated class (positive). Next, an FNN classifier is trained to model the dominance relation.

```
Algorithm 2 MOEA/D-EGO
Input: $N$ (population size), $FE_{max}$, weight vectors ($W = \{\lambda_i | i = 1, \cdots, N\}$)
Output: $Arc$ (Archive of all evaluated solutions)
    /* Initialization */
 1: Generate the initial population $P$ of $N$ solutions by the Latin hypercube sampling
 2: $FE \leftarrow N$
 3: while $FE \leq FE_{max}$ do
    /* Build Gaussian process models*/
 4:    $clusters \leftarrow$ FCM($P$)
 5:    $GPs \leftarrow$ Train($clusters$)
    /* Select new solutions ($New$) to evaluate */
 6:    $New \leftarrow$ SurrogateSelection($GPs, P, W$)
 7:    $P \leftarrow P \cup New$
 8:    $FE \leftarrow FE + |New|$
 9: end while
10: $Arc \leftarrow P$
```

3) Selection of Solutions for Real Function Evaluation: Offspring are generated by means of crossover and mutation. The trained FNN is used to select the potentially good solutions that are not dominated by the reference solutions from the offspring. The procedure is repeated until a given number of solutions for real function evaluations are selected.

B. MOEA/D-EGO

MOEA/D-EGO [5] combines differential evolution (DE) and a Gaussian process model. DE is used as a stochastic operator to generate new solutions [18]. A Gaussian process model is employed to approximate the objective functions and guide the selection of solutions for real function evaluation. MOEA/D-EGO has a high diversification ability, but cannot obtain a set of well-converged solutions on expensive multi-modal MOPs. The framework of MOEA/D-EGO is given in Algorithm 2. There are two main components:

1) Construction of the Surrogate Model: First, MOEA/D-EGO uses the fuzzy C-means (FCM) algorithm [19] to divide the solutions into different clusters. Then, a Gaussian process model is constructed for each cluster.

2) Selection of Solutions for Real Function Evaluation: New solutions are generated following the same procedure as in MOEA/D-DE [10]. After decomposing a problem, the subproblems are optimized collaboratively with the assistance of the Gaussian process models. Finally, MOEA/D-EGO selects potentially good solutions. The procedure is repeated until a given number of solutions are selected.

III. The Proposed Algorithm

Whereas the diversification ability of MOEA/D-EGO is high, its convergence ability to efficiently find good solutions is not always high on expensive multi-modal MOPs [10]. In contrast, the convergence ability of CSEA is high, but its diversification ability to maintain a set of well-distributed solutions is not always high [8]. Our approach, called C-M-EA, combines CSEA and MOEA/D-EGO in a single framework to accelerate the convergence of solutions and maintain their good diversity. Figure 1 illustrates the main framework of the proposed approach.

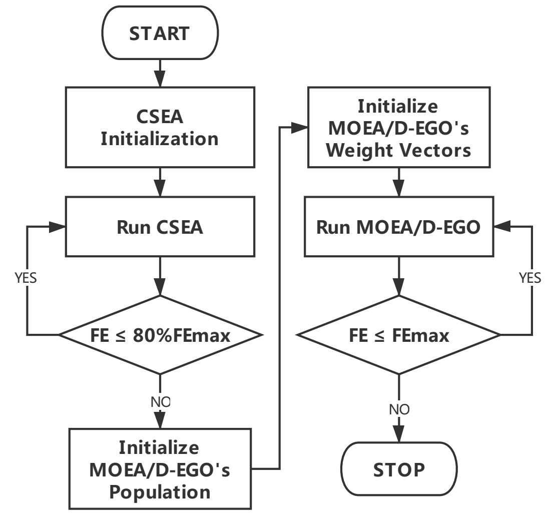

Fig. 1. The framework of C-M-EA.

At the beginning, we use CSEA to effectively push the population close to the true PF. Before switching to MOEA/D-EGO, we select potentially good solutions from CSEA's archive as the initial population of MOEA/D-EGO. Afterwards, with the assistance of the selected population, we initialize the weight vectors of MOEA/D-EGO. The framework of C-M-EA is given in Algorithm 3. The main components are described as follows:

1) Initialization: C-M-EA uses the Latin hypercube sampling [20] to generate an initial population P with $11d - 1$ solutions, where d is the number of decision variables. An archive (Arc) is used to store all the evaluated solutions. After evaluations, the solutions are stored in the archive (Arc).

2) Execution of CSEA (Algorithm 1): C-M-EA adopts CSEA at the beginning of the optimization. In this paper, 80% of the FEs are allocated to CSEA.

3) Population Initialization for MOEA/D-EGO: C-M-EA selects the initial population for MOEA/D-EGO by the same environmental selection strategy in CSEA.

4) Weight Vector Initialization for MOEA/D-EGO: Based on the selected initial population, C-M-EA initializes the weight vectors in MOEA/D-EGO. In this paper, 20% of the FEs are allocated to MOEA/D-EGO.

