An Evolutionary Many-Objective Optimization Algorithm Based on Corner Solution Search

Haoran Sun, Chunyang Zhu, Xinye Cai*

***Abstract*—In this report, we propose an evolutionary many-objective optimization algorithm based on corner solution search (MaOEA-CS). The corner solution is approximated by both decomposition and Pareto-dominance based strategies with exploitative search at the early stage; and then angle-based selection is adopted with explorative search for the extension of PF approximation at the last search stage. Experimental results on all benchmark functions for CEC′2017 Competition show the effectiveness and efficiency of MaOEA-CS.**

I. Introduction

A multiobjective optimization problem (MOP) can be defined as follows:

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

where Ω is the *decision space*, $F : \Omega \to R^m$ consists of m real-valued objective functions. The *attainable objective set* is $\{F(x)|x \in \Omega\}$. Let $u, v \in R^m$, u is said to *dominate* v, denoted by $u \prec v$, if and only if $u_i \leq v_i$ for every $i \in \{1, \ldots, m\}$ and $u_j < v_j$ for at least one index $j \in \{1, \ldots, m\}$[1]. A solution $x^* \in \Omega$ is *Pareto-optimal* to (1) if there exists no solution $x \in \Omega$ such that $F(x)$ dominates $F(x^*)$. The set of all the Pareto-optimal points is called the *Pareto set* (*PS*) and the set of all the Pareto-optimal objective vectors is the *Pareto front* (*PF*) [1]. MOPs with more than three objectives are informally known as many-objective optimization problems (MaOPs).

In [2], 15 test problems with different shapes of PFs are proposed for CEC′2017 Competition on Evolutionary Many-Objective Optimization. This test suite aims to promote the research of evolutionary many-objective optimization (EMaO) via suggesting a set of test problems with a good representation of various real-world scenarios.

To address these problems, an evolutionary many-objective optimization algorithm based on corner solution search (MaOEA-CS) is proposed. MaOEA-CS can be considered as a novel combination of decomposition and Pareto-dominance based strategies with exploitative search for corner solutions at the early stage, and uses angle-based selection [3] with explorative search for the extension of PF approximation at the last search stage.

*Corresponding author

Haoran Sun and Chunyang Zhu, Xinye Cai are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, 210016 P. R. China (e-mail: nuaa_sunhr@yeah.net).

This work was supported in part by the National Natural Science Foundation of China (NSFC) under grant 61300159, by the Natural Science Foundation of Jiangsu Province of China under grant BK20130808 and by China Postdoctoral Science Foundation under grant 2015M571751.

[1]In the case of maximization, the inequality signs should be reversed.

The rest of this report is organized as follows. Section II introduces MaOEA-CS. Section III presents the experimental results of MaOEA-CS on 15 MaOPs with the different number of objectives for CEC′2017 competition [2]. Section IV concludes this report.

II. MaOEA-CS

A. Corner solution search

Based on [4], Pareto corner solution can be defined as follows.

Definition 1. *(Corner solution): For an optimization problem with m-objectives, the objective set is denoted by F_m and a subset of the original objective set is denoted by F_k ($k < m$). If minimizing the k-objectives in this subset results in a single (Pareto optimal) solution in the m-objective space, then this solution is referred to as a corner (or Pareto corner) solution.*

For instance, the corner solutions of three different PFs in Fig. 1 is shown in red circles.

Given a population P and its nondominated subset P', two types of solutions can be used to approximate corner solutions, as follows.

1) The solutions in P' that are closest to the coordinate axis in the objective space.

$$P_1 = \{x|x = \underset{x \in P'}{\operatorname{argmin}}\, dist^{\perp}(F(x), e^i), i = 1, 2, \ldots, m\} \tag{2}$$

