Success-History Based Parameter Adaptation in MOEA/D Algorithm

Shakhnaz Akhmedova(✉) and Vladimir Stanovov

Reshetnev Siberian State University of Science and Technology, "Krasnoyarskiy Rabochiy" av. 31, 660037 Krasnoyarsk, Russia
shahnaz@inbox.ru, vladimirstanovov@yandex.ru

Abstract. In this paper two parameter self-adaptation schemes are proposed for the MOEA/D-DE algorithm. These schemes use the fitness improvement ration to change four parameter values for every individual separately, as long as in the MOEA/D framework every individual solves its own scalar optimization problem. The first proposed scheme samples new values and replaces old values with new ones if there is an improvement, while the second one keeps a set of memory cells and updates the parameter values using the weighted sum. The proposed methods are testes on two sets of benchmark problems, namely MOEADDE functions and WFG functions, IGD and HV metrics are calculated. The results comparison is performed with statistical tests. The comparison shows that the proposed parameter adaptation schemes are capable of delivering significant improvements to the performance of the MOEA/D-DE algorithm. Also, it is shown that parameter tuning is better than random sampling of parameter values. The proposed parameter self-adaptation techniques could be used for other multi-objective algorithms, which use MOEA/D framework.

Keywords: Multi-objective optimization · Differential evolution · Parameter adaptation · Self-adaptation · MOEA/D

1 Introduction

The multi-objective and many-objective optimization techniques based on evolutionary algorithms (EA) and swarm intelligence algorithms (SI) have proved to be efficient for solving complex problems due to their population-based nature. The success of SPEA2 [1], NSGA-II [2], MOEA/D [3] and other methods developed later have shown that it is possible to find representative sets of points of the true Pareto set even for many objectives. However, very small attention has been paid to the underlying optimization techniques, used in the multi-objective optimization algorithms. In most cases, the multi-objective evolutionary algorithm relies on problem specific operators, for example, simulated binary crossover (SBX) and polynomial mutation [4, 5] form genetic algorithms (GA) or differential evolution (DE) mutation and crossover operators [6] in case of numerical optimization.

Y. Tan et al. (Eds.): ICSI 2020, LNCS 12145, pp. 455–462, 2020.
https://doi.org/10.1007/978-3-030-53956-6_41

The search operators taken from single-objective optimization algorithms have a set of parameter values, which influence the algorithm efficiency, and for single objective optimization various self-adaptation schemes have been proposed [7]. However, applying these techniques to multi-objective optimization is problematic due to difficulties in estimating the parameters influence on the algorithm performance. Moreover, MOEA aims to find a set of points, each in its own region of search space, while single-objective algorithms need to find only a single solution. To solve the mentioned problem some research have been made about adaptive operator selection (AOS) in MOEAs, for example JADE2 [8], MOSaDE [9], MODE/SN [10] and MOEA/D-FRRMAB [11].

In this study the scheme of operator efficiency estimation proposed in [11] is used to adaptively tune the search parameters of DE and GA operators, with a set of parameters kept separately for every point. The MOEA/D-DE algorithm is taken as baseline, and two approaches are proposed: the self-adaptation (SA) similar to the one used in [12], and the success-history adaptation, proposed in [13]. The experiments are performed on DTLZ and WFG sets of problems, the results are compared with statistical tests.

The rest of the paper is organized as follows: Sect. 2 describes the MOEA/D framework and search operators, Sect. 3 proposes the new parameter adaptation schemes, Sect. 4 contains experimental setup and results, and Sect. 5 concludes the paper.

2 MOEA/D Algorithm and Self-tuning

The multi-objective evolutionary algorithm based on decomposition (MOEA/D) was originally proposed in [3]. The main idea of this approach was to decompose the initial problem into a set of scalar problems. The multi-objective optimization problem (MOP) is formulated as follows:

$$\begin{array}{c} minimize\, F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\ subject\ to\ x \in \Omega \end{array}, \tag{1}$$

where $\Omega \subset R^m$ is the variable space, x is a solution, and F: $\Omega \rightarrow R^m$ is a vector-function to be optimized.