5) Execution of MOEA/D-EGO (Algorithm 2): C-M-EA runs MOEA/D-EGO with the initialized population and weight vectors at the final stage of optimization.

In C-M-EA, the strategy to perform the environmental selection is the same as the selection strategy in CSEA. The strategy to initialize the weight vectors for MOEA/D-EGO is a critical component. We will give more details of this strategy.

Algorithm 3 C-M-EA

```
Input: $N$ (population size), $FE_{max}$, $K$ (number of reference solutions)
Output: $Arc$ (Archive of all evaluated solutions)
    /* Initialization */
 1: Generate the initial population $P$ of $N$ solutions by the Latin hypercube sampling
 2: $FE \leftarrow N$
 3: Initialize the feedforward neural network $FNN$
 4: $Arc \leftarrow P$
 5: $Flag \leftarrow \textbf{TRUE}$
 6: while $FE \leq FE_{max}$ do
 7:     if $FE/FE_{max} \leq 0.8$ then
    /* Run CSEA from beginning to middle stage */
 8:         $Arc \leftarrow$ CSEA($P, K, 0.8FE_{max}, N$)
 9:         $FE \leftarrow |Arc|$
10:     else
11:         if $Flag$ then
    /* Initialize population and weight vectors */
12:             $P \leftarrow$ ReferenceSelection($Arc, N$)
13:             $W \leftarrow$ WeightVectorInitialization($P$)
14:             $Flag \leftarrow \textbf{FALSE}$
15:         end if
    /* Run MOEA/D-EGO at final stage */
16:         $Arc \leftarrow$ MOEA/D-EGO($P, 0.2FE_{max}, N, W$)
17:         $FE \leftarrow |Arc|$
    /* Select the next population */
18:         $P \leftarrow$ ReferenceSelection($Arc, N$)
19:     end if
20: end while
```

A. Weight Vector Initialization for MOEA/D-EGO

After decomposing a problem, MOEA/D-DE solves the subproblems collaboratively by employing a fixed set of weight vectors. Furthermore, all the subproblems are treated equally using the same computational load. However, when the PF is complex or irregular, the uniformly distributed weight vectors may not guide the search effectively. Thus, the performance of the algorithms can be deteriorated [21].

In order to enhance the diversity maintenance ability of decomposition-based methods, different weight adaptation mechanisms have been proposed [22]–[25]. These mechanisms adapt the weight vectors according to the shape of the PF in order to get a better result. Such an adaptation is performed frequently during the search process. However, these mechanisms may face difficulties when the number of FEs is limited. Given this, we propose to adjust the weight vectors at the beginning of MOEA/D-EGO's execution only. In this way, MOEA/D-EGO can employ the adaptive weight vectors to solve the expensive MOPs. Inspired by [24], we propose to initialize the weight vectors as follows:

1) Weight Vector Generation: We generate the weight vector candidates by the simplex-lattice method [26], [27]. The number of vectors is about $2^m|P|$, where m is the number of objectives and $|P|$ is the population size.

2) Active Weight Vector Selection: Active selection [9] is an angle-based method for selecting weight vectors. First, every solution in the population is assigned to the nearest weight vector. A weight vector is considered active in this population if at least one solution in the population is assigned to it. Otherwise, it is considered inactive. Here we select all active weight vectors.

3) Inactive Weight Vector Selection: The remaining inactive weight vectors are clustered into m^2 groups by the k-medoids method [28] and then we select their cluster centers as part of the weight vectors.

4) Weight Vector Generation: If the total number of the selected weight vectors is smaller than the population size, we randomly generate weight vectors until the amount of weight vectors is equal to the population size. If the total number of the selected weight vectors is larger than the population size, we randomly remove some weight vectors from the set of inactive weight vectors until the amount of weight vectors is equal to the population size.

B. Condition for Switching Algorithms

It is critical to decide when to switch from CSEA to MOEA/D-EGO. We empirically study this parameter by testing C-M-EA's performance on several problems with different settings of the switching time. The switching time is formulated as the ratio of the number of FEs to the maximum number of FEs (FE/FE_{max}). We have adopted the inverted generational distance (IGD) [29] as a performance indicator. A smaller IGD value indicates a better performance. Figure 2 shows the obtained mean IGD values on DTLZ1, DTLZ2, DTLZ4, DTLZ5, DTLZ7, and IDTLZ2 with 6, 8, 10 objectives over 60 independent runs. The selected problems have different properties. DTLZ1 is a multi-modal MOP. DTLZ2 is a problem for testing the diversity maintenance ability. DTLZ4 is a variation of DTLZ2 with biased regions. DTLZ5 has a degenerate PF. DTLZ7 has a disconnected PF that is difficult to approximate using a set of uniformly distributed solutions. IDTLZ2 is also a variation of DTLZ2 with an inverted PF.