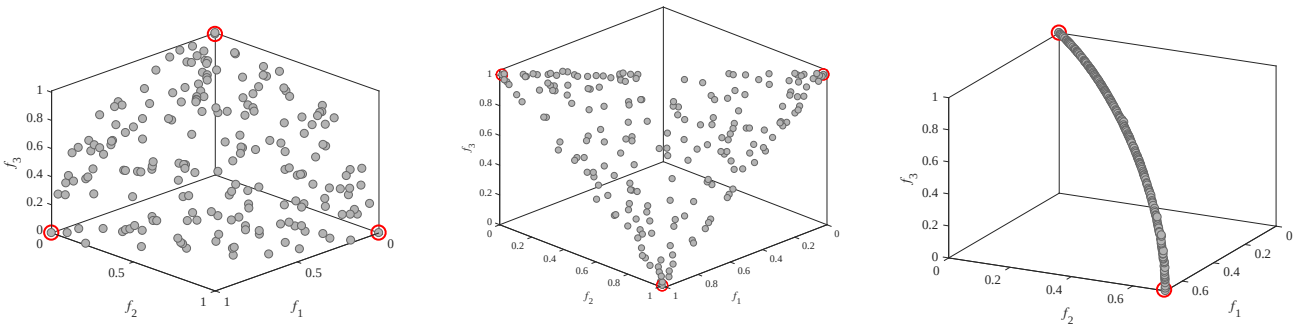
where $dist^{\perp}(a, b)$ indicates the the perpendicular distance from a vector a to a direction vector b and e^i denotes the direction vector of i-th axis.

2) The solutions in P' that has the lowest value along each of the coordinate axes.

$$P_2 = \{x|x = \underset{x \in P'}{\operatorname{argmin}}\, f_i(x), i = 1, 2, \ldots, m\} \tag{3}$$

The above method for obtaining corner solutions can be considered as the combination of Pareto dominance and decomposition only using important direction vectors along all the coordinate axes. After the corner solution set P_c ($|P_c| \leq 2m$) is approximated, the nadir point z^{nad} can be further approximated as follows.

$$\begin{aligned} &z^{nad} = (z_1^{nad}, z_2^{nad}, \ldots, z_m^{nad})^T, \\ &\text{where} \quad z_i^{nad} = \max_{x \in P_c} f_i(x). \end{aligned} \tag{4}$$



(a) (b) (c)

Fig. 1: Three different PF approximations and their corner solutions

The whole procedures of the corner solution search is given in Algorithm 1. P_1 is firstly obtained based on Eq. (2) and z^{nad} is approximated by P_1 based on Eq. (4). After that, P_2 is obtained based on Eq. (3). If the objective value of any solution x in P_2 is larger than that of the approximated z^{nad}, x is also added to P_c as a corner solution.

Algorithm 1 can be regarded as the combination of Pareto-dominance and a simple decomposition based selection [5]. A very natural extension is to consider more search directions (i.e., subproblems in MOEA/D [5]), which may improve the performance for many-objective optimization problems with very irregular PFs. However, the appropriate balance between more subproblems for better coverage of PFs and the affordable computational cost should be carefully considered, which could be an interesting research direction for the future.

Algorithm 1: Corner Solution Search (CS)

Input : P: a population;
Output: P_c: population of corner solutions;

Search P_1 according to Eq (2);
$P_c = P_1$;
Calculate z^{nad} according to Eq (4);
Search P_2 according to Eq (3);
for $x \in P_2$ **do**
  for $i = 1$ **to** m **do**
    if $f_i(x) > z_i^{nad}$ **then**
      $P_c = P_c \cap \{x\}$;
      break;
    end
  end
end
return P_c;

B. The Framework of MaOEA-CS

The main procedure of MaOEA-CS is presented in Algorithm 2. At first, a population P and the corner solution set P_c are initialized. Then, the reproduction and selection steps are applied to P and P_c iteratively until the termination criterion is fulfilled. The steps of INITIALIZATION, REPRODUCTION, and DSA-SELECTION in Algorithm 2 are explained as follows.

Algorithm 2: Framework of MaOEA-CS (MAOEA-CS)

Input : N: Population size;
δ: reproduction probability parameter;
m: The number of objectives.
Output: The final population.

$[P, P_c] =$ INITIALIZATION(N, m);
while *termination criterion is not fulfilled* **do**
  $Q =$ REPRODUCTION(P, P_c);
  $[P, P_c] =$ DSA-SELECTION$(P \bigcap Q)$;
end
return P;

C. Initialization

The initialization steps are given in algorithm 3. The population P is randomly generated. m direction vectors E along every the coordinate axes are initialized. The corner solution set P_c is also initialized by calling Algorithm 1.

