A Cone Decomposition Evolutionary Algorithm with Dominance-based Archive for Many-objective Optimization Problems with Irregular Fronts

1st Weiqin Ying
School of Software Engineering
South China University of Technology
Guangzhou 510006, PR China

2nd Junjie Huang
School of Software Engineering
South China University of Technology
Guangzhou 510006, PR China

3rd Yu Wu*
School of Computer Science and Cyber Eng.
Guangzhou University
Guangzhou 510006, PR China
wuyu@gzhu.edu.cn (Corresponding author)

4th Yali Deng
School of Software Engineering
South China University of Technology
Guangzhou 510006, PR China

5th Yanqi Lan
School of Software Engineering
South China University of Technology
Guangzhou 510006, PR China

***Abstract*—Most of the existing multi-objective evolutionary algorithms (MOEAs) are designed to solve many-objective optimization problems (MaOPs) with regular fronts. However, their effectiveness for MaOPs with irregular fronts are yet to be improved. In this paper, a cone decomposition evolutionary algorithm with dominance-based archive (CDEA-DA) is presented to extend decomposition-based MOEAs for MaOPs with irregular fronts. In CDEA-DA, an improved cone decomposition strategy is adopted to decompose one MaOP into several scalar subproblems. Then, a dominance-based archive is designed to collect the non-dominated solutions eliminated during evolution, so as to improve the quality of the obtained front. The proposed algorithm is compared with four state-of-the-art algorithms on unconstrained benchmark MaOPs. Empirical results demonstrate that CDEA-DA achieves the superior quality of fronts.**

***Index Terms*—Many-objective optimization, irregular fronts, evolutionary algorithms, decomposition, dominance**

I. Introduction

In scientific research and engineering applications, many problems involve optimizing multiple objectives simultaneously. These problems are called multi-objective optimization problems (MOPs). In general, an MOP can be defined as:

$$\begin{aligned} &minimize \quad \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_m(\mathbf{x}))^T \\ &subject\ to \quad \mathbf{x} \in \Omega \end{aligned} \tag{1}$$

where Ω denotes the decision space, and $\mathbf{F} : \Omega \rightarrow \mathbb{R}^m$ consists of m objective functions. Furthermore, MOPs involving four or more objectives are generally referred to as many-objective optimization problems (MaOPs).

In general, the improvement of one objective of an MOP may cause the degradation of the other objectives. The algorithm can only coordinate and compromise among multiple conflicting objectives in order to maximize each objective as much as possible. Therefore, an MOP has a set of optimal solutions, rather than a unique optimal solution. Specifically, $\mathbf{x}' \in \Omega$ is said to dominate $\mathbf{x}'' \in \Omega$, written as $\mathbf{x}' \prec \mathbf{x}''$, if and only if $\forall i \in \{1, 2, \cdots, m\} : f_i(\mathbf{x}') \leq f_i(\mathbf{x}'') \wedge \exists j \in \{1, 2, \cdots, m\} : f_j(\mathbf{x}') < f_j(\mathbf{x}'')$. A solution $\mathbf{x}^*$ is said to be non-dominated if and only if $\nexists \mathbf{x} \in \Omega : \mathbf{x} \prec \mathbf{x}^*$. The set of non-dominated solutions is called the Pareto set (PS). The image of PS in the objective space is called the Pareto front (PF).

Evolutionary algorithms are widely used to solve MaOPs [1]. The existing multi-objective optimization evolutionary algorithms (MOEAs) can be divided into three categories: dominance-based MOEAs, indicator-based MOEAs and decomposition-based MOEAs [2]. However, dominance-based MOEAs have no strong selection pressure for MaOPs since almost all solutions in the population become non-dominated as the number of objectives increases. Indicator-based MOEAs have to calculate the hypervolume of the population repeatedly while the computational complexity increases exponentially with the number of objectives. Fortunately, decomposition-based MOEAs do not have these troubles mentioned above and they are very suitable for MaOPs. Furthermore, a cone decomposition evolutionary algorithm (CDEA) [3] introduces a cone decomposition strategy to establish more reasonable associations between solutions and subproblems in decomposition-based MOEAs for MaOPs.

Decomposition-based MOEAs could hardly obtain the fronts of complex MaOPs accurately due to the evenly dis-

This work was supported in part by the Natural Science Foundation of Guangdong Province under Grant 2020A1515011491, in part by the National Natural Science Foundation of China under Grant 61203310 and Grant 61503087, in part by the Pearl River S&T Nova Program of Guangzhou under Grant 2014J2200052, in part by the Research and Development Program in Key Areas of Guangdong Province under Grant 2019B010154004, in part by the Fundamental Research Funds for the Central Universities, SCUT, under Grant 2017MS043, and in part by the Platform Development Program for Innovation and Entrepreneurship at Colleges in Guangzhou under Grant 2019PT103.

tributed direction vectors. Several recent studies focus on combining the ideas of decomposition and dominance. For example, Deb and Jain proposed a non-dominated sorting genetic algorithm based on reference points (NSGA-III) [4]. An MOEA based on dominance and decomposition (MOEA/DD) [5] was also designed to solve MaOPs. Recently, several MOEAs have shown great performances for MaOPs with irregular fronts. For instance, NSGA-III$_{\mathrm{TS}}$-NL [6] is designed to considering three schemes in NSGA-III: Thompson sampling, probability matching and adaptive pursuit.