The MOEA/D framework proposes several methods of problem decomposition, including weighted sum, Tchebycheff and penalized boundary intersection approaches [3]. The Tchebycheff approach is one of the commonly used methods, with scalar optimization problems defined as follows:

$$minimize\ g\left(x|\lambda, z^*\right) = \max_{1 \le i \le m} \left\{\lambda_i \left|f_i(x) - z_i^*\right|\right\}, \tag{2}$$

where λ is a weight vector, z* is a reference point. The MOEA/D algorithm defines a set of weight vectors λ^j, j = 1 … N, N is the population size. In this manner, the algorithm optimizes N scalar problems in one run, allowing finding representative distribution of points in the Pareto front, if the corresponding weight vectors are evenly distributed.

The variation operators, used in MOEA/D, should it be the SBX crossover or DE mutation operators, use a neighborhood of size T, which is defined based on distances between the weight vectors λ. So, for every individual i = 1…N a set of vectors B(i) = $\{i_1, \ldots, i_T\}$, where ($\lambda^{i1} \ldots \lambda^{iT}$) are T closest vectors to λ^i. In case of SBX crossover

one of B(i) is chosen, and in case of DE all 3 vectors are chosen from B(i) to perform mutation.

The implementation of self-tuning in multi-objective framework requires the definition of improvement rates, i.e. feedback values, which could be used to drive parameters towards optimal values at every stage of the search. In [14] the fitness improvement rates (FIR) were proposed to solve this problem, defined as follows:

$$FIR_{i,t} = \frac{pf_{i,t} - cf_{i,t}}{pf_{i,t}}, \tag{3}$$

where $pf_{i,t}$ is the fitness of parent, $cf_{i,t}$ is the fitness of child for individual i at step t. As long as MOEA/D solves a set of N scalar optimization problems, it is possible to calculate improvements rates in a similar manner to single-objective optimization algorithms.

Further in this study the DE will be used as the main optimization engine. The main idea of DE is to use the scaled difference vectors between the members of the population to produce new solutions. Classical DE uses rand/1 mutation strategy:

$$v_{i,j} = x_{r1} + F\left(x_{r2,j} - x_{r3,j}\right), \tag{4}$$

where r1, r2 and r3 are mutually different indexes chosen from B(i), and F is the scaling factor, v_i is the mutant vector. The crossover is performed with probability Cr:

$$u_{i,j} = \begin{cases} v_{i,j} if\ rand\ (0, 1) < c_r\ or\ j = j\ rand \\ x_{i,j} \qquad\qquad otherwise \end{cases}, \tag{5}$$

where *jrand* is the randomly selected index from [1, *D*], required to make sure that at least one coordinate is inherited from the mutant vector, u_i is the trial vector.

The polynomial mutation is performed with probability pm and scale parameter η. This operator is applied after the crossover step to produce an offspring. Totally, there are 4 numeric parameters to be tuned: scaling factor F, crossover rate Cr, mutation probability pm and mutation parameter η.

The next section contains the description of the proposed self-adaptation schemes for the MOEA/D-DE algorithm.

3 Proposed Parameter Self-adaptation Schemes

Among single-objective EAs, it is well known that adaptation of parameter values to the problem in hand is an important part of every algorithm. However, for multi-objective algorithms the typical scenario is to use fixed parameters, or probably use several operators, as described in [14] to adapt the algorithm to the problem being solved. One more important difference of MOPs is that each point in the population is a part of the final solution, and its optimal position is in an area of search space, different from other areas, so that the properties of fitness functions landscape could vary significantly for every individual, especially for decomposition-based approaches.

Taking into account these considerations, one may come up with an idea of parametric adaptation, where every individual has its own set of parameter values, which could be

considered as suboptimal for the problem in hand. Similar are already known from literature, where in jDE algorithm [12] pairs of F and Cr parameters were stored for every individual separately. As the search proceeded, these F and Cr were updated in pursuit of finding best possible values. Another well-known approach for parameter adaptation was proposed in the SHADE algorithm [13]. In SHADE algorithm, as well as those developed based on it, a set of H memory cells (usually $H = 5$) is maintained to keep the most suitable parameter values. The memory cells are used to sample new parameter values, and are updated with respect to the fitness improvement rates.

