An Estimation of Distribution Algorithm for Large-Scale Optimization with Cooperative Co-evolution and Local Search

Jia-Ying Lin, Wei-Neng Chen(✉), and Jun Zhang

South China University of Technology, Guangzhou, China
cwnraul634@aliyun.com

Abstract. Cooperative co-evolution (CC) is an effective framework for evolutionary algorithms (EAs) to solve large-scale optimization problems. By combining a divide-and-conquer strategy and the classic evolutionary algorithms (EA) like genetic algorithm (GA), CC has shown promising performance in many fields. As a family of EAs, the estimation of distribution algorithm (EDA) is good at search diversity maintenance, but its capability in solving large-scale problems has not been fully explored. In this paper, we aim to propose a new estimation of distribution algorithm with the cooperative co-evolution framework (EDACC). The proposed EDACC has the following features. (1) The differential grouping (DG) strategy is applied for variable decomposition. (2) A combination of the Gaussian and Cauchy distributions are adopted to generate offspring. (3) A local search method is performed in promising domains to accelerate the search. To verify the performance of EDACC, experiments are conducted on 20 single-objective functions in the CEC 2010 benchmarks. The experimental results show that EDACC can still achieve competitive performance in spite of the weakness of the original EDAs like the low accuracy in global optima searching compared with classical EAs.

Keywords: Estimation of distribution algorithm (EDA)
Large-scale optimization · Cooperative Co-evolution (CC)

1 Introduction

Many complex and difficult real-world problems such as traffic control [21] and data mining [19] in different areas can be reduced to some large-scale optimization problems. Classical evolutionary algorithms (EAs) like PSO [10], DE [6] and CSO [5], cannot solve these high-dimensional problems perfectly in terms

This work was supported in part by the National Natural Science Foundation of China under Grant 61622206, 61332002, and the Natural Science Foundation of Guangdong under Grant 2015A030306024.

L. Cheng et al. (Eds.): ICONIP 2018, LNCS 11302, pp. 442–452, 2018.
https://doi.org/10.1007/978-3-030-04179-3_39

of effectiveness and feasibility since they lack some specific framework for handling them. Therefore, it is desirable to find some general solution for large-scale optimization problems.

Cooperative Co-evolution [1,7,14] is a framework for improving the efficacy of various evolutionary algorithms in large-scale optimization problems. It employs a divide-and-conquer strategy to divide a large-scale optimization problem into some small-scale problems then uses the methods supposed to solve them adaptively and independently. Hence, as an effective and powerful framework for large-scale optimization problems, cooperative co-evolutionary is widely used in some variations of EAs including Cooperative Co-evolutionary Genetic Algorithm (CCGA) [14] and CPSO [3], taking the advantage of EAs in solving function optimization problems. Recently, some new methods that using new decomposition method called differential grouping have been proposed and performed well in benchmark problems like DECC-DG [13] by efficient decomposition strategy. Many satisfying experimental results have been achieved by embedding evolutionary algorithm into Cooperative Co-evolution framework. However, how to maintain the search diversity under CC framework is still a challenging problem, since the number of local optima will rapidly increase as the dimension of solution space grows higher in a large-scale optimization problem.

To maintain good search diversity, a family of EAs called estimation of distribution algorithms (EDAs) [9,12] has attracted much attention as it can maintain good search diversity at the population level working with probabilistic models. Comparing with EDAs, conventional EAs use crossover and mutation operators to generate new population and trial solutions which may easily get closer to the parents but far away from global solutions. The population level diversity of EDAs allows itself to be a critical role in solving high-dimensional problems among all variants of EAs.

However, keeping diversity at the population level makes EDA hard to directly control similarities among offspring and parent at the population level. To deal with this weakness of EDAs, some efforts have been made by using some local search methods to refine the offspring solution in promising areas of the optimal locations when some solutions are found [23].