The cone decomposition strategy in CDEA performs well for MaOPs with regular PFs. However, it might lead to the loss of excellent offsprings and the survival of some dominated solutions in the population due to the even distribution of reference directions. In this paper, a cone decomposition evolutionary algorithm with a dominance-based archive (CDEA-DA) is presented to compensate for the disadvantage of CDEA for MaOPs with irregular fronts. In CDEA-DA, a dominance-based archive is designed to preserve a set of representative non-dominated solutions by the use of corner solutions [7], non-dominated sorting [8] and a distance-based selection strategy. The archive helps the decomposition-based population to avoid the potential troubles of decomposition-based MOEAs for MaOPs with irregular fronts.

The remainder of this paper is organized as follows. The main ideas of CDEA-DA are firstly introduced in Section II. The procedure of CDEA-DA is described in Section III. Section IV provides the experimental settings and results. Finally, Section V concludes this paper and prospects some future research issues.

II. Decomposition and Archive

This section describes the main ideas of CDEA-DA. CDEA-DA adopts an improved cone decomposition strategy. At the same time, since decomposition-based MOEAs are sensitive to the shapes of PFs of MaOPs, CDEA-DA introduces a dominance-based archive to assist the decomposition-based population in capturing irregular fronts.

A. Cone Decomposition Strategy

This paper makes two improvements on the original cone decomposition strategy in CDEA [3]. The first improvement is the adjustment of the number of direction vectors. In this paper, the improved cone decomposition strategy first generates plenty of direction vectors, and then it selects a specified number of reference directions based on Euclidean distance. The pseudo code for the initialization of reference directions is shown in Algorithm 1.

The function in line 5 of Algorithm 1 is introduced in this paragraph. It first picks m extreme direction vectors similar to $(0, \cdots, 1, \cdots, 0)$ and adds them to the set of reference directions. Then the minimum Euclidean distance between each remaining direction vector and the reference directions is calculated, and the direction vector corresponding to the maximum distance is chosen and added to the set of reference directions. This procedure is repeated until the number of the reference directions reaches a specified number.

Algorithm 1: Initialization of Reference Directions (InitializeDirection)

Input: m: the number of objectives; H_1: the parameter for generating reference directions in the boundary layer; H_2: the parameter for generating reference directions in the inner layer; T: the neighborhood size.

Output: D: the reference directions; B: the neighbor set; I: the kd tree.

1 **if** $m < 7$ **then**
2 Generate direction vectors D by the method in [9] ;
3 **else**
4 Generate direction vectors D by the method in [5] ;
5 $D = \texttt{ReduceVector}(D)$;
6 **for** $i = 1$ *to* N **do**
7 $B^j = \{j_1, j_2, \cdots, j_T\}$, where $\lambda^{j_1}, \lambda^{j_2}, \cdots, \lambda^{j_T}$ are T closest direction vectors of λ^i;
8 $I = \texttt{BuildKDTree}\ (D)$;
9 **return** D,B,I;

The second improvement is the redefinition of observation vectors. Observation vectors are important to the cone decomposition strategy [3]. In this paper, the modified observation vector $V(\mathbf{x}, \mathbf{z}^{ide}, \mathbf{z}^{nad})$ for solution $\mathbf{x}$ is defined as:

$$\begin{aligned} &V(\mathbf{x}, \mathbf{z}^{ide}, \mathbf{z}^{nad}) = (v_1, v_2, \cdots, v_m), \\ &v_i = \frac{f_i^*(\mathbf{x})}{\sum_{j=1}^{m} f_j^*(\mathbf{x})}, f_i^*(\mathbf{x}) = \frac{f_i(\mathbf{x}) - z_i^{ide}}{z_i^{nad} - z_i^{ide}}, \\ &i = 1, 2, \cdots, m. \end{aligned} \tag{2}$$

where $\mathbf{z}^{ide}$ is the current ideal point, and $\mathbf{z}^{nad}$ denotes the current nadir point.