In the experimental results on DTLZ1 and DTLZ5 in Fig. 2, the mean IGD value decreases (i.e., the performance improves) as FE/FE_{max} increases. For DTLZ2 and DTLZ7 in Fig. 2, it increases (i.e., degrades) as FE/FE_{max} increases. Whereas the best specification of FE/FE_{max} is problem-dependent, good results are obtained over a wide range of its values from 50% to 90% in Fig. 2. In Section IV, we report our experimental results with $FE/FE_{max} = 80\%$. Similar results are obtained from other settings (e.g., $FE/FE_{max} = 70\%$).

IV. Numerical Experiments

In this section, we compare C-M-EA with some state-of-the-art algorithms, including NSGA-III [26], MOEA/DD [30], RSEA [31], ParEGO [32], MOEA/D-EGO [5], K-RVEA [9], and CSEA [8]. Among these algorithms, ParEGO, MOEA/D-EGO, K-RVEA, and CSEA are SAEAs. We use three benchmark problems: DTLZ, IDTLZ, and MaF with 6, 8, 10

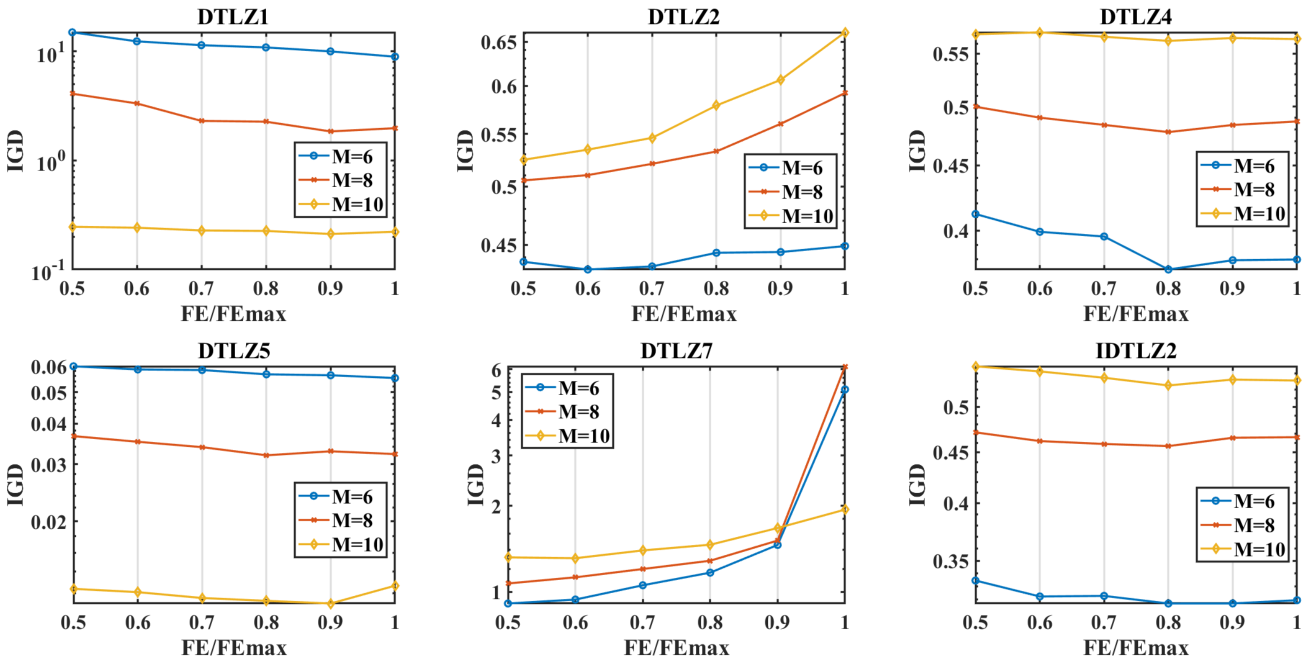

Fig. 2. The obtained mean IGD values of benchmark problems at different time to switch.

objectives. The experiments are also conducted on two real-world problems: the car cab design problem (CCD) [33] and the reactive power optimization problem (IEEE14) [34]. Since a single run of ParEGO takes a considerable amount of time, ParEGO is applied only to the two real-world problems. Our experiments are conducted on the platform, PlatEMO [35].

In our experiments, the number of decision variables is set to ten for all the benchmark problems. Besides, we define a small number of FEs to evaluate the behavior of an MOEA when the number of FEs is limited. The maximum number of FEs is set to 600 for all the problems, including the benchmark problems and the real-world problems.

For the benchmark problems, the IGD is used as a performance indicator, which is suggested in [8]. For the real-world problems, the hypervolume (HV) [36] is adopted. A greater HV value indicates a better performance. Since CCD has nine objectives, the Monte Carlo method is employed with 1,000,000 sampling points to estimate the HV [35].