Based on these ideas, the first parameter self-adaptation method is proposed. For every individual $i = 1, \ldots, N$ in the population the a set of parameter values [F_i, Cr_i, pm_i, η_i] is maintained. Initially all these memory cells set to fixed values, for example [0.5, 1, 1/D, 20]. For every individual, new parameters are sampled using normal distribution:

$$\begin{aligned} F_i^{new} &= rnorm(F_i,\ 0.2) \\ Cr_i^{new} &= rnorm(Cr_i,\ 0.2) \\ pm_i^{new} &= rnorm(pm_i,\ 0.2) \\ \eta_i^{new} &= rnorm(\eta_i,\ 5) \end{aligned} \tag{6}$$

If some of the parameters were sampled below 0, then they were sampled again. However, if F, Cr or pm were above 1, then the value of 1 was kept.

After the application of combination and variation operators, the fitness value of the child is compared to the parent's fitness, and if there is an improvement, then the new parameter values replace the old ones. This approach will be further referred to as MOEA/D-DE-SA.

The second self-adaptation approach uses the idea of success-history based adaptation, where the parameter values are averaged over several last steps. For every individual a set of $H = 5$ memory cells is maintained, with $MF_{i,h}$, $MCr_{i,h}$, $Mpm_{i,h}$ and $M\eta_{i,h}$ values, $i = 1, \ldots, N$, $h = 1, \ldots, H$. Memory cell initialization and sampling is performed in the same way as in MOEA/D-DE-SA, however, the update scheme is changed. The FIR values from Eq. 3 are calculated and used as weights $w_{i,h}$ for the update, performed as follows:

$$\begin{aligned} F_i^{new} &= \tfrac{1}{2}\left(F_i^{old} + \frac{\sum_{k=1}^{H} w_{i,k}(MF_{i,k}^{new})^2}{\sum_{k=1}^{H} w_{i,k} MF_{i,k}^{new}}\right) \\ \eta_i^{new} &= \tfrac{1}{2}\left(\eta_i^{old} + \frac{\sum_{k=1}^{H} w_{i,k} M\eta_{i,k}^{new}}{\sum_{k=1}^{H} w_{i,k}}\right) \end{aligned}, \tag{7}$$

where F_i^{new} and F_i^{old} are the new and old values of F parameter. The update scheme for Cr and pm parameters is the same as for F. Note that if there was no improvement, i.e. FIR $<$ 0, the weight was set to zero, and if all weights were zero, there was no parameter update. The h index responsible for the memory cell index to write the temporary parameter values is incremented every generation, and if the h value exceeds $H = 5$, then it is reset back to 1. This algorithm will be further referred to as MOEA/D-DE-SHA.

In addition to the described algorithm, the approach with random sampling was used, i.e. the parameters were generated as shown in Eq. 6, but no memory cells update schemes were applied. This approach will be further referred to as MOEA/D-DE-RS. The experimental setup and results are presented in the next section.

4 Experimental Setup and Results

The experiments were performed on a set of benchmark problems proposed in [14] for the MOEA/D-DE algorithm, as well as on a set of WFG problems [15]. The population size was set to 100, actual population size depended on the number of objectives. The maximum number of function evaluations was set to 10000 for all problems. The algorithms were implemented using the PlatEMO 2.5 system [16].

Table 1. Comparison of MOEA/D to the proposed approaches, MOEA/D-DE-SHA as baseline, IGD metric.