After generating the specified number of reference directions, the objective space is divided into several cone subregions corresponding to the reference directions. The cone subregions Φ^i for reference direction λ^i can be defined as:

$$\begin{aligned} \Phi^i = \{&\mathbf{F}(\mathbf{x}) \in \mathbb{R}^m | \forall j \neq i : \\ &d(V(\mathbf{x}, \mathbf{z}^{ide}, \mathbf{z}^{nad}), \lambda^i) \leq d(V(\mathbf{x}, \mathbf{z}^{ide}, \mathbf{z}^{nad}), \lambda^j)\}. \end{aligned} \tag{3}$$

B. Dominance-based Archive

Since decomposition-based MOEAs are very sensitive to the shapes of PFs of MaOPs, they face many challenges when the PFs are irregular. Two challenges are described below. When the projection of one PF can not cover the entire observation hyperplane, the solutions associated with some subproblems are not Pareto optimal solutions, resulting in many final solutions being dominated. When the distribution of the PF is uneven, the solutions associated with the evenly distributed subproblem can not approximate the PF very well.

In order to deal with these challenges, this paper proposes a dominance-based archive strategy. When a solution eliminated

during evolution was not dominated by the solutions associated with its neighbor subproblems, it would be preserved in a dominance-based archive. The archive is used in the update procedure. The useful information in the archive helps the population evolve better. In the first case, when many solutions are dominated, the dominance-based archive can provide enough non-dominated solutions to ensure the quantity and quality of the final solutions. In the second case, when the solutions are unevenly distributed, the dominance-based archive can help to adjust the distribution of the solutions to approximate the PF well.

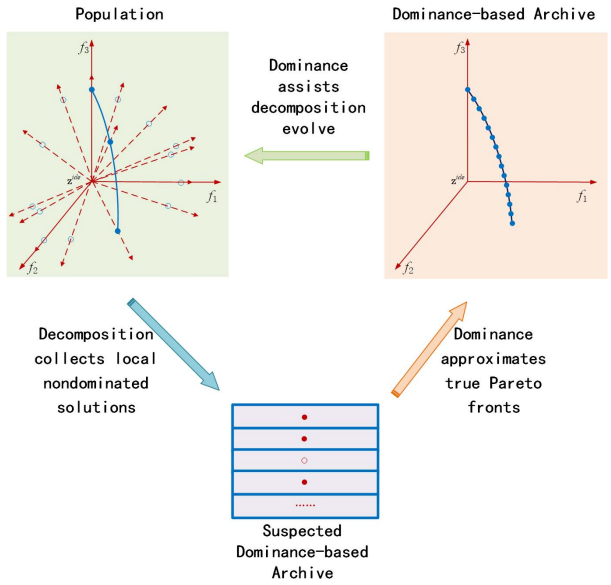

Fig. 1. The relationship among the population, suspected dominance-based archive and dominance-based archive.

The dominance-based archive can help decomposition-based MOEAs perform better on MaOPs with irregular PFs. Since the dominance-based archive is often full, how to effectively update the dominance-based archive is a key issue. This paper designs a suspected dominance-based archive to store the solutions that are eliminated during evolution but are not dominated by the other solutions in the current population. When the size of the suspected dominance-based archive reaches the population size, the dominance-based archive is mixed with the suspected dominance-based archive and some representative non-dominated solutions are selected to form a new dominance-based archive.

The relationship among the population, suspected dominance-based archive and dominance-based archive is shown in Fig. 1. The decomposition-based population collects local non-dominated solutions. In other words, when a local non-dominated solution in the population was eliminated during evolution, it would be stored in the suspected dominance-based archive. The dominance-based archive is updated by a dominance-based method, which is introduced in detail in Section II-D. The dominance-based archive helps the population evolve. For example, some solutions from the dominance-based archive are chosen as parents in the reproduction procedure.

C. Cone Update Scheme

A cone update scheme whose pseudocode is presented in Algorithm 2, is designed to update the current population with an offspring individual and to store the local non-dominated solutions during evolution. The specific process is as follows. First, the subregion Φ^k associated with $\mathbf{y}$ is determined, and the number c of updated parents, a sign U and the neighbors G that have not been compared are initialized in Lines 1∼4. If c is less than n_r and G is not empty, the solution $\mathbf{x}^i$ associated with one of the neighbors in G will be compared with $\mathbf{y}$ in Lines 5∼13. Otherwise, if $\mathbf{y}$ does not update any neighbor and $\mathbf{y}$ is locally non-dominated, it will be stored in A'. A and A' will be updated if the size of A' reaches the population size N. Finally, P, A and A' are returned. A normalized PBI aggregation approach is used in the cone update scheme. The aggregation approach can be formulated as follows:

$$
\begin{aligned}
minimize \quad & \mathbf{g}^{npbi}(\mathbf{x}|\lambda, \mathbf{z}^{ide}, \mathbf{z}^{nad}) = d_1 + \theta d_2 \\
& d_1 = \frac{||\mathbf{F}^*(\mathbf{x})^T \lambda||}{||\lambda||} \\
& d_2 = ||\mathbf{F}^*(\mathbf{x}) - d_1 \frac{\lambda}{||\lambda||}|| \\
& \mathbf{F}^*(\mathbf{x}) = (f_1^*(x), f_2^*(x), \ldots, f_m^*(x)) \\
& f_i^*(x) = \frac{f_i(x) - z_i^{ide}}{z_i^{nad} - z_i^{ide}} \\
subject\ to \quad & \mathbf{x} \in \Omega.
\end{aligned} \tag{4}
$$