Algorithm 3: Initialization (INITIALIZATION)

Input : N: population size;
m: the number of objectives;
Output: P: The initial population;
P_c:cornoer population selected from P;

Initialize P randomly;
/* E stores the direction vector along all the coordinate axes. */
$E = (e^1, e^2, \ldots, e^m)$;
$P_c = \text{CS}(P)$;
return P, P_c;

D. Reproduction

Two types of reproduction methods, called exploitative search and explorative search, are used in MaOEA-CS. The mutation operator in [6], [7] is used as exploitative search as follows. Given a solution x, every component in x is mutated with the probability of P_m and x is mutated once at least. If the

i-th component of x is selected to be mutated, its offspring's i-th component x_i^c is calculated by:

$$\begin{aligned} & x_c^i = x^i + rnd \times (ub[i] - lb[i]), \\ \text{where} \quad & rnd = 0.5 \times (rand - 0.5) \times (1 - rand^{\alpha}), \\ & \alpha = 0.7 \times (-(1 - \frac{fe}{max\_fe})). \end{aligned} \tag{5}$$

where $ub[i]$ and $lb[i]$ are the upper bound and lower bound of x_i; $rand$ is a random number in $[0, 1]$; α is a simulated annealing variable in which fe is the current number of function evaluations and max_fe is the maximal number of function evaluations.

SBX crossover [8] and polynomial mutation [9] operator is used as explorative search in MaOEA-CS. In addition, the probability of calling exploitative or explorative search is controlled by a predefined parameter δ.

The reproduction procedures are given in Algorithm 4. At first, the offspring population Q is initialized as an empty set. When a random number in $[0, 1]$ is less than δ, each solution x in P_c undergoes the exploitative search for $\lfloor \frac{|P|}{|P_c|} \rfloor$ times to generate $\lfloor \frac{|P|}{|P_c|} \rfloor$ offspring where $\lfloor \bullet \rfloor$ is the floor function. All the solutions in P_c will generate N offsprings where N is the population size. When the random number in $[0, 1]$ is larger than δ, explorative search is conducted on P to generate N offspring.

As the fast convergence towards corner solutions is more important in the early stage and the extension of the corner solutions for diversity becomes more important at the last stage, δ is switched from a large number for exploitative search in the early stage to a small number $(1 - \delta)$ for explorative search at the last stage, determined by the decrease of z^{nad} (Δ_t), as follows.

$$\Delta_t = \max_{x \in \{1,2,\ldots,m\}} \frac{|z_i^{nad}(t) - z_i^{nad}(t - len)|}{|z_i^{nad}(t - len)|} \tag{6}$$

where $z_i^{nad}(t)$ indicates the i-th objective of z^{nad} approximation at the t-the iteration and len is the learning period.

If Δ_t is less than a preset very small number, indicating the exploitative search has already converged to the corner solutions, the value of δ is switched to $(1 - \delta)$.

E. DSA-Selection

The environmental selection of MaOEA-CS, presented in Algorithm 5, is called DAS, including **d**ominance, **s**pace division and **a**ngle based selection, detailed as follows.

At first, the solution set R is ranked into various levels $(R_1, ...R_l)$ by calling nondominated sorting. The nondominated solutions in R can be obtained directly from R_1. Then, it can be used to approximate ideal point z^* as follows.

$$\begin{aligned} & z^* = (z_1^*, z_2^*, \ldots, z_m^*)^T, \\ \text{where} \quad & z_i^* = \min_{x \in P_c} f_i(x). \end{aligned} \tag{7}$$

The corner solutions are selected from R_1 by calling Algorithm 1 and z^{nad} is computed based on Eq (4). The objective space can be divided into the inside and outside space by z^{nad}, as shown in Fig. 2. Given a solution x, if there exists

Algorithm 4: Reproduction (REPRODUCTION)

Input : P: A population;
P_c: A corner solution population;
Output: Q: Offspring population;

```
1  Q = φ;
   /* δ is a probability parameter to
      control exploitative and explorative
      search                                  */
2  if rand < δ then
3      for x ∈ P_c do
4          for i = 1 to ⌊|P|/|P_c|⌋ do
               /* Apply exploitative search on
                  x according to Eq (5)       */
5              x^c = EXPLOITATIVE-SEARCH(x);
6              Q = Q ∩ {x^c};
7          end
8      end
9  else
       /* Apply SBX crossover and
          polynomial mutation on all
          solutions in P.                     */
10     Q = EXPLORATIVE-SEARCH(P);
11 end
12 return Q;
```

$i \in \{1, 2, \ldots, m\}$, where $f_i(x)$ is larger than z_i^{nad}, we say x is located in the outside space. Otherwise, we say x is located in the inside space.