In this paper, we aim to propose a new estimation of distribution algorithm with the cooperative co-evolution framework (EDACC). The differential grouping (DG) strategy applied in EDACC performs decomposition of variables. Thus, a large-scale optimization problem can be divided into several small-scale problems and solved efficiently under CC framework. With regard to possibility distribution used in EDA, a combination of the Gaussian and Cauchy distributions are adopted in EDACC to generate offspring with high diversity. In order to improve solution accuracy and accelerate the search, a local search method is performed in promising domains found by EDACC. The experimental results show that EDACC can still achieve competitive performance in spite of the weakness of the original EDAs like the low accuracy in global optima searching compared with classical EAs.

The rest of this paper is organized as follows. Section 2 introduces some related works and the background of this research. Section 3 introduces our estimated distribution algorithm with CC framework and Sect. 4 shows our experiment result.

2 Relate Work

2.1 Estimation of Distribution Algorithm

EDA is a family of evolutionary algorithm firstly proposed in [12], combining classic EAs with probabilistic models to increase diversity in EA offspring generating at the population level. Unlike traditional EAs, EDAs do not use crossover or mutation operators. Instead, the probability model sampled in population level is a role to control the generation of offspring and this model of promising solutions can help keep and improve the diversity in offspring generation. A general framework of EDA is built as Algorithm 1.

Algorithm 1. EDA

```
Input: population size $M$, the number of selected individuals from parent $N$
Output: the best solution and the corresponding fitness
1: while termination criterion is not met do
2:     Select N individuals from population
3:     Estimate the new probability distribution of selected N individuals
4:     Generate offspring by partly replacing parent with individuals sampled by the
       new probability distribution
5: end while
```

EDA has shown its power in both discrete and continuous domains, single and multiple optimization problems [2,11,22]. Recently, some research fields like multi-policy insurance investment [18] planning and protein folding problem [16] start to adapt EDA model to enhance their solution performance due to the feasibility and effectiveness of EDAs. However, few attempts have been made to enhance EDA performance with cooperative co-evolution framework.

2.2 Cooperative Co-evolution

Cooperative Co-evolution is a framework to solve large-scale optimization problems by divide-and-conquer strategy, decomposing the complex, high dimensional problem into simpler, lower dimensional subproblems. The performance of CC is directly influenced by its grouping strategy, which is the principle of generating suitable subproblems. On the basis of the CC framework, different evolutionary algorithms can be used to solve the large-scale optimization problem effectively. Algorithm 2 presents the framework of CC.

Algorithm 2. Cooperative Co-evolution

```
Input: Original population $P$
Output: the best solution and the corresponding fitness
1: Decompose original problem into $K$ subproblems, which $K < P$
2: while termination criterion is not met do
3:     for $i = 1 : K$ do
4:         Solve the subproblem by optimizer
5:         Update current best solution
6:     end for
7: end while
```

2.3 Differential Grouping Strategy

Recently, some grouping strategies are proposed to find the dependence among variables in specific optimization problems under CC framework. These grouping strategies aim to generate a reasonable subset of variables for problem optimizer. Differential Grouping (DG) is a dynamic grouping method dividing the variables of original problems according to the interaction between any two different variables. DG keeps the correlation among subproblems and helps the problem optimizer perform later work.

In DG, the dependence between two variable x_i and variable x_j are detected by following inequation:

$$f(x_i + \delta_i, x_j) - f(x_i, x_j) \neq f(x_i + \delta_i, x_j + \delta_j) - f(x_i, x_j + \delta_j),$$

which $\delta_i, \delta_i \neq 0$. The satisfaction of above inequation demonstrates that two variable x_i and x_j interact.

3 Estimation of Distribution Algorithm with CC Framework

3.1 Distribution Estimation

In EDACC, to preserve the diversity at the population level and make sure promising solution converge, the probabilistic distribution models we use are not only traditional probabilistic model such as Gaussian distribution but also Cauchy distribution model, which can expand the offspring's range because of its long fat tail.