D. Archive Update Scheme

It is important to update the dominance-based archive effectively. The archive update scheme is as follows. First, A and A' are mixed into a solution set R, and then the first non-dominated level R_1 of R is obtained by non-dominated sorting [8]. If the size of R_1 is greater than the population size N, a distance-based selection is performed to select N solutions to replace A in Lines 2∼18. Otherwise, R_1 is used as A. Finally, A is returned. The pseudo code for the archive update scheme is shown in Algorithm 3.

The distance-based selection is introduced as follows. First, a solution set R' is initialized to the corner set CS [7]. For the i-th solution in R_1, the Euclidean distances between its observation vector and the observation vector of solutions in R' are calculated, and the minimum distance is recorded as d_i in Lines 4∼8. The solution $\mathbf{x}^k$ with the largest value d_{max} among the minimum distances is chosen, added to R' and deleted from R_1 in Lines 10∼13. Finally, the distance set d is updated. These operations are repeated until $|R'|$ equals to N.

III. Procedure of CDEA-DA

The procedure of CDEA-DA is introduced in detail in this section.

Algorithm 2: Cone Update Procedure of Subproblem (ConeUpdate)

Input: $\mathbf{y}$: an offspring solution for update; $\mathbf{z}^{ide}$: the current ideal point; $\mathbf{z}^{nad}$: the current nadir point; P: the current population; I: the kd Tree; B: the index set of neighbor reference directions; A: the dominance-based archive; A': the suspected dominance-based archive; n_r: the maximum number of parents that can be updated.

Output: P: the population after update; A: the dominance-based archive after update.; A': the suspected dominance-based archive after update.

```
c = 0;
U = True;
k = ConeSubregion (y,z^ide,z^nad));
G = B^k;
while c < n_r and G ≠ ∅ do
    i = RandPick (G);
    if g^npbi(y|λ^j, z^ide, z^nad) < g^npbi(x^i|λ^i, z^ide, z^nad) then
        x^i = y;
        c = c + 1;
        U = False;
    else if x^i ≺ y then
        U = False;
    G = G \ i;
if c = 0 and U = True then
    add y to A';
    if |A'| == N then
        A = ArchiveUpdate (A ∪ A');
        A' = ∅;
return P, A, A'
```

Algorithm 3: Update Procedure of Dominance-based Archive (ArchiveUpdate)

Input: A: the dominance-based archive; A': the suspected dominance-based archive; CS: the corner set.

Output: A: the dominance-based archive after update.

```
R_1 = NondominatedSorting (A ∪ A');
if |R_1| > N then
    R' = CS;
    d = (d_1, d_2, ··· , d_|R_1|);
    foreach x^i ∈ R_1 do
        v^i = V(x^j, z^ide, z^nad);
        v^j = V(x^j, z^ide, z^nad);
        d_i = min_{x^j ∈ R} d(v^i, v^j);
    while |R'| < N do
        k = arg max_{k∈{1,2,··· ,|d|}} d_k;
        R_1 = R_1 \ x^k;
        d = d \ d_k;
        R' = R' ∪ x^k;
        foreach d_j ∈ d do
            v^j = V(x^j, z^ide, z^nad);
            v^k = V(x^k, z^ide, z^nad);
            d_j = min{d_j, d(v^j, v^k)};
    A = R';
else
    A = R_1;
return A
```

Algorithm 4: CDEA-DA Framework

Input: H_1: the parameter for generating reference directions in the boundary layer; H_2: the parameter for generating reference directions in the inner layer; T: the neighborhood size;

Output: $\tilde{P}^*$: An approximation to the PS.

```
[D, B, I, P, A, A', z^ide, CS, z^nad] = Initialization (m, H_1, H_2, T);
while the termination criterion is not met do
    for i ∈ {1, ··· , N} do
        y = Reproduction (i, B, N_p, A);
        z^ide = {z_1^ide, z_2^ide, ··· , z_m^ide}, where
        z_i^ide = min(y_i, z_i^ide);
        CS is updated by the method in [7];
        z^nad = {z_1^nad, z_2^nad, ··· , z_m^nad}, where
        z_j^nad = max_{y∈CS} f_j(y);
        [P, A, A'] = ConeUpdate
        (y, z^ide, z^nad, P, I, B, A, A');
    Associate solutions with subproblems;
P̃* = ArchiveUpdate (A ∪ P);
return P̃*
```

A. Framework of CDEA-DA

Algorithm 4 presents the framework of CDEA-DA. First, all global variables are initialized. Then, all the generated offsprings are used to update $\mathbf{z}^{ide}, CS, \mathbf{z}^{nad}, P, A$ and A'. When the population evolves, the solutions in the population are re-associated with the suitable subproblems. Finally, P and A are mixed to generate the final solution set $\tilde{P}^*$ through the archive update scheme. The implementation details are described in the subsequent subsections.