Based on the size of R_1, there could be three cases, as follows.

1) $|R_1| > N$ (line 5 - 20): All the solutions in R_1 that are located in the inside space are added to P_{in}, the rest solutions are added to P_{out}. When $|P_{in}| > N$, angle based selection (ABS) (Algorithm 6) is applied to P_{in} to further select N solutions (line 13 - 14). When $|P_{in}| < N$, $N - |P_{in}|$ solutions closest to z^* in P_{out} are selected and added to P with P_{in}.
2) $|R_1| < N$ (line 21 - 23): $N - |R_1|$ solutions nearest to z^* are selected from $R \setminus R_1$ and added to P with R_1.
3) $|R_1| = N$ (line 24 - 26): R_1 is assigned to P directly.

In the angle-based selection (ABS), The objective vector $F^n(x)$ of each solution x is firstly normalized.

$$\begin{aligned} & F^n(x) = (f_1^n(x), f_2^n(x), \ldots, f_m^n(x))^T, \\ \text{where} \quad & f_i^n(x) = \frac{f_i(x) - z_i^*}{z_i^{nad} - z_i^*}. \end{aligned} \tag{8}$$

The angle between two solutions, x and y, can be calculated by:

$$angle(x, y) = \arccos(\frac{F^n(x)^T \cdot F^n(y)}{\|F^n(x)\| \|F^n(y)\|}) \tag{9}$$

The procedures of ABS are given in Algorithm 6. The corner solutions are firstly added to P and deleted from Q. For i-th solution in Q, θ_i is used to record the minimal angle from it to its nearest solution in P. The solution with maximal θ is added to P and deleted from Q one by one until the population

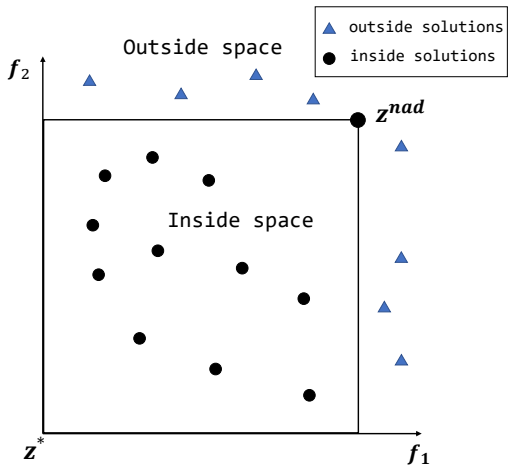


Fig. 2: An illustration of division of the inside and outside space by the nadir point approximation.

Algorithm 5: The selection of MaOEA-CS (DSA-SELECTION)

Input : R: The merged population($|R| > N$);
Output: P: The selected solution set ($|P| = N$). P_c: The corner solution set.

/* Use fast nondominated sorting rank all the solutions. */
$(R_1, R_2, \ldots, R_l) =$ NONDOMINATED-RANKING(R);
Update z^* based on Eq (7);
$P_c =$ CS(R_1);
Update z^{nad} based on Eq (4);
if $|R_1| > N$ **then**
  $P_{out} = \phi$;
  foreach $x \in R_1$ **do**
    if $\exists i \in \{1, 2, \ldots, m\}, f_i(x) > z_i^{nad}$ **then**
      $P_{out} = P_{out} \cap \{x\}$;
    end
  end
  $P_{in} = R_1 \setminus P_{out}$;
  if $|P_{in}| > N$ **then**
    $P =$ ABS$(P_{in}, P_c, z^*, z^{nad})$;
  else if $|P_{in}| < N$ **then**
    select $|N - P_{in}|$ solutions closest to z^* from P_{out} and add them to P_{in};
    $P = P_{in}$;
  else
    $P = P_{in}$;
  end
else if $|R_1| < N$ **then**
  Select $|N - R_1|$ solutions closet to z^* from $R \setminus R_1$ and add them to R_1;
  $P = R_1$;
else
  $P = R_1$;
end
return P, P_c;

size of P reaches N. If the angle between each solution in Q with newly added solution x is larger than its previous θ, θ is updated by the angle value.

It is worth noting that ABS works similarly to the weight setting method in [10], except that ABS uses angles to select solutions while distances are used in [10] to select weight vectors (direction vectors).