As mentioned in Algorithm 1 line 3, we adopt both Gaussian distribution and Cauchy distribution to EDACC and choose one of them for offspring generation decided by the current promising solution. In EDACC, these two distributions are assigned according to the current decision variable variance of all individuals. If current variance of variables is too small, Cauchy distribution should be assigned since it can offer high diversity for later population and avoid dropping in local optima. In order to accelerate the convergence speed in EDACC, the

Algorithm 3. Offspring generation on EDA

```
Input: Population $P$, Smallest variance $\sigma_s$ that applied Gaussian distribution
Output: Offspring generated by Gaussian distribution or Cauchy distribution
1: Calculate the mean $\mu_{ij}$ and the variance $\sigma_{ij}$ of $j^{th}$ variable
2: if $\sigma_{ij} < \sigma_s$ then
3:     Generate corresponding variable offspring by Cauchy$(\mu_{ij}, scale)$, where scale is
       a parameter of Cauchy distribution.
4: else
5:     Generate corresponding variable offspring by Gaussian distribution$(\mu_{ij}, \sigma_{ij})$
6: end if
7: Update current population
```

threshold of variance assigning these two distributions should be set to a smaller value.

μ_i stands for the mean value of x_i decision variable among all j individuals and σ_i stands for the variance of x_i decision variable among all j individuals. It should be noted that because of the characteristic of Cauchy distribution, we select μ_i as its median and σ_i as its scale parameter.

3.2 Cooperative Co-evolution

One important factor of EDA offspring generation is the way we select individuals and decision variables for probabilistic distribution model simulation. The problem of selecting individuals can be solved by original EDA like PBILc, while the selection of decision variables is still required developed in EDA algorithm. CC framework gives us a new insight for selecting subproblems in the original problem and DG offers a great grouping strategy for CC framework.

Taking all these factors into consideration, we propose a new generating offspring method in EDA under integrating CC framework and DG grouping strategy. In population updating procedure, EDACC doesn't take all decision variables in each selected individuals for the offspring generation. According to the decision variable division result using DG grouping method, we generate new offspring for each decision variable in one group together, while different decision variable groups are updated separately.

We denote that x_{ij} stands for the j^{th} decision variables in i^{th} individual. In offspring generation, for K selected individuals like $(x_1, x_2, \ldots, x_K)$, each individual contains M decision variables and the decision variables only required for probabilistic model parameters updating are those variables in the same subgroup generated by DG strategy. Since DG strategy can reallocate interacting decision variables into the same subgroup, it's necessary to generate offspring one group by one group for EDA offspring generating integrating CC framework. The interaction among population and the characteristic of EDA is suitable for CC framework and DG strategy in solving a large-scale optimization problem by a divide-and-conquer method.

3.3 Local Search

Some previous researches [18,22] in EDA have shown that original EDA may be poor at finding a high accuracy solution due to its offspring generating strategy. In EDACC, since the promising solution area can be found approximately, an efficient local search method can be used in the final procedure of generating the best solution when the current solution can be guaranteed converged. Powell method [8] is a classic optimization method without derivatives. It uses bi-directional search along each search vector and ensures efficiency during its searching. This characteristic is suitable for some method that solving large-scale optimization problems with high diversity but low accuracy like EDA. The total EDACC method is presented as Algorithm 4.

Algorithm 4. EDACC method

```
Input: Population $P$, Smallest variance $\sigma_s$ that applied Gaussian distribution
Output: the best solution and the corresponding fitness
 1: Decompose original problem into $K$ subproblems by DG strategy, which $K < P$
 2: while termination criterion is not met do
 3:    for $i = 1 : K$ do
 4:       for $j = 1 : M$, where $M$ is the problem size of $i$(the number of variables in
          specific subproblem $i$) do
 5:          Calculate the mean $\mu_{ij}$ and the variance $\sigma_{ij}$ of $j^{th}$ variable
 6:          if $\sigma_{ij} < \sigma_s$ then
 7:             Generate corresponding variable offspring by Cauchy($\mu_{ij}, scale$),
                where scale is a parameter of Cauchy distribution.
 8:          else
 9:             Generate corresponding variable offspring
                by Gaussian distribution($\mu_{ij}, \sigma_{ij}$)
10:          end if
11:          Update current best solution
12:       end for
13:       Use local search method
14:       Update current best solution
15:    end for
16: end while
```

4 Experiment Studies

In this section, a series of experiments are conducted on a large-scale optimization benchmark called CEC2010 [4] to verify the performance of EDACC. Then we compare the result of EDACC with other classic methods in different aspects of optimizer and grouping strategy. Some experimental analysis of EDACC will be demonstrated. All the algorithms are implemented in C and executed in Hasee K660 with i5-4210M CPU @ 2.60 GHz, 8.00 GB RAM, and Windows 10.