B. Initialization Procedure

This stage mainly implements the initialization of global variables. The specific pseudo code is shown in Algorithm 5. Furthermore, the operation associating solutions with subproblems consists of two phases. In the first phase, each solution finds its suitable subproblem according to the improved cone decomposition strategy, and it is associated with this subproblem. When multiple solutions correspond to the same subproblem, the normalized PBI approach is used for

comparison. Then the best solution is retained and the other solutions are stored in a set. In the second phase, the Euclidean distances between the observation vector of each solution in the set and the direction vectors of the subproblems not yet associated are calculated. Finally, each solution in the set is associated with the subproblem corresponding to the closest direction vector.

Algorithm 5: Initialization Procedure (Initialization)

Input: H_1: the parameter for generating reference directions in the boundary layer; H_2: the parameter for generating reference directions in the inner layer; T: the neighborhood size.

Output: D: the reference directions; B: the neighbor set; I: the kd tree; P: the population after association; A: the dominance-based archive; A': the suspected dominance-based archive; $\mathbf{z}^{ide}$: the ideal point; CS: the corner set; $\mathbf{z}^{nad}$: the nadir point;

$[D, B, I] =$ `InitializeDirection` (m, H_1, H_2, T);
$P = \{\mathbf{y}^1, \mathbf{y}^2, \cdots, \mathbf{y}^N\}$;
$A = A' = \varnothing$;
$\mathbf{z}^{ide} = \{z_1^{ide}, z_2^{ide}, \cdots, z_m^{ide}\}$, where $z_j^{ide} = \min_{\mathbf{y} \in P} f_j(\mathbf{y})$;
$CS = \{\mathbf{cs}_1, \mathbf{cs}_2, \cdots, \mathbf{cs}_m\}$;
$\mathbf{z}^{nad} = \{z_1^{nad}, z_2^{nad}, \cdots, z_m^{nad}\}$, where $z_j^{nad} = \max_{\mathbf{y} \in CS} f_j(\mathbf{y})$;
Associate solutions with subproblems;
return $D, B, I, P, A, A', \mathbf{z}^{ide}, CS, \mathbf{z}^{nad}$

C. Reproduction Procedure

The reproduction procedure includes the following four modes. In the first mode, a parent is selected from the corner set of the dominance-based archive to generate an offspring through uniform mutation. In the second mode, the parents are selected from the neighbors of the i-th subproblem. In the third mode, the parents are selected from the dominance-based archive. And in the fourth mode is, one parent is selected from the dominance-based archive, another is selected from i-th subproblem. Finally, SBX crossover [10] and polynomial mutation [11] are used to generate an offsprings in the last three modes. It is worth noting that if two parents are the same, one will be removed and reselected. In this paper, the probabilities of the above four modes are set to 0.1, 0.4, 0.4 and 0.1, respectively.

IV. Experiments And Analysis

This section mainly introduces the relevant configuration and results of the experiments, including benchmark test problems, comparative algorithms, settings for algorithms and quality analysis.

TABLE I
Main properties of MaF1 to MaF15.

Problem	Properties	Problem	Properties
MaF1	Linear	MaF9	Linear, Degenerate
MaF2	Concave	MaF10	Mixed, Biased
MaF3	Convex, Multimodal	MaF11	Convex, Disconnected, Nonseparable
MaF4	Concave, Multimodal	MaF12	Concave, Nonseparable, Biased Deceptive
MaF5	Concave, Biased	MaF13	Concave, Unimodal, Nonseparable, Degenerate
MaF6	Concave, Degenerate	MaF14	Linear, Partially separable, Large scale
MaF7	Mixed, Disconnected, Multimodal	MaF15	Convex, Partially separable, Large scale
MaF8	Linear, Degenerate		

A. Benchmark Test Problems

Fifteen benchmark test problems, MaF1 to MaF15 of the MaF test suite [12], are used for our empirical study. They contain a variety of properties to represent the real world better, such as being multimodal, disconnected, degenerate, and/or nonseparable, and having an irregular PF, a complex PS or a large number of decision variables. MaF1 to MaF15 are all minimization problems, and their properties are shown in Table I.

As suggested in [12], the number of decision variables of MaF8, MaF9 and MaF13 are set as 2, 2 and 5 respectively, and the number of decision variables of MaF14 and MaF15 are determined by $D = 20 \cdot M$, where M denotes the number of objectives. The number of decision variables of the remaining problems are set by $D = M + K - 1$, where $K = 20$ for MaF7 and $K = 10$ for others.