Algorithm 6: The angle based selection (ABS)

Input : Q: A population whose size is larger than N;
P_c: The corner solution population;
z^*: The ideal point;
z^{nad}: The nadir point;
Output: P: A population whose size is equal to N;

$P = \phi$;
$P = P \bigcap P_c$;
$Q = Q \setminus P_c$;
$\theta = (\theta_1, \theta_2, \ldots, \theta_{|Q|})$;
foreach $x^i \in Q$ **do**
  $\theta_i = \min_{x^j \in P}\{angle(x^i, x^j)\}$;
end
while $|P| < N$ **do**
  $k = \text{argmax}_{k \in \{1,2,\ldots,|Q|\}}\{\theta_k\}$;
  $Q = Q \setminus \{x^k\}$;
  $\theta = \theta \setminus \{\theta^k\}$;
  $P = P \bigcap \{x^k\}$;
  foreach $\theta_j \in \theta$ **do**
    $\theta_j = \max\{\theta_j, angle(x^k, x^j)\}$;
  end
end
return P;

III. EXPERIMENTAL RESULTS

A. Parameter settings

The population size, termination criterion and the run times are set as suggested in [2]. In reproduction step, δ is set to 0.9; the crossover probability p_c is set to 1; the distribution index η_c is set to 20 for SBX; the mutation probability p_m is set to $1/D$, where D is the number of decision variables; the distribution index η_m for polynomial mutation is set to 20; Δ_t is set to $0.001 \times m$ where m is the number of objectives and the learning period len is set to 50. The algorithm is programmed in MATLAB and embedded in the open-source MATLAB software platform platEMO [11], suggested by [2].

B. Performance of MaOEA-CS

IGD [12]–[14] and HV [15] are two widely used performance metrics. Both of them can simultaneously measure the convergence and diversity of the obtained solutions.

Table. I shows the mean IGD and HV values over 31 runs obtained by MaOEA-CS. All of these values in the table are calculated by platEMO.