Problem	MOEA/D-DE	MOEA/D-DE-RS	MOEA/D-DE-SA	MOEA/D-DE-SHA
MOEADDE1	**1.061e-2 (3.59e-3) +**	1.096e-2 (2.55e-3) =	1.162e-2 (2.88e-3) =	1.210e-2 (3.72e-3)
MOEADDE2	1.579e-1 (3.61e-2) -	1.013e-1 (2.23e-2) -	9.355e-2 (1.94e-2) =	**9.309e-2 (1.48e-2)**
MOEADDE3	1.097e-1 (4.32e-2) -	9.156e-2 (4.22e-2) -	**6.614e-2 (1.63e-2) =**	6.631e-2 (2.81e-2)
MOEADDE4	1.103e-1 (3.72e-2) -	9.483e-2 (3.18e-2) =	8.092e-2 (2.49e-2) =	**7.945e-2 (1.61e-2)**
MOEADDE5	7.272e-2 (2.41e-2) -	**5.975e-2 (1.51e-2) =**	6.054e-2 (1.33e-2) =	6.037e-2 (1.78e-2)
MOEADDE6	1.572e-1 (3.64e-2) -	1.583e-1 (4.37e-2) =	1.452e-1 (2.84e-2) =	**1.435e-1 (3.25e-2)**
MOEADDE7	3.489e-1 (1.14e-1) =	3.762e-1 (1.42e-1) =	**2.612e-1 (9.58e-2) +**	3.122e-1 (6.91e-2)
MOEADDE8	2.462e-1 (5.23e-2) +	2.831e-1 (6.11e-2) =	**2.457e-1 (5.74e-2) +**	3.127e-1 (5.96e-2)
MOEADDE9	1.503e-1 (2.74e-2) -	1.076e-1 (2.61e-2) -	9.414e-2 (2.06e-2) =	**9.359e-2 (2.66e-2)**
WFG1	1.546e+0 (3.10e-2) -	1.452e+0 (7.00e-2) -	1.368e+0 (8.58e-2) =	**1.338e+0 (8.86e-2)**
WFG2	3.593e-1 (2.96e-2) =	**3.372e-1 (1.61e-2) +**	3.563e-1 (2.20e-2) =	3.514e-1 (2.00e-2)
WFG3	2.759e-1 (3.74e-2) -	2.073e-1 (2.55e-2) -	1.919e-1 (2.13e-2) =	**1.837e-1 (1.87e-2)**
WFG4	3.962e-1 (1.59e-2) =	3.941e-1 (1.06e-2) +	**3.929e-1 (1.43e-2) +**	4.022e-1 (1.40e-2)
WFG5	**3.362e-1 (4.97e-3) =**	3.369e-1 (6.47e-3) =	3.363e-1 (5.43e-3) =	3.373e-1 (6.31e-3)
WFG6	4.385e-1 (2.14e-2) =	4.405e-1 (2.04e-2) =	**4.339e-1 (1.57e-2) +**	4.414e-1 (1.46e-2)
WFG7	3.839e-1 (1.25e-2) -	3.743e-1 (9.08e-3) =	**3.730e-1 (9.69e-3) =**	3.742e-1 (1.01e-2)
WFG8	4.878e-1 (4.08e-2) -	4.479e-1 (2.00e-2) -	4.334e-1 (1.30e-2) =	**4.302e-1 (1.05e-2)**
WFG9	**3.468e-1 (9.42e-3) =**	3.489e-1 (2.04e-2) =	3.613e-1 (3.18e-2) =	3.557e-1 (2.78e-2)
Total	2+/6=/10-	2+/10=/6-	4+/14=/0-	

For every test function 30 independent runs were performed, and the inverted generational distance (IGD) and hypervolume (HV) metrics were calculated. All WFG functions had m = 3 objectives, while all MOEA/D-DE functions had 2 objectives except for F6, which had 3 objectives.

To compare the performance of different algorithms, the Wilcoxon rank sum statistical test with significance level p = 0.05 was used. Table 1 shows the IGD values for all test problems, and Table 2 shows HV values.

The best average values in Tables 1 and 2 are marked. If the MOEA/D-DE-SHA was significantly better than the other algorithm, then the "+" sign was used, if worse, then "−" sign, otherwise "=". From Tables 1 and 2 it could be seen that MOEA/D-DE-SA and MOEA/D-DE-SHA outperform both standard MOEA/D-DE with fixed parameters and the MOEA-D/DE-RS with random sampling. However, even the random sampling is most of the times better than fixed parameter values.

Comparing MOEA/D-DE-SA and MOEA/D-DE-SHA, in terms of IGD metric the former has shown better results, i.e. significantly better on 4 problems out of 18, but for HV metric the two approaches have similar performance, i.e. 3 wins and 3 losses. But, it

Table 2. Comparison of MOEA/D to the proposed approaches, MOEA/D-DE-SHA as baseline, HV metric.