B. Comparative Algorithms

Four MOEAs are chosen as comparative algorithms, including two decomposition-based MOEAs, MOEA/D [2] and RVEA [13], as well as two dominance-based MOEAs, NSGA-III [4] and MOEA/DD [5]. CDEA-DA is implemented on JMetal [14], other algorithms are implemented on PlatEMO [15].

C. Setting for algorithms

The general parameter settings in our experiments and the special parameter settings for each algorithm are given below.

1) Number of objectives : $m \in \{5, 10, 15\}$.
2) Population size: $N = 240$.
3) Neighborhood size: $T = 20$.
4) Maximum number of fitness evaluations: $FE = \max\{100000, 10000 \cdot D\}$, where D denotes the number of decision variables.
5) Number of runs: Each algorithm runs independently 10 times on each test instance.
6) Reproduction operators: For the SBX crossover operator, the crossover probability and distribution index are set to $p_c = 1.0$ and $\eta_c = 20$, respectively. For the polynomial mutation, the mutation probability and distribution index are set to $p_m = \frac{1}{n}$ and $\eta_m = 20$.

TABLE II
THE AVERAGE NHV RESULTS OBTAINED BY FIVE ALGORITHMS FOR ALL TEST PROBLEMS.

Problem	m	MOEA/D	RVEA	NSGAIII	MOEA/DD	CDEA-DA
MaF1	5	$0.0054_{(2)}$	$0.0024_{(4)}$	$0.0048_{(3)}$	$0.0021_{(5)}$	$\mathbf{0.0055_{(1)}}$
	10	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$
	15	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$
MaF2	5	$0.1612_{(2)}$	$0.1505_{(4)}$	$0.1538_{(3)}$	$0.1383_{(5)}$	$\mathbf{0.1695_{(1)}}$
	10	$0.2090_{(3)}$	$0.1589_{(5)}$	$0.2212_{(2)}$	$0.1754_{(4)}$	$\mathbf{0.2236_{(1)}}$
	15	$0.1800_{(2)}$	$0.0699_{(5)}$	$0.1382_{(3)}$	$0.1056_{(4)}$	$\mathbf{0.2197_{(1)}}$
MaF3	5	$0.9848_{(5)}$	$0.9950_{(3)}$	$\mathbf{0.9987_{(1)}}$	$0.9858_{(4)}$	$0.9968_{(2)}$
	10	$0.9674_{(4)}$	$0.9903_{(2)}$	$0.2105_{(5)}$	$0.9801_{(3)}$	$\mathbf{0.9943_{(1)}}$
	15	$0.9664_{(5)}$	$0.9991_{(2)}$	$0.9891_{(4)}$	$0.9943_{(3)}$	$\mathbf{0.9995_{(1)}}$
MaF4	5	$0.0127_{(5)}$	$0.0132_{(4)}$	$\mathbf{0.0631_{(1)}}$	$0.0227_{(3)}$	$0.0272_{(2)}$
	10	$0_{(3)}$	$0_{(3)}$	$\mathbf{0.0002_{(1)}}$	$0_{(3)}$	$0.0001_{(2)}$
	15	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$
MaF5	5	$0.3812_{(5)}$	$0.7514_{(3)}$	$\mathbf{0.7793_{(1)}}$	$0.4740_{(4)}$	$0.7676_{(2)}$
	10	$0.3937_{(5)}$	$0.9536_{(2)}$	$\mathbf{0.9678_{(1)}}$	$0.5655_{(4)}$	$0.8744_{(3)}$
	15	$0.2613_{(5)}$	$0.9442_{(2)}$	$\mathbf{0.9910_{(1)}}$	$0.5006_{(4)}$	$0.8741_{(3)}$
MaF6	5	$0.1154_{(4)}$	$0.1158_{(3)}$	$\mathbf{0.1259_{(1)}}$	$0.1103_{(5)}$	$0.1177_{(2)}$
	10	$0.0995_{(2)}$	$0.0829_{(4)}$	$0.0723_{(5)}$	$0.0942_{(3)}$	$\mathbf{0.1000_{(1)}}$
	15	$0.0635_{(5)}$	$0.0917_{(3)}$	$0.0801_{(4)}$	$0.0922_{(2)}$	$\mathbf{0.0946_{(1)}}$
MaF7	5	$0.0077_{(5)}$	$0.2154_{(2)}$	$\mathbf{0.2375_{(1)}}$	$0.0909_{(4)}$	$0.2086_{(3)}$
	10	$0.0001_{(4)}$	$0.1475_{(2)}$	$\mathbf{0.1773_{(1)}}$	$0_{(5)}$	$0.1434_{(3)}$
	15	$0_{(4)}$	$0.0848_{(3)}$	$\mathbf{0.1325_{(1)}}$	$0_{(4)}$	$0.1049_{(2)}$
MaF8	5	$\mathbf{0.1035_{(1)}}$	$0.0723_{(4)}$	$0.0868_{(2)}$	$0.0713_{(5)}$	$0.0863_{(3)}$