We use the Wilcoxon rank sum test to compare the adopted algorithms at a significance level of 0.05 over 30 independent runs. In the following tables, "$+$" means that the compared algorithm significantly outperforms C-M-EA, whereas "$-$" means that C-M-EA significantly outperforms the compared algorithm. Finally, "$\approx$" means that there is no significant difference between the compared algorithms.

A. *Experimental Settings*

The specific parameters for each algorithm are set to the recommended settings in the literature. The details of the general parameter settings are given below.

1) Reproduction Operators: The parameters of DE are set to $CR = 1$, $F = 0.5$, and $p_m = 1/d$ (the mutation probability) where d is the number of decision variables. The simulated binary crossover and the polynomial mutation are also employed for the offspring generation. The parameters of them are set to $p_m = 1/d$, $p_c = 1$ (the crossover probability), $n_c = 20$ (the distribution index in the crossover), and $n_m = 20$ (the distribution index in the mutation).

2) Population Size: For fair comparison, we use a similar population size for each algorithm. The population size of CSEA, ParEGO, and RSEA is set to 50. Since the population size of NSGA-III, MOEA/D-EGO, K-RVEA, and MOEA/DD cannot be arbitrarily specified and is given by the parameters (p_1 and p_2) [26], the population size is not precisely set to 50. Table I shows the settings of the two parameters and the corresponding population size in each algorithm.

B. *Performance on the Benchmark Problems*

Table II summarizes the average IGD values obtained by the six algorithms over 30 independent runs. It can be observed that C-M-EA performed best among the compared algorithms on average. DTLZ1, DTLZ3, IDTLZ1, and MaF1 are multi-modal problems. It is difficult to search for good solutions of those problems, especially when the number of FEs is

TABLE I
THE POPULATION SIZE IN NSGA-III, MOEA/D-EGO, K-RVEA, AND MOEA/DD, WHERE M IS THE NUMBER OF OBJECTIVES, AND (p_1, p_2) ARE PARAMETERS CONTROLLING THE POPULATION SIZE.

M	(p_1, p_2)	Population size
3	(8, 0)	45
6	(2, 2)	42
8	(2, 1)	44
9	(2, 1)	54
10	(2, 0)	55

limited. The experimental results show that C-M-EA is ranked second among all the compared algorithms on these problems, which means that it inherits the high convergence ability from CSEA. On the other hand, for DTLZ2, DTLZ4, IDTLZ2, and MaF2, high convergence ability is not needed whereas high diversification ability is needed. The experimental results show that C-M-EA is the best on DTLZ4, IDTLZ2, and MaF2, which indicates that C-M-EA inherits high diversity maintenance ability from MOEA/D-EGO.

DTLZ5 and DTLZ6 are problems with degenerate PFs. The experimental results show that, on DTLZ5, C-M-EA has achieved better results than the other three algorithms (i.e., MOEA/DD, MOEA/D-EGO, and RSEA) and competitive experimental results compared to CSEA. On DTLZ6, C-M-EA show competitive results to MOEA/D-EGO, and better results than the other three algorithms. DTLZ7 is a problem with a disconnected PF. The experimental results show that the performance of K-RVEA is the best, followed by MOEA/D-EGO and C-M-EA. The experimental results also show that K-RVEA is the best on DTLZ5, DTLZ6, and DTLZ7, which indicates that K-RVEA may be effective in solving problems with degenerate or disconnected PFs. Besides, C-M-EA almost outperformed all the conventional MOEAs on all the benchmark problems, which shows the effectiveness of the surrogate models.

C. Performance on the Real-world Problems

We also tested the performance of our proposed algorithm on two real-world problems. The first one is a vehicle performance optimization problem, called car cab design (CCD), which has 11 decision variables and 9 objectives. The formal definition and details of this problem are given in [26]. The second one is a reactive power optimization problem with 14 buses, called IEEE14, which has 11 decision variables and 3 objectives. The formal definition and details of this problem are given in [34].

Table III presents the average HV values obtained by the six algorithms over 30 independent runs. C-M-EA performed better than NSGA-III, ParEGO, MOEA/D-EGO and K-RVEA on the two real-world problems, and achieved competitive performance compared to CSEA. Figure 3 depicts the PFs obtained by a single run of each algorithm. The single run is with the median HV values on CCD. From Table III and Fig. 3, we observe that the solutions obtained by C-M-EA have maintained a larger diversity than CSEA.

V. CONCLUSION

In this paper, a hybrid SAEA (called C-M-EA) has been proposed for computationally expensive many-objective optimization. Our approach combines CSEA and MOEA/D-EGO in a sequential way. Also, a weight vector initialization method has been proposed for MOEA/D-EGO.

The performance of C-M-EA has been examined on a wide range of benchmark problems and two real-world problems. The overall performance of C-M-EA is highly competitive for expensive MaOP. The experimental results also suggest that C-M-EA is a good choice for problems that are not only hard to converge but also difficult for diversity maintenance.

