2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration

An Evolutionary Many-Objective Optimization Algorithm Based on Coverage and Cache Strategy

Haoran Sun
The College of Computer Science and Technology
Nanjing University of Aeronautics and Astronautics,
Nanjing, Jiangsu, 210016 P. R. China
Email: nuaa_sunhr@yeah.net

Xinye Cai
The College of Computer Science and Technology
Nanjing University of Aeronautics and Astronautics,
Nanjing, Jiangsu, 210016 P. R. China
Email: xinye@nuaa.edu.cn

Muhammad Sulaman
The College of Computer Science and Technology
Nanjing University of Aeronautics and Astronautics,
Nanjing, Jiangsu, 210016 P. R. China
Email: sulman0909@gmail.com

Zhun Fan
Department of Electronic Engineering,
School of Engineering,
Shantou University,
Guangdong, P. R. China
Email: zfan@stu.edu.cn

***Abstract*—How to balance the diversity and convergence plays an important role on the performance of a multiobjective evolutionary optimizer. Due to the loss of selection pressure and the exponential expansion in the high-dimensional objective space, it is even more difficult for an optimizer to balance between convergence and diversity for a many-objective optimization problem. To address this issue, in this paper, we propose a cache mechanism to improve the convergence and a coverage-based method for maintaining better diversity. Based on these two mechanisms, a many-objective evolutionary algorithm is further proposed. The experimental studies are conducted to verify the effectiveness of the proposed approach.**

I. Introduction

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

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

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 $i \in \{1, \dots, m\}$ and $u_j < v_j$ for at least one index $j \in \{1, \dots, m\}$[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]. A Pareto front approximation apparently can be very helpful for decision makers to understand the tradeoff relationship among different objectives and choose their preferred solutions. Over the past decades, multiobjective evolutionary algorithms (MOEAs) have been recognized as a major methodology for approximating the PFs in multiobjective optimization problems (MOPs) [2]–[4].

In MOEAs, selection is of great importance for the performance of MOEAs. Usually, it is desirable to balance between *convergence* and *diversity* for obtaining good approximation to the set of Pareto optimal solutions [5]. Convergence can be measured as the distance of solutions towards the PF, which should be as small as possible. Diversity can be measured as the spread of solutions along the PF, which should be as uniform as possible.

It is usually even more difficult for a MOEA to address many-objective optimization problems (MaOPs), i.e., MOPs with more than three objectives, which are very common in the real-world applications. For dominance-based MOEAs [3], [6], it becomes increasingly difficult to keep the selection pressure of nondominated solutions in the high-dimensional objective space, which makes them ineffective, in terms of maintaining the convergence. In addition, the density estimation of the population in MaOPs also becomes more difficult for the diversity maintenance.

In this paper, we propose a many-objective evolutionary algorithm based on coverage and cache strategy (MaOEA-CC). In MaOEA-CC, the density of a solution in the population is evaluated by the number of reference vectors covered by it and the convergence is further improved by a cache strategy. The rest of this paper is organized as follows. Section II elaborates the MaOEA-CC. In Section III, the systematic experiments are conducted to verify the effectiveness of MaOEA-CC. Finally, Section IV concludes this paper.

II. MaOEA-CC

In this section, we firstly introduce the method to evaluate the diversity and improve the convergence, which is the main contributions of this paper. Then, these two methods are embedded in a dominance-based MOEA.

A. Density estimation based on coverage

It is well-known that a set of uniformly distributed reference vectors can divide the objective space into subregion [7], [8]. Given a solution set S, to evaluate the diversity of these solutions, a set of uniformly distributed reference vectors V ($|V| = |S|$) are firstly generated by the method in [9] or [10]. Then, each reference vector in V finds its nearest solution.

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

978-1-5386-2434-0/17 $31.00 © 2017 IEEE
DOI 10.1109/ICIICII.2017.82

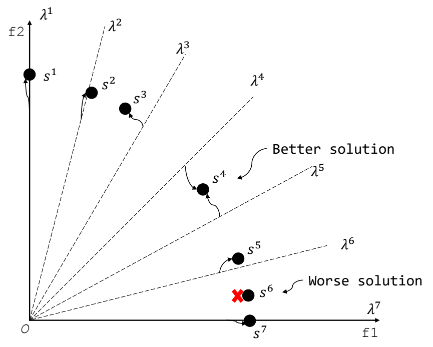

Fig. 1: An illustration of diversity evaluation by coverage method.