Problem	MOEA/D-DE	MOEA/D-DE-RS	MOEA/D-DE-SA	MOEA/D-DE-SHA
MOEADDE1	7.0899e-1 (5.55e-3) =	**7.0944e-1 (3.78e-3) +**	7.0872e-1 (3.68e-3) =	7.0805e-1 (5.07e-3)
MOEADDE2	5.0146e-1 (3.85e-2) -	5.6748e-1 (3.07e-2) =	5.7485e-1 (3.01e-2) =	**5.7629e-1 (2.39e-2)**
MOEADDE3	6.1718e-1 (2.60e-2) -	6.2621e-1 (2.48e-2) -	6.4138e-1 (1.22e-2) =	**6.4330e-1 (1.66e-2)**
MOEADDE4	6.1748e-1 (2.52e-2) -	6.2907e-1 (1.83e-2) =	6.3283e-1 (1.82e-2) =	**6.3679e-1 (1.13e-2)**
MOEADDE5	6.4418e-1 (1.38e-2) =	**6.5126e-1 (9.65e-3) =**	6.4800e-1 (1.12e-2) =	6.5056e-1 (1.22e-2)
MOEADDE6	3.8415e-1 (3.59e-2) =	3.8533e-1 (3.73e-2) =	3.9858e-1 (2.53e-2) =	**3.9884e-1 (3.00e-2)**
MOEADDE7	2.6923e-1 (8.38e-2) =	2.6947e-1 (7.96e-2) =	**3.7552e-1 (1.17e-1) +**	2.9384e-1 (7.79e-2)
MOEADDE8	3.6255e-1 (7.55e-2) +	3.1275e-1 (8.94e-2) +	**3.7399e-1 (8.55e-2) +**	2.6100e-1 (8.97e-2)
MOEADDE9	2.2668e-1 (3.47e-2) -	2.7780e-1 (4.21e-2) -	2.9973e-1 (3.10e-2) =	**3.0060e-1 (3.72e-2)**
WFG1	2.6741e-1 (1.49e-2) -	2.9960e-1 (2.32e-2) -	3.2444e-1 (2.71e-2) =	**3.3359e-1 (2.83e-2)**
WFG2	8.5717e-1 (1.60e-2) -	**8.6870e-1 (9.08e-3) =**	8.6731e-1 (8.38e-3) =	8.6816e-1 (8.78e-3)
WFG3	2.8778e-1 (2.32e-2) -	3.3166e-1 (1.32e-2) -	3.3926e-1 (9.56e-3) -	**3.4490e-1 (8.68e-3)**
WFG4	4.4927e-1 (7.59e-3) -	4.6641e-1 (7.35e-3) -	4.8247e-1 (5.48e-3) -	**4.8606e-1 (6.45e-3)**
WFG5	4.5894e-1 (4.32e-3) =	4.5837e-1 (3.81e-3) =	4.6025e-1 (4.28e-3) =	**4.6048e-1 (3.84e-3)**
WFG6	3.8575e-1 (2.20e-2) +	3.8426e-1 (2.68e-2) =	**4.0163e-1 (3.33e-2) +**	3.7719e-1 (2.12e-2)
WFG7	4.5816e-1 (9.43e-3) -	4.8504e-1 (6.75e-3) -	4.9494e-1 (6.22e-3) =	**4.9605e-1 (5.45e-3)**
WFG8	3.4231e-1 (2.05e-2) -	3.8402e-1 (1.11e-2) -	3.9655e-1 (7.70e-3) -	**4.0106e-1 (5.33e-3)**
WFG9	4.5905e-1 (1.13e-2) +	**4.6384e-1 (2.49e-2) =**	4.5031e-1 (4.10e-2) =	4.5777e-1 (3.48e-2)
Total	3+/5=/10-	2+/9=/7-	3+/12=/3-	

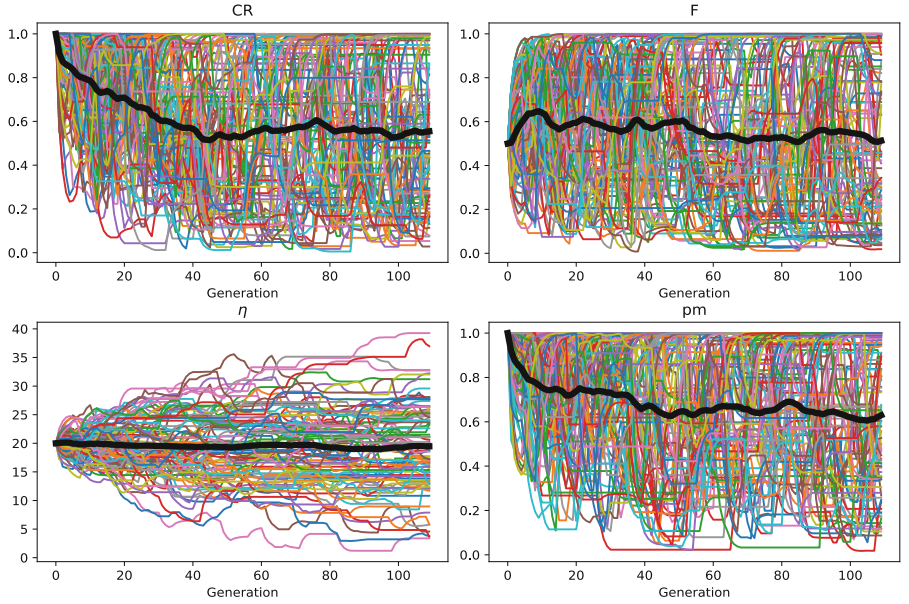

Fig. 1. Change of parameters of the MOEA/D-DE-SHA algorithm during one of the runs, WFG8 problem.