The following are future research directions.

1) Analysis of Algorithms: In C-M-EA, CSEA and MOEA/D-EGO are adopted as the two main components. They can be replaced by other algorithms.

2) Initialization of Weight Vectors: In this paper, we proposed a strategy to initialize weight vectors. However, this paper mainly focuses on the hybridization of different algorithms. The effectiveness of this strategy (and other possible strategies) need to be further examined.

3) Adaptive Switching Strategy: In C-M-EA, the time to switch algorithms from CSEA to MOEA/D-EGO is prefixed. Since the best timing of switching seems to be problem dependent from Fig. 2, the development of an adaptive mechanism is needed to adjust the timing of switching based on the progress of multi-objective evolution. For instance, if the current main difficulty is the convergence, we postpone the time to switch to MOEA/D-EGO. On the contrary, if the current main difficulty is the diversity, C-M-EA switches to MOEA/D-EGO earlier.

ACKNOWLEDGMENT

The authors would like to thank L. M. Pang, Y. Liu, and C. Hu for their comments on the paper.

REFERENCES

[1] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *2008 IEEE Congress on Evolutionary Computation, CEC 2008*. IEEE, 2008, pp. 2419–2426.

[2] Y. Liu, D. Gong, X. Sun, and Y. Zhang, "Many-objective evolutionary optimization based on reference points," *Applied Soft Computing*, vol. 50, pp. 344–355, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494616305786

[3] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, "A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms," *Soft Computing*, pp. 1–30, 2017.

[4] S. Zapotecas Martínez and C. A. Coello Coello, "MOEA/D assisted by rbf networks for expensive multi-objective optimization problems," in *2013 Proceedings of the Genetic and evolutionary computation conference, GECCO 2013*. ACM, 2013, p. 1405.

[5] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with gaussian process model," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 3, pp. 456–474, 2010.

[6] L. M. Pavelski, M. R. Delgado, C. P. De Almeida, R. A. Gonçalves, and S. M. Venske, "ELMOEA/D-DE: Extreme learning surrogate models in multi-objective optimization based on decomposition and differential evolution," in *2014 Brazilian Conference on Intelligent Systems, BRACIS 2014*, 2014, pp. 318–323.

TABLE II
STATISTIC RESULTS OF IGD VALUES ON THE BENCHMARK PROBLEMS.

