A Multi-objective Artificial Flora Optimization Algorithm

Xuehan Wu, Huaizong Shao(✉), Shafei Wang, and Wenqin Wang

School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, People's Republic of China
hzshao@uestc.edu.cn

Abstract. Most of the practical problems need to consider many aspects at the same time, multi-objective optimization can be used to deal with this kind of problems. Swarm intelligence optimization algorithm can use a simple evolutionary step to find the optimal solution. Due to the advantages of swarm intelligence optimization algorithm, many researchers focus on multi-objective swarm intelligence optimization algorithms. Artificial flora (AF) optimization algorithm is a recently proposed swarm intelligence optimization algorithm. This paper proposes a multi-objective artificial flora (MOAF) optimization algorithm based on the standard artificial flora (AF) optimization algorithm. The algorithm uses the four basic elements and three main behavior patterns of the migration process and adds external document to find the Pareto optimal solution set. Simulation results show that the proposed algorithm can cover the true Pareto front with satisfactory convergence compared with the NSGA-II.

Keywords: Swarm intelligence · Artificial flora (AF) optimization algorithm · Multi-objective optimization

1 Introduction

In real life, there are many complex problems that various aspects should be considered for modeling and programming, such as urban traffic, allocation of resources, capital budget and so on [1, 2]. To find an optimal solution set for a multi-objective problem called multi-objective optimization [3]. There are two main methods to deal with multi-objective optimization problem. One is to weight multiple targets to make it into a single objective problem. This method is influenced by subjective factors. The other method is to use Pareto dominance relation to obtain a set of optimal solution, this method is the more widespread one [4].

Researchers mainly focus on heuristic algorithm in multi-objective optimization problem at present [5]. The basic theory of swarm intelligence optimization algorithm is to simulate the communication and cooperation between individuals in the actual biological group [6]. Artificial flora (AF) optimization algorithm is a recently proposed

This work was supported by National Natural Science Foundation of China under grant 61871092 and 61471103, Sichuan Science and Technology Program under grant 2017JY0262 and 2018JY0546.

H. Song et al. (Eds.): SIMUtools 2019, LNICST 295, pp. 267–277, 2019.
https://doi.org/10.1007/978-3-030-32216-8_26

intelligent optimization algorithm [7]. AF optimization algorithm using the characteristics of the plants migration to update the solutions.

This paper proposes a multi-objective artificial flora (MOAF) optimization algorithm based on standard artificial flora optimization algorithm. Algorithm using four basic elements, namely original plant, offspring plant, plant location and propagation distance, and an external document to find the optimal solution set. The external document is used to store non-dominated solution of each iteration. Grid is used in the external document to produce uniform-distributed Pareto fronts and maintain the diversity of solution set [8]. The rest of this paper is organized as follows. Section 2 introduces the multi-objective optimization problem and the related work. MOAF is introduced in Sect. 3. Section 4 uses multi-objective functions to test the efficiency of MOAF algorithm and compare it with NSGA-II. The conclusion is presented in Sect. 5.

2 Multi-objective Optimization

Most practical problems require to satisfy multiple objectives which are conflicting with each other at the same time when certain conditions are met. Suppose there are m objective functions, the dimensions of searching space is D. Multi-objective optimization problem can be expressed as finding decision variables $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_D^*]^T$ to minimize the function y:

$$\begin{aligned} \min \quad & \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x})]^T \\ s.t. \quad & g_j(\vec{x}) \geq 0 (j = 1, 2, \ldots, p) \\ & h_j(\vec{x}) = 0 (j = 1, 2, \ldots, q) \end{aligned} \tag{1}$$

Where decision variable $x_i (i = 1, 2, \ldots, D)$ satisfy $X_i^{\min} \leq x_i \leq X_i^{\max}$, $\vec{X}^{\min} = [X_1^{\min}, X_2^{\min}, \ldots, X_D^{\min}]^T$ and $\vec{X}^{\max} = [X_1^{\max}, X_2^{\max}, \ldots, X_D^{\max}]^T$ are the lower limit and the upper limit of decision variables respectively. $g_j(\vec{x}) \geq 0 (j = 1, 2, \ldots, p)$ and $h_j(\vec{x}) = 0 (j = 1, 2, \ldots, q)$ are p inequality constraints and q equality constraints. The multi-objective optimization is to find an optimal solution set rather than a unique solution, this paper uses the Pareto dominance relations to find a set of optimal solution, called Pareto optimal set [9]. If and only if component of $\vec{u}$ less than $\vec{v}$:

$$\forall i \in (1, 2, \ldots, k), u_i \leq v_i \wedge \exists i \in (1, 2, \ldots, k), u_i \prec v_i \tag{2}$$

vector $\vec{u} = (u_1, u_2, \ldots, u_k)$ dominate $\vec{v} = (v_1, v_2, \ldots, v_k)$, recorded as $\vec{u} \prec \vec{v}$. For a certain multi-objective optimization problem, Pareto optimal set (P^*) is defined as follow:

$$P^* := \{x \in \Omega | \neg \exists x' \in \Omega \ and \ \vec{f}(x') \prec \vec{f}(x)\} \tag{3}$$

Therefore, a Pareto front (PF^*) can be defined as follow:

$$PF^* := \{\vec{u} = \vec{f} = (f_1(x), \ldots, f_m(x)) | x \in P^*\} \tag{4}$$

More and more multi-objective swarm intelligence optimization algorithms have been proposed. In [10], the authors present a non-dominated sorting based multi-objective evolutionary algorithm, called NSGA-II. A dynamic sub-swarms multi-objective particle swarm optimization algorithm is proposed in [11]. In [12], a multi-objective optimization method based on the artificial bee colony, called the MOABC, is presented.