	10	$0.0062_{(3)}$	$0.0046_{(5)}$	$0.0092_{(2)}$	$0.0062_{(3)}$	$\mathbf{0.0101_{(1)}}$
	15	$0.0003_{(3)}$	$0.0001_{(5)}$	$\mathbf{0.0005_{(1)}}$	$0.0002_{(4)}$	$\mathbf{0.0005_{(1)}}$
MaF9	5	$\mathbf{0.2861_{(1)}}$	$0.1938_{(4)}$	$0.1629_{(5)}$	$0.2352_{(3)}$	$0.2452_{(2)}$
	10	$0.0137_{(2)}$	$0.0044_{(5)}$	$0.0084_{(4)}$	$0.0098_{(3)}$	$\mathbf{0.0149_{(1)}}$
	15	$0_{(5)}$	$0.0002_{(4)}$	$0.0007_{(2)}$	$0.0005_{(3)}$	$\mathbf{0.0011_{(1)}}$
MaF10	5	$0.9352_{(4)}$	$0.9965_{(2)}$	$\mathbf{0.9976_{(1)}}$	$0.9853_{(3)}$	$0.8605_{(5)}$
	10	$0.7360_{(5)}$	$0.9968_{(2)}$	$\mathbf{0.9991_{(1)}}$	$0.9951_{(3)}$	$0.9788_{(4)}$
	15	$0.4255_{(5)}$	$0.9984_{(3)}$	$\mathbf{0.9996_{(1)}}$	$0.9955_{(4)}$	$\mathbf{0.9996_{(1)}}$
MaF11	5	$0.9547_{(5)}$	$0.9889_{(2)}$	$\mathbf{0.9954_{(1)}}$	$0.9793_{(4)}$	$0.9832_{(3)}$
	10	$0.9348_{(5)}$	$0.9887_{(3)}$	$\mathbf{0.9978_{(1)}}$	$0.9508_{(4)}$	$0.9949_{(2)}$
	15	$0.9411_{(5)}$	$0.9697_{(3)}$	$0.9975_{(2)}$	$0.9666_{(4)}$	$\mathbf{0.9979_{(1)}}$
MaF12	5	$0.6488_{(4)}$	$\mathbf{0.7349_{(1)}}$	$0.7149_{(2)}$	$0.6740_{(3)}$	$0.6303_{(5)}$
	10	$0.3718_{(5)}$	$\mathbf{0.8729_{(1)}}$	$0.8530_{(2)}$	$0.6979_{(4)}$	$0.7344_{(3)}$
	15	$0.2035_{(5)}$	$0.8646_{(2)}$	$\mathbf{0.9012_{(1)}}$	$0.7752_{(4)}$	$0.7899_{(3)}$
MaF13	5	$\mathbf{0.2327_{(1)}}$	$0.1633_{(5)}$	$0.2031_{(4)}$	$0.2123_{(3)}$	$0.2150_{(2)}$
	10	$0.0888_{(3)}$	$0.0862_{(4)}$	$0.1144_{(2)}$	$0.0852_{(5)}$	$\mathbf{0.1258_{(1)}}$
	15	$0.0251_{(4)}$	$0.0510_{(3)}$	$0.0563_{(2)}$	$0.0159_{(5)}$	$\mathbf{0.0869_{(1)}}$
MaF14	5	$0.0912_{(5)}$	$0.2536_{(4)}$	$0.4375_{(2)}$	$\mathbf{0.5170_{(1)}}$	$0.2547_{(3)}$
	10	$\mathbf{0.9504_{(1)}}$	$0.6399_{(3)}$	$0.0011_{(5)}$	$0.7691_{(2)}$	$0.2100_{(4)}$
	15	$0.1350_{(4)}$	$0.2938_{(3)}$	$0.0316_{(5)}$	$\mathbf{0.5756_{(1)}}$	$0.3578_{(2)}$
MaF15	5	$\mathbf{0.0413_{(1)}}$	$0.0174_{(4)}$	$0_{(5)}$	$0.0205_{(3)}$	$0.0216_{(2)}$
	10	$0_{(2)}$	$0_{(2)}$	$0_{(2)}$	$0_{(2)}$	$\mathbf{0.0001_{(1)}}$
	15	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$

7) Aggregation approach: CDEA-DA adopts the normalized PBI approach and the other algorithms adopt their respective default aggregation approaches.

D. Quality Analysis

The average NHV [16] results for MaF test problems are listed in Table II. Here, the values of all comparative algorithms are sorted, and the ranking of each value is shown in the subscript. According to the NHV values of the comparative algorithms for MaF test problems in Table II, it is obvious that CDEA-DA ranks first in 21 of 45 comparisons. Simultaneously, MOEA/D, RVEA, NSGA-III and MOEA/DD achieve 9, 6, 21 and 6 best results, respectively. It can be concluded that, as a decomposition-based MOEA, CDEA-DA performs better than the other decomposition-based MOEAs on most of MaF test problems. Though decomposition-based MOEAs encounter many difficulties on MaF test problems, CDEA-DA is competitive with dominance-based MOEAs. According to Table I and Table II, CDEA-DA performs well in the problems with three kinds of fronts.