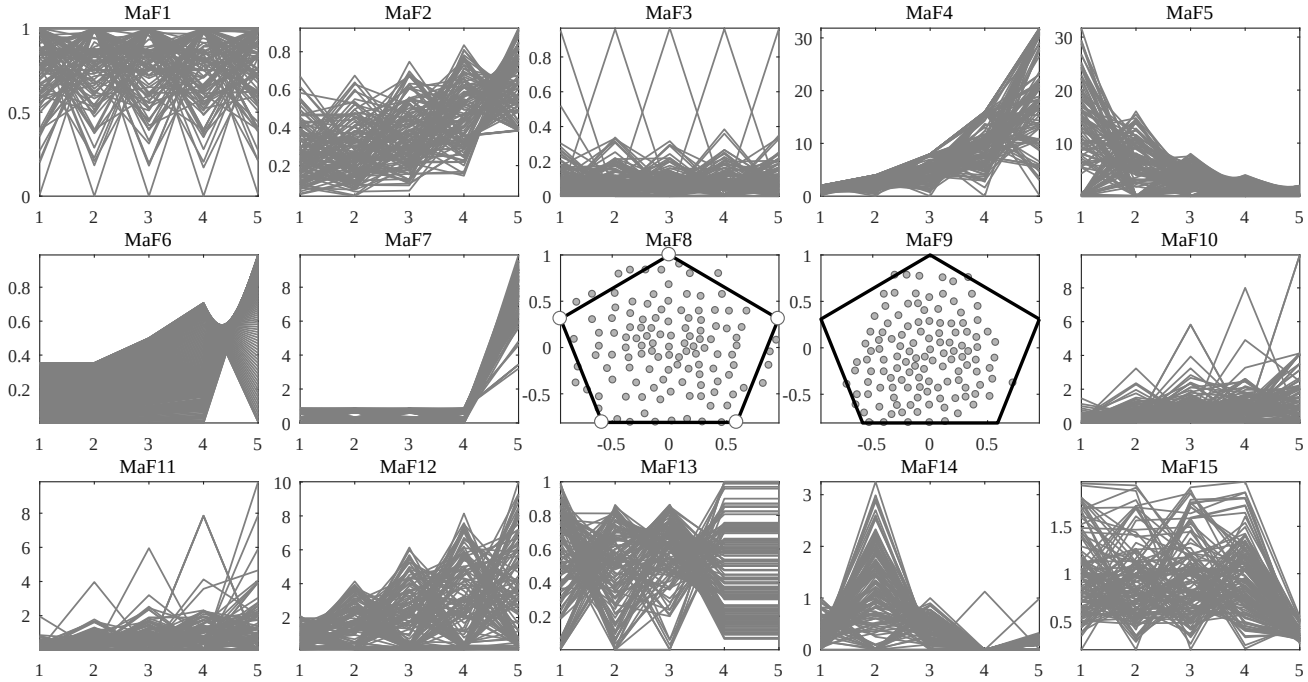


Fig. 3: Final PF approximations obtained by MaOEA-CS with median IGD values on all 5-objective problems.

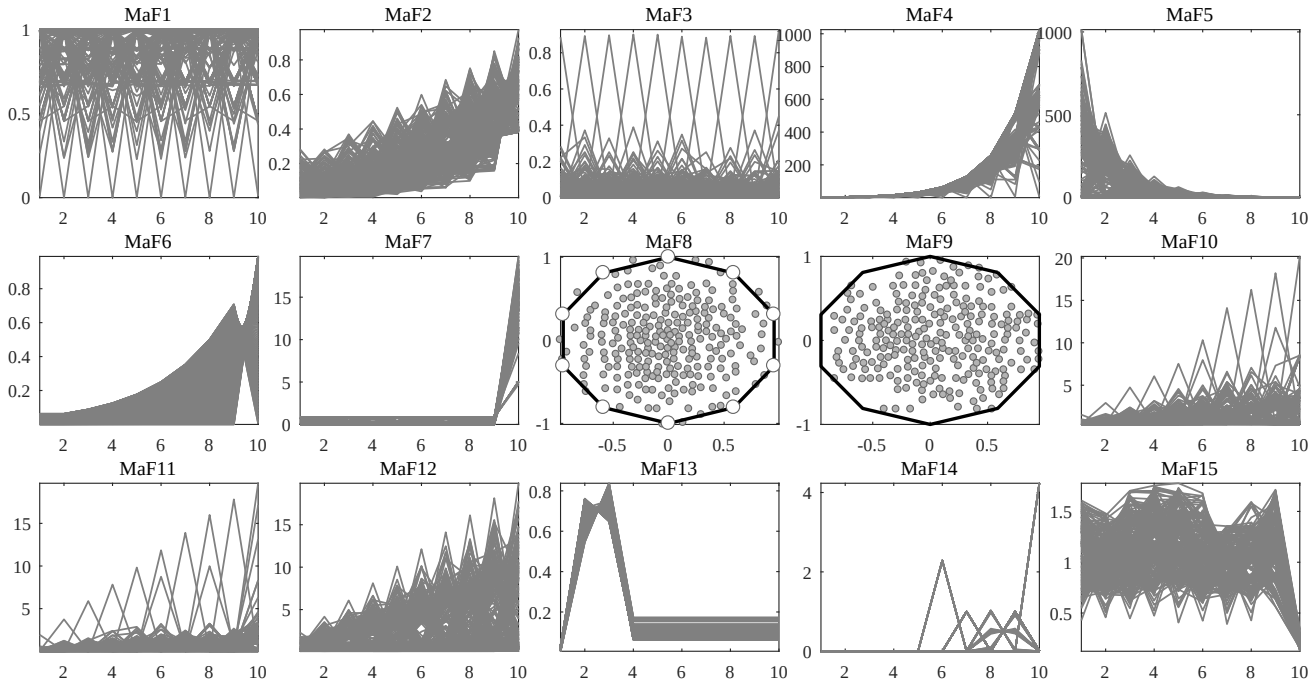


Fig. 4: Final PF approximations obtained by MaOEA-CS with median IGD values on all 10-objective problems.