If the nearest solution of a reference vector λ is s, we say that s covers λ. In this case, one solution can cover different number of reference vectors and one reference vector can only have one nearest solution. If a solution covers more than one reference vectors, it must be located in a sparse region. The number of reference vectors covered by a solution is called coverage of the solution. For instance, in Fig. 1, the solution s^4 is a good solution because it covers two reference vectors while s^6 is considered as a bad solution as it covers no reference vectors, in terms of diversity.

The pseudocode of set coverage for solutions in S is given by the Algorithm. 1. Each reference vector λ^i finds its closest solution s^{im}, where im is the index of the closest solution. The coverage of solution s^{im}, denoted by c_{im}, is incremented by one accordingly. The closeness between a reference vector λ^i and solution s^j is represented by the angle between them, which is defined as follows:

$$angle(\lambda^i, F(s^j)) = \arccos(\frac{(\lambda^i)^T \cdot (F(s^j) - z^*)}{\|\lambda^i\| \|F(s^j) - z^*\|}) \quad (2)$$

where $z^* = (z_1^*, z_2^*, ..., z_m^*)^T$ is the ideal objective vector with $z_i^* = \min_{x \in \Omega} f_i(x), i = \{1, 2, ..., m\}$.

B. Improving convergence based on cache strategy

In the previous dominance-based MOEAs, to select a solution set S (size of N) from a merged solution set $U(\ |U| > N)$, the Pareto dominance is used as the first selection criterion and diversity maintenance mechanism is applied as the secondary selection criterion. The unselected solutions in U (denoted by $C = U \setminus S$) are discarded. As the selection pressure of Pareto dominance becomes increasingly weak in MaOPs, the secondary diversity maintenance mechanism becomes the major selection operator. Under this circumstance, solutions that have much better convergence and slightly worse diversity are very likely to be discarded. A simple cache strategy is adopted to avoid this issue, where the solutions is saved in an archive (cache set) C, instead of discarding directly. Given a solution set S and a cache set C, the procedure of cache strategy can be described as Algorithm 2.

Algorithm 1: Set Coverage (SETCOVERAGE)

```
Input:
  • Solution set $S : \{s^1, s^2, \ldots, s^N\}$
  • Reference vectors $V : \{\lambda^1, \lambda^2, \ldots, \lambda^M\}$.
Output: $c$.
1  Initialize $c : (c_1, c_2, \ldots, c_N)^T$, and each $c_i = 0$;
2  for $i = 1$ to $M$ do
3      $im = \infty, \theta_m = \infty$;
4      foreach $s^j \in S$ do
5          $\theta_j = angle(s^j, \lambda^i)$;
6          if $\theta_j < \theta_m$ then
7              $im = j$;
8              $\theta_m = \theta_j$;
9          end
10     end
11     $c_{im} = c_{im} + 1$;
12 end
13 return $c$;
```

At first, the norm and radian functions should be defined. In Algorithm 2, the norm of a solution s is defined as the norm from the function vector of s to ideal point which can be formulated as follows:

$$norm(s) = \|F(s) - z^*\| \quad (3)$$

The radian between two solutions, s^i and s^j, is defined as follows accordingly:

$$radian(s^i, s^j) = \min\{norm(s^i), norm(s^j)\} \times angle(s^i, s^j) \quad (4)$$

For each solution s^{ci} in C, its nearest solution s^m in S is found firstly. Then, if $norm(s^{ci})$ is less than $norm(s^m)$ and the difference value of them is larger than the radian of s^m and s^{ci}, then s^m is replaced by s^{ci}. In fact, $norm(s^{ci}) < norm(s^m)$ means that the convergence of s^{ci} is better than s^m; $|norm(s^m) - norm(s^{ci})| > radian(s^m, s^{ci})$ means the improvement of convergence is larger than the degeneration of diversity; thus we use s^{ci} to replace s^m.

Algorithm 2: Cache strategy (CACHE)

```
Input:
  • $S$ : The current population.
  • $C$ : The cache population.
Output: $S$ : The updated population.
1  foreach $s^{ci} \in C$ do
       // find the nearest solution in $S$
2      $s^m$ = argmin$_{s^m \in S}$ $angle(s^m, s^{ci})$;
3      if $norm(s^{ci}) < norm(s^m) \wedge$
4      $|norm(s^m) - norm(s^{ci})| > radian(s^m, s^{ci})$ then
5          $s^m = s^{ci}$;
6      end
7  end
8  return $S$;
```

C. The framework of MaOEA-CC

In this section, the coverage-based diversity maintenance mechanism and cache strategy are integrated into a dominance-based MOEA framework, to address MaOPs. The framework of MaOEA-CC is the same as the classical MOEA NSGA-II [3], which is described as follows. A population S is initialized randomly and then the reproduction and selection methods are applied iteratively until the stop criterion is fulfilled. In MaOEA-CC, SBX crossover [11] and polynomial mutation [12] are used to generate new solutions.