1) Linear Fronts: Linear Fronts are more simple than other fronts. MaF1, MaF8, MaF9 and MaF14 have linear fronts according to Table I. CDEA-DA performs better than the other comparative algorithms on MaF1, MaF8 and MaF9, because the cone decomposition strategy improves diversity and the dominance-based archive strategy improves convergence. However, CDEA-DA does not perform well on MaF14, because the front of MaF14 has more properties such as being partially separable and having large scale.

2) Convex Fronts: Most of the existing decomposition-based MOEAs face many challenges on the convex front, because the convex front is unevenly distributed. When we use a decomposition-based MOEA, the solutions associated with evenly distributed subproblems cannot generally approximate the PF well. MaF3, MaF11 and MaF15 are with convex fronts according to Table I, and CDEA-DA preforms well in these problems according to Table II. CDEA-DA can help to adjust the distribution of solutions and make them approximate the true PF due to the dominance-based archive. Also, the corner

set can reserve the best solutions on some objectives.

3) Degenerate Fronts: According to Table I, MaF6, MaF8, MaF9 and MaF13 have degenerate fronts. The projections of these fronts are part of the observation hyperplane. Most of the solutions obtained by decomposition-based MOEAs are dominated. Also, dominance-based MOEAs do not perform well on these MaOPs with degenerate fronts. CDEA-DA achieves the best results on the problem with degenerate fronts, because it can provide enough non-dominated solutions to ensure the quantity and quality of the final solutions.

In addition, inverted fronts are similar to degenerate fronts. MaF1 and MaF15 are with the typical inverted fronts. The projection of these PFs cannot cover the entire observation hyperplane. Since inverted fronts bring the same challenges as degenerate fronts, CDEA-DA also achieves very good performances for MaF1 and MaF15.

V. Conclusion

In this paper, the CDEA-DA algorithm is proposed to extend decomposition-based MOEAs for MaOPs with irregular fronts. First, a dominance-based archive strategy is designed and combined with an improved cone decomposition strategy. When decomposition-based MOEAs are used for complex MaOPs, the evenly distributed reference directions result in the loss of the excellent offspring during evolution. The dominance-based archive strategy can deal with these challenges well. Moreover, a cone update scheme maintains the balance between convergence and diversity. Furthermore, an archive update scheme is used to heighten the quality of the solutions. The performance of CDEA-DA is investigated on a number of unconstrained benchmark MaF test problems with 5-, 10- and 15-objectives. The empirical results demonstrate that CDEA-DA performs very well at maintaining the population convergence and diversity for most of MaF test problems when compared with four state-of-the-art MOEAs.

Our future research will focus on the adaptive adjustment of reference directions. Most of the existing decomposition-based MOEAs preset a set of reference directions. This fact causes a problem that they are sensitive to irregular PFs. In the future, we plan to adjust the number and positions of reference directions adaptively during evolution.

References

[1] S. Bechikh, M. Elarbi, and L. B. Said, "Many-objective optimization using evolutionary algorithms: A survey," in *Recent Advances in Evolutionary Multi-objective Optimization*. Springer, 2017, pp. 105–137.

[2] 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.

[3] W. Ying, Y. Deng, Y. Wu, Y. Xie, Z. Wang, and Z. Lin, "A cone decomposition many-objective evolutionary algorithm with adaptive direction penalized distance," in *Proc. International Conference on Bio-Inspired Computing: Theories and Applications*. Springer, 2018, pp. 389–400.

[4] 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, 2013.

[5] 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, 2014.

[6] J. Kuk, R. Goncalves, C. Almeida, S. Venske, and A. Pozo, "A new adaptive operator selection for NSGA-III applied to CEC 2018 many-objective benchmark," in *Proc. Brazilian Conference on Intelligent Systems*. IEEE, 2018, pp. 7–12.

[7] H. K. Singh, A. Isaacs, and T. Ray, "A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 539–556, 2011.

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

[9] 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.

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

[11] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and Informatics*, vol. 26, no. 4, pp. 30–45, 1996.

[12] R. Cheng, M. Li, Y. Tian, X. Xiang, X. Zhang, S. Yang, Y. Jin, and X. Yao, "Benchmark functions for the CEC'2018 competition on many-objective optimization," University of Birmingham, United Kingdom, Tech. Rep., 2018.

[13] 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.

[14] J. J. Durillo and A. J. Nebro, "jMetal: A java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, no. 10, pp. 760–771, 2011.

[15] 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.

[16] 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.