is important to mention that MOEA/D-DE-SHA has achieved better IGD and HV more often than other methods.

Figure 1 shows the change of parameters of the MOEA/D-DE-SHA algorithm during one of the runs. Cr and pm parameters start from 1 and gradually reduce to around 0.5–0.7, while for F value the average value, shown by black line, oscillates near 0.5. It is seen, especially on the η graph, that some of the parameter values gradually increase, while others decrease, allowing each point to adapt to its own part of the goal function landscape.

Similar graphs were obtained for other test problems, and the general trend is that the average values of parameters do not change much. For the MOEA/D-DE-SA algorithm, the parameter changes are sharper, as the previous parameter value does not influence the new one directly.

5 Conclusion

The parameter adaptation mechanism is an important part of every single-objective evolutionary and swarm intelligence search algorithm, which allows significant improvement of efficiency. Likewise, for multi-objective optimization, the parameter adaptation scheme allows receiving more representative Pareto sets, as demonstrated in this study. The proposed parameter adaptation schemes, MOEA/D-DE-SA and MOEA/D-DE-SHA use the improvement ratio, which is relatively easy to calculate for the MOEA/D framework, however, the ideas of these algorithms could be used for other multi-objective optimization approaches.

This study shows that there is a potential in improvement of MOEAs by developing new parameter adaptation schemes, as well as new improvement ration estimation approaches. Further studies in this direction may include testing the proposed SA and SHA algorithms on algorithms using SBX crossover, or other problem-specific operators, which have numerical parameters.

Acknowledgments. The is work was supported by the internal grant of Reshetnev Siberian State University of Science and Technology for the support of young researchers.

References

1. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength Pareto evolutionary algorithm. Technical report TIK-Report103, Swiss Federal Institute of Technology, Zurich, Germany (2001)
2. Deb, K., Pratab, A., Agrawal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
3. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
4. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Syst. **9**(2), 115–148 (1995)
5. Deb, K., Deb, D.: Analysing mutation schemes for real-parameter genetic algorithms. Int. J. Artif. Intell. Soft Comput. **4**(1), 1–28 (2014)
6. Das, S., Mullick, S.S., Suganthan, P.N.: Recent advances in differential evolution – an updated survey. Swarm Evol. Comput. **27**, 1–30 (2016)

7. Eiben, A.E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. IEEE Trans. Evol. Comput. **3**(2), 124–141 (1999)
8. Zhang, J., Sanderson, A.C.: Self-adaptive multiobjective differential evolution with direction information provided by archived inferior solutions. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 2806–2815, July 2008
9. Huang, V.L., Qin, A.K., Suganthan, P.N., Tasgetiren, M.F.: Multiobjective optimization based on self-adaptive differential evolution algorithm. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 3601–3608, May 2007
10. Li, K., Kwong, S., Wang, R., Cao, J., Rudas, I.J.: Multiobjective differential evolution with self-navigation. In: Proceedings of the IEEE nternational Conference on Systems, Man, and Cybernetics, pp. 508–513, October 2012
11. Li, K., Fialho, A., Kwong, S., Zhang, Q.: Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **18**(1), 114–130 (2014)
12. Brest, J., Greiner, S., Bošković, B., Mernik, M., Žumer, V.: Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans. Evol. Comput. **10**(6), 646–657 (2006)
13. Tanabe, R., Fukunaga, A.: Success-history based parameter adaptation for differential evolution. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 71–78 (2013)
14. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated Pareto sets MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. **13**(2), 284–302 (2009)
15. Huband, S., Hingston, P., Barone, L., While, L.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans. Evol. Comput. **10**(5), 477–506 (2006)
16. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017). https://doi.org/10.1109/mci.2017.2742868