4.1 Parameter Setting

In EDACC, population size we set is 150 and the maximum number of function evaluation is 3.0E+6. The optimizer we use in EDACC is PBILc [17], which has an offspring update formula:

$$p_{l+1}(x) = (1-\alpha)p_l(x) + \frac{\alpha}{N}\sum_{k=1}^{K} x_k,$$

denoted that l is l^{th} iteration and x_k is the k^{th} best individuals. The reason why we use PBILc as optimizer is that PBILc is a simple and classical method among variants of EDA. In EDACC, α is set to 0.5 and K is set to 10. Besides, taking diversity into consideration, we add a truncated ratio r_t equals to 0.3 on the population level updated by EDA. The threshold of Gaussian distribution and Cauchy distribution on the variable variance is set to 1.0E−9, which is a small value that makes sure to convergence of the solution.

In local search part, we use a modified Powell method to make CEC2010 enhance solution accuracy and help EDACC converge quickly in a promising solution area. Comparing with traditional Powell method proposed in [15], the parameter in Powell method controlling accuracy is adjusted to a smaller value. For instance, the parameter TOL (a distance tol from a point already evaluated) are set to 2.0E−10.

4.2 Experiment Result

Table 1 presents the average results over 25 independent runs of EDACC, DECC-DG and DECC-XDG on CEC2010 benchmark. What should be highlighted is that the reason we compare EDACC with DECC-DG is to study how different evolutionary algorithms such as EDA and SaNSDE under CC framework. We use a traditional optimizer of EDA in EDACC, PBILc, to decrease the effect of the optimizer in EDA. Table 1 shows that EDACC significantly performs well in non-separable function like f_1, f_2 and f_{20}, especially in f_{20} which DG performs a bad grouping result. To study how EDA can overcome the weakness in grouping of DG, some functions that are hard to do grouping, are especially studied as follows. Table 1 also demonstrates that EDACC has a significantly better performance than DECC-DG especially on those benchmark functions which has bad grouping result by DG such as f_{13}, f_{18} and f_{20}. These benchmark functions are hard to adjust ϵ to get a good grouping result, according to the study of DG. Due to the low sensitivity of differential grouping result, EDACC can obtain a better result in spite of low efficacy of its optimizer. Besides, as argued in XDG [20], XDG can achieve 100% grouping accuracy on all of CEC2010 20 benchmark functions, so the comparison on the experiment result between EDACC and DECC-DG is more believable on the study of different grouping result. Although EDACC perform a little worse than DECC-XDG, it still can achieve a better result on f_1, f_2, f_4, f_9, f_{14}, f_{17} and f_{20}, where also outperforms DECC-DG. For fully separable functions, EDACC also outperforms significantly

Table 1. Comparison Result of EDACC, DECC-DG and DECC-XDG on CEC10'