Algorithm 3: Selection procedure (SELECTION)

```
Input:
  • $U$: The merged population;
  • $V$: The reference vector set;
Output: $S$: The elite population.
1  $(F_1, F_2, ...)$ = NONDOMINATED-RANKING(U) ;
2  $S = \Phi, i = 1$;
3  while $|S| + |F_i| \leq N$ do
4      $S = S \cup F_i$ and $i = i + 1$ ;
5  end
6  if $|S| = N$ then
7      return $S$ ;
8  end
9  $F_l = F_i$;
10 c = SETCOVERAGE($S \cup F_l$,V) ;
   /* Descending sort $F_l$ according to $c$
      values. $SF_i$ is the set in which the
      solution have the same $c$.            */
11 $(SF_1, SF_2, ...)$ = SORTING($F_l$);
12 $i = 1$;
13 while $|S| + |SF_i| \leq N$ do
14     $S = S \cup SF_i$ and $i = i + 1$ ;
15 end
16 $SF_l = SF_i$;
17 if $|S| < N$ then
18     $S$ = ANGLESELECTION(S, $SF_l$);
19 end
20 $S$ = CACHE($S$,$F_l \setminus S$) ;
21 return $S$;
```

The main different from NSGA-II is selection procedure which is shown in Algorithm 3. the fast non-dominated sorting [3] is applied firstly to divide the merged population U into a number of fronts $\{F_1, F_2, \dots\}$. For each solution y in F_j, a solution x in F_i is found which dominates y if $i < j$. All the solutions in the same front are nondominated to each other. Each non-dominated front F_i, starting from F_1, is selected one by one to form the offspring population S until $|S| + |F_l| > N$. All the solutions in the last front F_l are not able to be added to S completely; instead, $N - |S|$ solutions are selected from F_l and added to S.

To select solutions from F_l, the coverage for each solution is calculated by calling the Algorithm 1. Then, the solutions in F_l are sorted based on their coverage values in a descending order. It should be noted that some of the solutions may have the same coverage values. So all the solutions in F_l are divided into a group of solution sets $(SF_1, SF_2, ...)$ according to their coverage values. All the solutions in a SF_i have the same coverage values and the solutions in front SF have larger coverage values. After that, solutions from $(SF_1, SF_2, ...)$ are added to S until $|S|+|SF_i| = N$. If $|S|$ is less than N, $N-|S|$ solutions should be selected from the last solution set SF_l.

TABLE I: The population size of different objective problems

# of objectives	3	5	8	10
population size	300	210	156	275

For this purpose, we use the angle-based-selection, which is used in [13]. For the i-th solution in SF, the θ_i is used to record the minimal angle to its nearest solution in S. The solution with maximal θ is added to S and deleted from SF one by one until the population size of S reaches N.

If the angle between each solution in SF and the newly added solution x is less than its previous θ, then the θ is updated by the new angle value.

After the above procedures, the new offspring population S contains N individuals. Furthermore, to maintain better convergence, some solutions from $F_l \setminus S$ is used to replace the solutions in S based on cache strategy, by calling the Algorithm 2.

III. EXPERIMENTAL RESULTS

Two widely used test suites, DTLZ [14] and WFG [15], are used for our empirical studies. The number of objectives are set to 3,5,8 and 10. The population size for different problems are given in Table I.

Two performance metrics, inverted generational distance (IGD) [5] and Hypervolume (HV) [16], are used to compare the performance of the different algorithms.

Three many-objective evolutionary algorithms (MOEA/D [4], NSGA-III [10] and MOEA/DD [7]) are used to compare with MaOEA-CC. For all the four compared MOEAs, the crossover probability p_c is set to 1 and the distribution index η_c is set to 30; the mutation probability p_m is set to $1/l$, where l is the number of decision variables and its distribution index η_m is set to 20. In MOEA/D and MOEA/DD, the neighborhood size T is set to 20, and the penalty parameter θ for PBI is set to 5. The neighborhood selection probability for MOEA/DD is set to 0.9. Each test instance is run 30 times independently and the maximum number of generations is set to 1000.