Problem	M	K-RVEA	MOEA/DD	MOEA/D-EGO	RSEA	CSEA	C-M-EA
DTLZ1	6	2.2675e+1 (7.37e+0) −	1.0689e+1 (3.74e+0) ≈	2.7597e+1 (5.48e+0) −	1.5752e+1 (3.82e+0) −	1.0538e+1 (3.31e+0) ≈	1.0870e+1 (3.90e+0)
DTLZ1	8	7.8430e+0 (3.75e+0) −	2.1760e+0 (1.05e+0) ≈	9.2340e+0 (4.07e+0) −	4.0250e+0 (1.91e+0) −	1.9104e+0 (9.42e-1) ≈	2.1476e+0 (1.45e+0)
DTLZ1	10	2.4679e-1 (4.69e-2) ≈	4.3607e-1 (3.66e-1) −	3.2950e-1 (8.12e-2) −	3.0661e-1 (1.03e-1) −	2.2414e-1 (6.12e-2) ≈	2.2799e-1 (5.27e-2)
DTLZ2	6	3.0259e-1 (1.19e-2)+	4.0056e-1 (2.77e-2) +	4.2335e-1 (2.24e-2) +	4.1585e-1 (3.20e-2) +	4.4862e-1 (3.29e-2) ≈	4.3646e-1 (2.24e-2)
DTLZ2	8	4.1533e-1 (1.25e-2) +	5.3452e-1 (3.78e-2) ≈	4.7684e-1 (2.04e-2) +	5.6865e-1 (3.74e-2) −	5.8685e-1 (3.50e-2) −	5.3284e-1 (2.52e-2)
DTLZ2	10	5.0116e-1 (1.02e-2) +	6.8640e-1 (6.50e-2) −	4.6295e-1 (1.50e-2) +	6.4932e-1 (4.81e-2) −	6.5176e-1 (2.73e-2) −	5.8936e-1 (3.76e-2)
DTLZ3	6	6.8135e+1 (1.97e+1) −	4.1114e+1 (1.45e+1) ≈	8.2533e+1 (2.40e+1) −	5.3280e+1 (1.63e+1) −	3.0593e+1 (1.00e+1) +	3.9888e+1 (1.44e+1)
DTLZ3	8	2.0583e+1 (8.60e+0) −	1.0760e+1 (5.61e+0) ≈	2.8787e+1 (1.14e+1) −	1.3205e+1 (6.75e+0) ≈	7.8804e+0 (5.97e+0) ≈	1.0137e+1 (6.46e+0)
DTLZ3	10	8.8987e-1 (2.11e-1) ≈	1.8926e+0 (1.76e+0) −	1.1436e+0 (3.72e-1) −	1.1985e+0 (6.64e-1) −	9.6806e-1 (2.26e-1) ≈	8.6837e-1 (9.66e-2)
DTLZ4	6	3.7666e-1 (4.89e-2) ≈	6.5706e-1 (1.48e-1) −	6.4378e-1 (2.36e-2) −	5.0017e-1 (5.26e-2) −	3.7878e-1 (4.75e-2) ≈	3.6564e-1 (3.84e-2)
DTLZ4	8	4.5566e-1 (2.83e-2) ≈	7.0952e-1 (9.45e-2) −	6.1188e-1 (1.39e-2) −	5.9482e-1 (4.68e-2) −	4.7929e-1 (4.15e-2) ≈	4.6970e-1 (3.29e-2)
DTLZ4	10	5.6962e-1 (2.22e-2) ≈	8.9908e-1 (1.02e-1) −	6.0158e-1 (1.37e-2) −	6.6097e-1 (4.35e-2) −	5.7095e-1 (3.19e-2) ≈	5.6716e-1 (2.35e-2)
DTLZ5	6	1.9820e-2 (4.82e-3) +	1.7638e-1 (5.32e-2) −	1.5205e-1 (1.68e-2) −	9.6652e-2 (2.84e-2) −	5.3002e-2 (1.32e-2) ≈	5.9266e-2 (1.53e-2)
DTLZ5	8	1.5759e-2 (4.57e-3) +	1.0350e-1 (1.16e-2) −	7.4702e-2 (1.04e-2) −	1.5326e-1 (7.97e-2) −	3.2208e-2 (5.87e-3) ≈	3.0834e-2 (5.30e-3)
DTLZ5	10	9.7846e-3 (1.35e-3) +	2.1214e-1 (1.29e-2) −	1.9025e-2 (2.14e-3) −	3.9785e-1 (2.22e-1) −	1.2087e-2 (1.38e-3) ≈	1.1894e-2 (1.14e-3)
DTLZ6	6	7.4134e-1 (2.15e-1) +	2.8372e+0 (6.98e-1) −	7.6500e-1 (3.56e-1) +	2.8216e+0 (3.34e-1) −	2.0894e+0 (6.52e-1) −	1.1059e+0 (4.55e-1)
DTLZ6	8	2.6553e-1 (1.13e-1) +	1.3926e+0 (6.04e-1) −	3.3200e-1 (2.16e-1) ≈	1.3713e+0 (4.30e-1) −	7.0472e-1 (4.21e-1) −	4.3834e-1 (2.82e-1)
DTLZ6	10	4.7362e-2 (2.57e-2) −	5.1222e-1 (3.15e-1) −	6.6093e-2 (4.44e-2) −	2.6549e-1 (1.79e-1) −	2.8462e-2 (1.53e-2) ≈	3.2245e-2 (1.52e-2)
DTLZ7	6	3.5457e-1 (1.99e-2) +	1.5774e+0 (8.57e-1) −	7.4134e-1 (3.95e-2) +	6.4413e+0 (1.89e+0) −	5.2445e+0 (1.74e+0) −	1.1669e+0 (2.58e-1)
DTLZ7	8	6.7167e-1 (2.43e-2) +	1.3474e+0 (3.00e-1) ≈	9.3159e-1 (3.25e-2) +	6.8987e+0 (2.28e+0) −	6.3347e+0 (2.39e+0) −	1.2925e+0 (2.34e-1)
DTLZ7	10	8.7714e-1 (2.51e-2) +	2.4111e+0 (4.27e-1) −	1.1039e+0 (3.05e-2) +	5.6342e+0 (2.76e+0) −	1.9654e+0 (3.65e-1) −	1.4887e+0 (2.42e-1)
IDTLZ1	6	3.3424e+1 (1.39e+1) ≈	3.5304e+1 (1.31e+1) ≈	6.9838e+1 (3.82e+1) −	2.9638e+1 (1.23e+1) ≈	3.0812e+1 (1.53e+1) ≈	2.7642e+1 (1.58e+1)
IDTLZ1	8	1.7404e+1 (9.97e+0) −	1.1502e+1 (7.10e+0) ≈	2.6642e+1 (1.43e+1) −	7.7465e+0 (4.41e+0) ≈	3.8718e+0 (2.82e+0) +	9.3818e+0 (6.49e+0)
IDTLZ1	10	2.7633e-1 (3.29e-1) ≈	1.5443e+0 (2.07e+0) −	4.3263e-1 (3.25e-1) −	5.3393e-1 (5.15e-1) −	3.7490e-1 (4.93e-1) −	2.7440e-1 (2.56e-1)
IDTLZ2	6	3.3032e-1 (9.75e-3) −	4.8149e-1 (1.18e-2) −	4.7039e-1 (2.64e-2) −	3.9752e-1 (1.69e-2) −	3.2163e-1 (2.54e-2) ≈	3.1248e-1 (2.23e-2)
IDTLZ2	8	5.3177e-1 (3.84e-2) −	7.1765e-1 (4.28e-2) −	5.7290e-1 (3.67e-2) −	5.4049e-1 (2.20e-2) −	4.6411e-1 (2.63e-2) ≈	4.5421e-1 (2.72e-2)
IDTLZ2	10	6.6364e-1 (2.90e-2) −	8.6063e-1 (4.77e-2) −	6.1510e-1 (3.89e-2) −	6.1666e-1 (1.75e-2) −	5.4005e-1 (2.77e-2) ≈	5.2747e-1 (2.45e-2)
MaF1	6	2.3039e-1 (2.58e-2) −	3.7760e-1 (7.18e-2) −	3.9413e-1 (4.36e-2) −	2.3721e-1 (1.02e-2) −	1.8771e-1 (1.08e-2) ≈	1.9082e-1 (1.43e-2)
MaF1	8	3.3605e-1 (2.64e-2) −	5.9892e-1 (5.76e-2) −	3.5511e-1 (2.95e-2) −	2.9481e-1 (1.89e-2) −	2.1506e-1 (9.95e-3) ≈	2.1716e-1 (1.09e-2)
MaF1	10	4.0183e-1 (2.28e-2) −	6.7400e-1 (4.25e-2) −	3.4840e-1 (2.79e-2) −	3.1538e-1 (2.13e-2) −	2.2043e-1 (6.74e-3) ≈	2.2247e-1 (9.00e-3)
MaF2	6	1.7794e-1 (1.32e-2) −	3.1660e-1 (9.68e-2) −	1.6888e-1 (2.14e-2) ≈	3.1938e-1 (4.57e-2) −	2.1413e-1 (1.47e-2) −	1.6468e-1 (2.18e-2)
MaF2	8	2.4169e-1 (2.17e-2) −	4.2286e-1 (7.31e-2) −	2.6116e-1 (2.38e-2) −	4.0189e-1 (4.49e-2) −	2.9289e-1 (1.78e-2) −	2.2847e-1 (2.92e-2)
MaF2	10	3.0522e-1 (2.17e-2) −	5.1413e-1 (8.49e-2) −	3.0352e-1 (2.90e-2) −	4.4295e-1 (5.15e-2) −	3.3723e-1 (2.76e-2) −	2.7664e-1 (3.51e-2)
+/ − / ≈		11/15/7	1/24/8	7/24/2	1/29/3	2/11/20	