Functions	EDACC		DECC-DG		DECC-XDG	
	Mean	Std	Mean	Std	Mean	Std
f_1	**2.84E–26**	5.98E−26	5.47E+03	2.02E+04	2.23E+04	8.01E+04
f_2	**2.66E+03**	2.17E+02	4.39E+03	1.97E+02	4.44E+03	1.64E+02
f_3	1.94E+01	4.95E−02	**1.66E+01**	3.34E−01	1.68E+01	4.12E−01
f_4	**4.89E+11**	2.30E+11	4.79E+12	1.44E+12	7.84E+11	1.67E+11
f_5	3.69E+08	7.77E+07	**1.55E+08**	2.17E+07	1.68E+08	1.74E+07
f_6	1.97E+07	8.80E+04	**1.64E+01**	2.71E−01	**1.63E+01**	3.28E−01
f_7	1.98E+09	2.65E+09	1.16E+04	7.41E+03	**1.39E+03**	2.61E+03
f_8	1.18E+08	1.63E+08	3.04E+07	2.11E+07	**4.78E+05**	1.32E+06
f_9	**6.92E+06**	1.79E+06	5.96E+07	8.18E+06	1.12E+08	1.13E+07
f_{10}	8.52E+03	2.41E+02	**4.52E+03**	1.41E+02	5.31E+03	1.55E+02
f_{11}	2.18E+02	2.36E−01	**1.03E+01**	1.01E+00	**1.04E+01**	1.15E+00
f_{12}	4.94E+04	2.71E+04	**2.52E+03**	4.86E+02	1.24E+04	2.32E+03
f_{13}	3.00E+05	2.92E+05	4.54E+06	2.13E+06	**1.12E+03**	2.25E+02
f_{14}	**1.45E+07**	2.98E+06	3.41E+08	2.41E+07	5.83E+08	4.11E+07
f_{15}	1.45E+04	4.29E+02	**5.88E+03**	1.03E+02	**5.91E+03**	7.56E+01
f_{16}	3.97E+02	3.73E−01	**7.39E−13**	5.70E−14	1.81E−08	1.57E−09
f_{17}	**1.24E+04**	2.65E+04	4.01E+04	2.85E+03	1.26E+05	7.47E+03
f_{18}	5.78E+05	3.33E+05	1.11E+10	2.04E+09	**1.41E+03**	1.88E+02
f_{19}	6.53E+06	2.16E+06	1.74E+06	9.54E+04	**1.59E+06**	4.96E+04
f_{20}	**1.04E+02**	1.10E+02	4.87E+07	2.27E+07	5.55E+05	1.75E+06

in f_1 and f_2 although putting all of the separable decision variables into the same subgroup is not a good choice as XDG studies. The reason may be that EDACC can give a reliable promising solution area and the way generating offspring can maintain high diversity in population level. Besides, as argued in DG, on instances of rotated elliptic function such as f_4, f_9 and f_{14}, an observation is that EDACC shows a better performance though DG and XDG also can get an optimal grouping.

Figure 1 shows the converging curves of EDACC on f_{20}. From Fig. 1, as can be seen, the fitness value evaluated by EDACC outperforms other methods significantly. EDACC can have achieved lower fitness in the 5.0e+05 evaluation comparing with the final fitness evaluated by DECC-DG and DECC-XDG. This result can show EDACC can also obtain good performance in some complex fully-nonseparable function in large-scale optimization problems, which may be associated with its high diversity in population level and CC framework's divide-and-conquer strategy used in it.

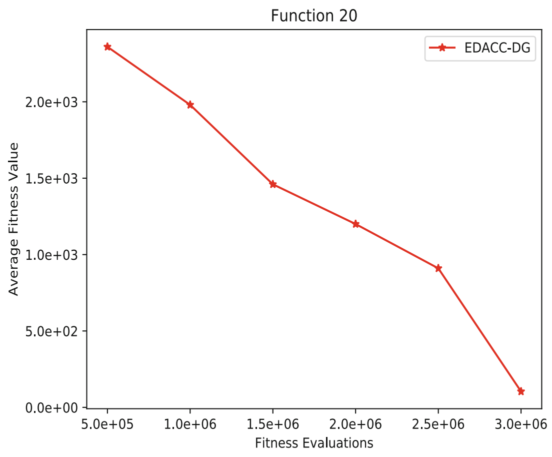

Fig. 1. The converging curves on f_{20} for EDACC

5 Conclusion

In this paper, we have proposed an estimation of distribution algorithm for large-scale optimization with cooperative co-evolution and local search (EDACC). Taking advantages of EDA, EDACC, which firstly combines CC framework with EDA can outperform in specific problems with a competitive final result comparing with DECC-DG and DECC-XDG. In EDACC, a simple optimizer used in its EDA and grouping strategy used in its CC framework, with low accuracy in some functions, can also keep outperforming in some large-scale optimization problems.

This work has shown its given competitive result in large-scale optimization problems. However, taking consideration of EDA we used in EDACC, the optimizer and the grouping strategy are still required to be investigated how to achieve higher performance incorporating CC framework with EDA.