TABLE I: IGD and HV results obtained by MaOEA-CS on all test problems

Problem	IGD			HV		
	M = 5	M = 10	M = 15	M = 5	M = 10	M = 15
MaF1	1.2260e-01 (7.47e-04)	2.2733e-01 (1.31e-03)	2.5363e-01 (1.18e-03)	1.1079e-02 (1.10e-04)	4.5161e-07 (5.06e-07)	0.0000e+00 (0.00e+00)
MaF2	1.0055e-01 (1.34e-03)	2.4240e-01 (2.18e-02)	4.2495e-01 (1.99e-02)	1.8454e-01 (2.35e-03)	2.1264e-01 (5.86e-03)	2.2896e-01 (5.98e-03)
MaF3	1.0154e-01 (2.41e-03)	1.0486e-01 (3.07e-03)	1.0135e-01 (3.17e-03)	9.9881e-01 (7.13e-05)	1.0000e+00 (9.67e-06)	1.0000e+00 (2.50e-07)
MaF4	2.1861e+00 (5.33e-02)	5.2902e+01 (2.43e+00)	1.5433e+03 (1.74e+02)	1.0617e-01 (1.85e-03)	5.1268e-05 (1.45e-05)	0.0000e+00 (0.00e+00)
MaF5	2.0736e+00 (3.28e-02)	4.7272e+01 (1.66e+00)	1.1007e+03 (7.66e+01)	7.8197e-01 (2.37e-03)	9.5377e-01 (1.86e-03)	9.8553e-01 (2.48e-03)
MaF6	4.0256e-03 (1.35e-04)	2.5991e-03 (2.15e-05)	1.9128e-03 (1.06e-04)	1.2969e-01 (3.36e-04)	1.0103e-01 (3.11e-04)	9.5470e-02 (2.21e-04)
MaF7	3.2770e-01 (8.70e-03)	8.9831e-01 (1.56e-02)	1.7807e+00 (9.32e-02)	2.5002e-01 (2.71e-03)	1.6945e-01 (7.13e-03)	1.3407e-01 (3.12e-03)
MaF8	1.0261e-01 (1.69e-03)	1.1542e-01 (2.92e-03)	1.2265e-01 (4.32e-03)	1.2142e-01 (3.82e-04)	1.0938e-02 (1.17e-04)	6.6450e-04 (3.15e-05)
MaF9	1.4359e-01 (3.68e-02)	1.2143e-01 (1.57e-02)	1.4772e-01 (4.38e-02)	2.9194e-01 (1.33e-02)	1.8487e-02 (2.14e-04)	1.3408e-03 (9.83e-05)
MaF10	5.3833e-01 (4.92e-02)	1.7940e+00 (1.62e-01)	2.7912e+00 (4.25e-01)	9.1234e-01 (3.97e-02)	6.0013e-01 (6.34e-02)	4.6456e-01 (1.49e-01)
MaF11	8.2281e-01 (4.61e-02)	2.5763e+00 (1.74e-01)	1.5681e-01 (8.27e-02)	9.9289e-01 (1.05e-03)	9.9378e-01 (1.61e-03)	9.9216e-01 (2.73e-03)
MaF12	1.1059e+00 (2.36e-02)	4.0455e+00 (3.44e-02)	6.5009e+00 (1.19e-01)	6.5874e-01 (5.38e-02)	7.6345e-01 (5.46e-02)	7.6286e-01 (5.39e-02)
MaF13	2.5106e-01 (7.45e-02)	1.3149e+00 (1.67e-01)	1.6720e+00 (2.51e-01)	2.5023e-01 (2.54e-02)	1.1529e-01 (2.16e-02)	7.9403e-02 (1.95e-02)
MaF14	4.4746e-01 (7.91e-02)	7.5174e-01 (1.53e-01)	8.7735e-01 (1.26e-01)	6.6177e-01 (1.34e-01)	4.6539e-01 (2.51e-01)	3.5599e-01 (2.17e-01)
MaF15	4.6998e-01 (6.84e-02)	9.2723e-01 (6.40e-02)	1.1060e+00 (4.03e-02)	4.0046e-02 (1.02e-02)	5.6854e-06 (4.90e-06)	1.3470e-10 (1.81e-10)