IGD is used to evaluate the performance of all the compared algorithms on DTLZ test suite while HV is used for WFG test suite, whose true PFs are unknown. [2]

The final ranking according to IGD or HV of all the DTLZ and WFG problems obtained by four algorithms are plotted in Fig. 2. The problem DTLZ1-3 means the 3 objective DTLZ1.

[2]The reference points of HV are set as $(3, 5, ..., 2m + 1)^T$, where m is the number of objectives.

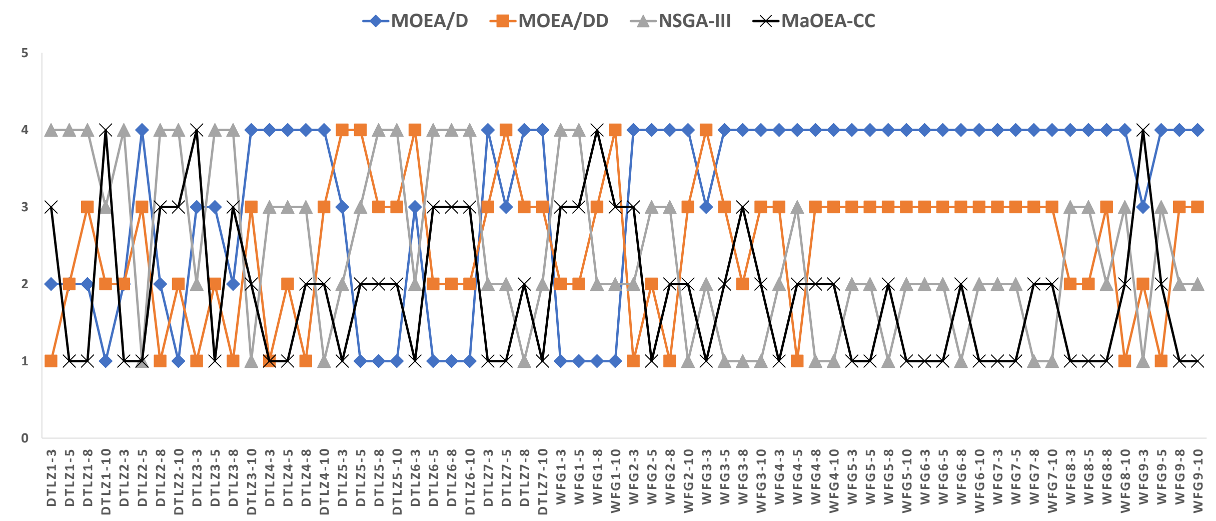

Fig. 2: The ranking of all the DTLZ and WFG problems with different number of objective obtained by 4 algorithms.

It can be seen that MaOEA-CC has the better comprehensive performance. The mean rankings of MOEA/D, MOEA/DD, NSGA-III and MaOEA-CC are 3.1, 2.5 2.4 and 1.9, respectively.

IV. Conclusion

In this paper, we propose an evolutionary many-objective optimization algorithm based on coverage and cache strategy (MaOEA-CC). The cache strategy method is used to improve the convergence by archiving solutions with much better convergence and slightly worse diversity; while the coverage-based method helps maintain better diversity. The experimental results show that the MaOEA-CC has better overall performance than other three state-of-the-art MaOEAs.

Acknowledgment

This work was supported in part by the National Program on Key Basic Research Project (973 Program) under grant 744901, by the National Natural Science Foundation of China (NSFC) under grant 61300159, 61175073 and 61473241, by the Natural Science Foundation of Jiangsu Province under grant BK20130808, by China Postdoctoral Science Foundation under grant 2015M571751 and by the Fundamental Research Funds for the Central Universities of China under grant NZ2013306.

References

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

[2] H. Li and D. Landa-Silva, "An adaptive evolutionary multi-objective approach based on simulated annealing," *Evolutionary Computation*, vol. 19, pp. 561–595, 2011.

[3] 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, pp. 182–197, April 2002.

[4] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, pp. 712–731, December 2007.

[5] P. A. Bosman and D. Thierens, "The Balance Between Proximity and Diversity in Multiobjective Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, pp. 174–188, April 2003.

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

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

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

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

[11] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 3, pp. 115–148, 2010.

[12] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," pp. 30–45, 1996.

[13] 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*, vol. PP, no. 99, pp. 1–14, 2016.

[14] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable Test Problems for Evolutionary Multiobjective Optimization," in *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications* (A. Abraham, L. Jain, and R. Goldberg, eds.), pp. 105–145, USA: Springer, 2005.

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

[16] 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.