TABLE III
STATISTIC RESULTS OF HV VALUES ON THE REAL-WORLD PROBLEMS.

Problem	NSGA-III	ParEGO	MOEA/D-EGO	CSEA	K-RVEA	C-M-EA
CCD	3.2085e-3 (5.12e-4) −	4.6013e-3 (1.46e-4) −	4.3622e-3 (1.52e-4) −	5.7304e-3 (6.28e-4) +	2.1033e-3 (3.27e-4) −	4.8731e-3 (2.41e-4)
IEEE14	3.3142e-2 (6.94e-4) −	3.3719e-2 (3.93e-4) −	3.3114e-2 (6.65e-4) −	3.4592e-2 (5.03e-4) −	3.5266e-2 (2.71e-4) ≈	3.5289e-2 (5.27e-4)
+/ − / ≈	0/2/0	0/2/0	0/2/0	1/1/0	0/1/1	

[7] C. He, Y. Tian, Y. Jin, X. Zhang, and L. Pan, "A radial space division based evolutionary algorithm for many-objective optimization," *Applied Soft Computing Journal*, vol. 61, pp. 603–621, 2017.

[8] L. Pan, C. He, C. He, Y. Tian, H. Wang, X. Zhang, and Y. Jin, "A classification based surrogate-assisted evolutionary algorithm for expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, no. c, 2018.

[9] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 129–142, 2018.

[10] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/ D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.

[11] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 28, no. 3, pp. 392–403, 1998.

[12] 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.

[13] Q. H. Nguyen, Y. Ong, and M. H. Lim, "A probabilistic memetic framework," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 604–623, 2009.

[14] B. Liu, L. Wang, and Y. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 18–27, 2007.

[15] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 4, pp. 495–511, 2013.

[16] D. Lim, Y. Jin, Y. Ong, and B. Sendhoff, "Generalizing Surrogate-Assisted Evolutionary Computation," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 3, pp. 329–355, 2010.

[17] M. Pilát and R. Neruda, "ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model," in *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, 2011, pp. 1202–1208.