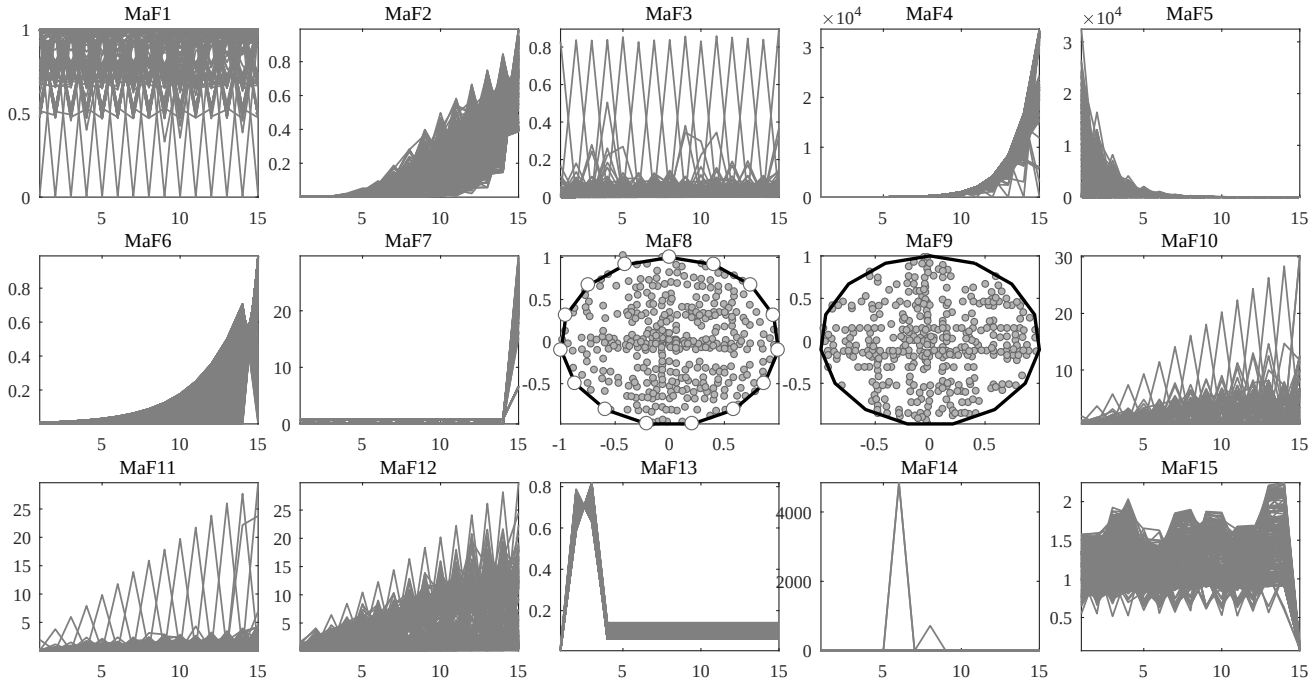


Fig. 5: Final PF approximations obtained by MaOEA-CS with median IGD values on all 15-objective problems.

C. Parallel coordinate plots

The parallel coordinate plots [16] of PF approximations on 5-, 10-, and 15-objective problems are given in Fig. 3, 4 and 5, respectively.

IV. Conclusion

In this report, a many-objective evolutionary algorithm based on corner solution search was proposed. It was tested on 45 challenging MaOPs for CEC′2017 MaOEA Competition and the experimental results were presented in details.

References

[1] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston: Kluwer Academic Publishers, 1999.

[2] 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&Intelligence Systems*, vol. 3, no. 1, pp. 67–81, 2017.

[3] X. Cai, Z. Yang, Z. Fan, and Q. Zhang, "Decomposition-based-sorting and angle-based-selection for evolutionary multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, in press, 2017.

[4] 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.

[5] 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.

[6] 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.

[7] H. L. Liu and X. Li, "The multiobjective evolutionary algorithm based on determined weight and sub-regional search," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, pp. 1928–1934, 2009.

[8] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 3, pp. 115–148, 1994.

[9] K. Deb and M. Goyal, "A combined genetic adaptive search for engineering design," in *International Conference on Genetic Algorithm*, pp. 30–45, 1999.

[10] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained mop test instances," Working Report CES-491, School of CS and EE, University of Essex, Feburary 2009.

[11] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "Platemo: A matlab platform for evolutionary multi-objective optimization," 2017.

[12] M. R. Sierra and C. A. Coello, "A new multi-objective particle swarm optimizer with improved selection and diversity mechanisms," 2004.

[13] C. A. C. Coello and M. R. Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *Mexican International Conference on Artificial Intelligence*, pp. 688–697, 2004.

[14] A. Zhou, Q. Zhang, Y. Jin, E. P. K. Tsang, and T. Okabe, "A model-based evolutionary algorithm for bi-objective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005, 2-4 September 2005, Edinburgh, UK*, pp. 2568–2575, 2005.

[15] E. Zitzler and L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 257–271, November 1999.

[16] A. Inselberg, *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. New York, NY, USA: Springer, 2009.