References

1. Antonio, L.M., Coello, C.A.C.: Use of cooperative coevolution for solving large scale multiobjective optimization problems. In: 2013 IEEE Congress on Evolutionary Computation (CEC), pp. 2758–2765. IEEE (2013)
2. Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A.: Estimation of distribution algorithms: a new evolutionary computation approach for graph matching problems. In: Figueiredo, M., Zerubia, J., Jain, A.K. (eds.) EMMCVPR 2001. LNCS, vol. 2134, pp. 454–469. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44745-8_30
3. Van den Bergh, F., Engelbrecht, A.P.: A cooperative approach to particle swarm optimization. IEEE Trans. Evol. Comput. **8**(3), 225–239 (2004)

4. Chen, W., Weise, T., Yang, Z., Tang, K.: Large-scale global optimization using cooperative coevolution with variable interaction learning. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN 2010. LNCS, vol. 6239, pp. 300–309. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15871-1_31
5. Cheng, R., Jin, Y.: A competitive swarm optimizer for large scale optimization. IEEE Trans. Cybern. **45**(2), 191–204 (2015)
6. Das, S., Suganthan, P.N.: Differential evolution: a survey of the state-of-the-art. IEEE Trans. Evol. Comput. **15**(1), 4–31 (2011)
7. Fan, J., Wang, J., Han, M.: Cooperative coevolution for large-scale optimization based on kernel fuzzy clustering and variable trust region methods. IEEE Trans. Fuzzy Syst. **22**(4), 829–839 (2014)
8. Fletcher, R.: Practical Methods of Optimization. Wiley, Hoboken (2013)
9. Hauschild, M., Pelikan, M.: An introduction and survey of estimation of distribution algorithms. Swarm Evol. Comput. **1**(3), 111–128 (2011)
10. Kennedy, J.: Particle swarm optimization. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 760–766. Springer, Boston (2011)
11. Luo, N., Qian, F.: Estimation of distribution algorithm sampling under Gaussian and Cauchy distribution in continuous domain. In: 2010 8th IEEE International Conference on Control and Automation (ICCA), pp. 1716–1720. IEEE (2010)
12. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: Voigt, H.-M., Ebeling, W., Rechenberg, I., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 178–187. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61723-X_982
13. Omidvar, M.N., Li, X., Mei, Y., Yao, X.: Cooperative co-evolution with differential grouping for large scale optimization. IEEE Trans. Evol. Comput. **18**(3), 378–393 (2014)
14. Potter, M.A., De Jong, K.A.: A cooperative coevolutionary approach to function optimization. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN 1994. LNCS, vol. 866, pp. 249–257. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58484-6_269
15. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes 3rd Edition: The Art of Scientific Computing. Cambridge University Press, New York (2007)
16. Santana, R., Larrañaga, P., Lozano, J.A.: Protein folding in simplified models with estimation of distribution algorithms. IEEE Trans. Evol. Comput. **12**(4), 418–438 (2008)
17. Sebag, M., Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 418–427. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0056884
18. Shi, W., Chen, W.N., Lin, Y., Gu, T., Kwong, S., Zhang, J.: An adaptive estimation of distribution algorithm for multi-policy insurance investment planning. IEEE Trans. Evol. Comput. (2017)
19. Srinivasa, K., Venugopal, K., Patnaik, L.M.: A self-adaptive migration model genetic algorithm for data mining applications. Inf. Sci. **177**(20), 4295–4313 (2007)
20. Sun, Y., Kirley, M., Halgamuge, S.K.: Extended differential grouping for large scale global optimization with direct and indirect variable interactions. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 313–320. ACM (2015)

21. Teklu, F., Sumalee, A., Watling, D.: A genetic algorithm approach for optimizing traffic control signals considering routing. Comput. Aided Civ. Infrastruct. Eng. **22**(1), 31–43 (2007)
22. Yang, Q., Chen, W.N., Li, Y., Chen, C.P., Xu, X.M., Zhang, J.: Multimodal estimation of distribution algorithms. IEEE Trans. Cybern. **47**(3), 636–650 (2017)
23. Zhou, A., Sun, J., Zhang, Q.: An estimation of distribution algorithm with cheap and expensive local search methods. IEEE Trans. Evol. Comput. **19**(6), 807–822 (2015)