[18] R. Storn and K. Price, "Differential Evolution – A simple and efficient

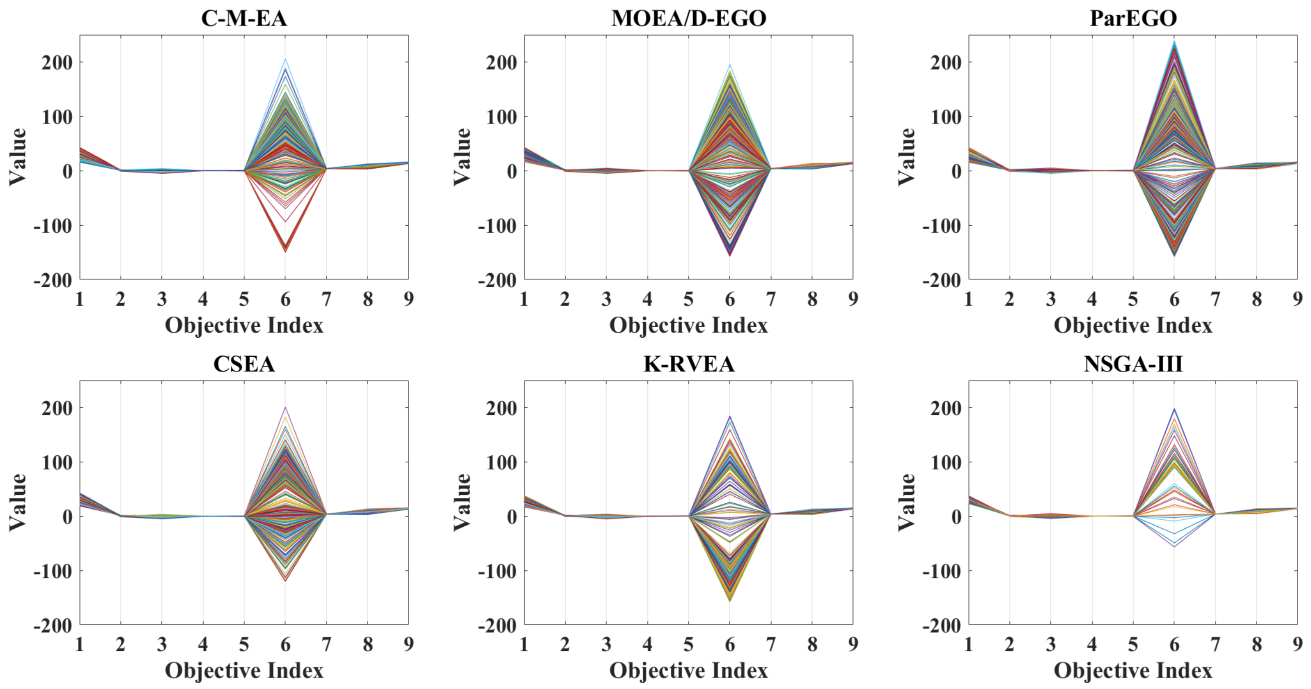

Fig. 3. The obtained PFs with the median HV value on CCD.

heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997. [Online]. Available: https://doi.org/10.1023/A:1008202821328

[19] J. C. Bezdek, R. J. Hathaway, M. J. Sabin, and W. T. Tucker, "Convergence theory for fuzzy c-means: Counterexamples and repairs," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, no. 5, pp. 873–877, 1987.

[20] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, no. 1, pp. 55–61, 2000.

[21] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh, "A survey of multiobjective evolutionary algorithms based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 440–462, 2017.

[22] L. R. C. de Farias, P. H. M. Braga, H. F. Bassani, and A. F. R. Araújo, "MOEA/D with uniformly randomly adaptive weights," *2018 Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018*, pp. 641–648, 2018.

[23] M. Wu, S. Kwong, Y. Jia, K. Li, and Q. Zhang, "Adaptive weights generation for decomposition-based multi-objective optimization using Gaussian process regression," in *2017 Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2017*. ACM, 2017, pp. 641–648.

[24] S. Jiang, Z. Cai, J. Zhang, and Y. S. Ong, "Multiobjective optimization by decomposition with Pareto-adaptive weight vectors," in *2011 the International Conference on Natural Computation, ICNC 2011*, vol. 3, 2011, pp. 1260–1264.

[25] X. Guo, X. Wang, and Z. Wei, "MOEA/D with adaptive weight vector design," in *2015 the International Conference on Computational Intelligence and Security, CIS 2015*, 2016, pp. 291–294.

[26] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[27] 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.

[28] H. S. Park and C. H. Jun, "A simple and fast algorithm for K-medoids clustering," *Expert Systems with Applications*, vol. 36, no. 2 PART 2, pp. 3336–3341, 2009.

[29] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[30] 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.

[31] 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.

[32] J. Knowles, "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[33] K. Deb, S. Gupta, D. Daum, J. Branke, A. K. Mall, and D. Padmanabhan, "Reliability-based optimization using evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1054–1074, 2009.

[34] R. Kavasseri and S. K. Srinivasan, "Joint placement of phasor and power flow measurements for observability of power systems," *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 1929–1936, 2011.

[35] 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.

[36] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.