3 A Multi-objective Artificial Flora Optimization Algorithm

MOAF optimization algorithm is based on the standard artificial flora (AF) optimization algorithm. The process of finding the optimal survival position is used in this algorithm to find the optimal solution set of problems. original plant, offspring plant, plant location and propagation distance are the four basic elements in MOAF. Evolution behavior, spreading behavior and select behavior are three main behavioral patterns. Every plant location represents to a solution, and fitness of the locations is used to denote the quality of the solution. Firstly, algorithm generate the original plants randomly. Then spread seeds to the positions within the spreading scope randomly, the spreading scope is decided by the propagation distance. Next the fitness of a seed in the position is calculated according to the objective functions, the fitness represents to the solution quality. Finally, roulette is used to decide survival seeds. survival seeds become new original plants. Repeated iteration until termination condition is satisfied. Algorithm adding external documents to store the optimal solutions.

3.1 Initialization

All the decision variables of test functions worked in this paper have upper limit $\vec{X}^{\max} = [X_1^{\max}, X_2^{\max}, \ldots, X_D^{\max}]^T$ and lower limit $\vec{X}^{\min} = [X_1^{\min}, X_2^{\min}, \ldots, X_D^{\min}]^T$. At the beginning, algorithm generates N original plants according to the upper and lower limits of the decision variables randomly. Algorithm use i rows and j columns matrix $P_{i,j}$ to represent the locations of original plants, where $i = 1, 2, \ldots, D$ represents dimensionality, $j = 1, 2, \ldots, N$ represents the number of original plants:

$$P_{i,j} = rand(0, 1) \cdot (X_i^{\max} - X_i^{\min}) + X_i^{\min} \tag{5}$$

Where $rand(0, 1)$ represents the uniformly distributed number in [0, 1].

3.2 Evolution Behavior

Original plants spread their offspring within a certain scope with a radius which is propagation distance, new propagation distance imitate the propagation distance of the parent plant and grandparent plant:

$$d_j = d_{1j} \cdot rand(0,1) \cdot c_1 + d_{2j} \cdot rand(0,1) \cdot c_2 \tag{6}$$

where c_1 and c_2 denote the learning coefficient, d_{1j} and d_{2j} represent the propagation distance of grandparent plant and parent plant respectively, and rand(0, 1) is the uniformly distributed number in [0, 1]. The parent propagation distance become the new grandparent propagation distance:

$$d'_{1j} = d_{2j} \tag{7}$$

The standard AF optimization algorithm use the standard deviation between the positions of the original plants and offspring plants as new parent propagation distance:

$$d'_{2j} = \sqrt{\sum_{i=1}^{N} (P_{i,j} - P'_{i,j})^2 / N} \tag{8}$$

In order to retain the information of the optimal solutions found so far, MOAF optimization algorithm use the plants in the external document. The new parent propagation distance is the difference between the positions of plants in external document $P^*_{i,j}$ and the offspring plants $P'_{i,j}$:

$$d'_{2j} = P^*_{i,j} - P'_{i,j} \tag{9}$$

3.3 Spreading Behavior

Algorithm generates offspring plants according to the original plant locations and the new propagation distance:

$$P'_{i,j \cdot b} = G_{i,j \cdot b} + P_{i,j} \tag{10}$$

where $b = 1, 2, \ldots, B$, B represents the amount of offspring plants that one original plant can propagate, $P'_{i,j \cdot b}$ denotes the position of offspring plant, $P_{i,j}$ denotes the position of the original plant, $G_{i,j \cdot b}$ represents a random number with the Gaussian distribution with mean 0 and variance j. Generate new original plants according to Eq. (5) if there is no offspring plant survives.

3.4 Select Behavior

In standard AF algorithm, the survival probability determines whether the offspring plant survived. the survival probability is as follow:

$$p = \left| \sqrt{F(P'_{i,j \cdot b}) \Big/ F_{\max}} \right| \cdot Q_x^{(j \cdot b - 1)} \tag{11}$$

Where Q_x is selecting probability ranging between 0 and 1. $F_{\max}$ is the fitness of the offspring plant with the highest fitness. $F(P'_{i,j\cdot b})$ is the fitness of $(j \cdot b)$th solution. Calculation formulas of fitness are the functions of the objective problem. In MOAF algorithm, the Pareto dominance relations have been used. the survival probability is defined as follow:

$$p = 0.9 \cdot \frac{\text{domi}(\text{j} \cdot \text{b})}{B} + 0.1 \tag{12}$$

Where $\text{domi}(\text{j} \cdot \text{b})$ represents the amount of the solutions that dominated by solution $(\text{j} \cdot \text{b})$.$B$ represents the amount of offspring plants that one original plant can propagate.

3.5 External Document

MOAF adds an external document to store the optimal solutions in the iterations. At the beginning, the external document is empty, any non-dominated solution should be accepted by the external document. Then the new solution generated in the iterations and the solutions in external document were compared one by one. If the solution is dominated by the one in external document, the solution in external document will be deleted from the external document, and the new solution will be added into the external document. If the new solution dominates the one in external document, whether to delete the new one will be decided by roulette selection method. In the iterations, if dominated solution exists in the external document, delete the solution. When the number of solutions in external document is more than a preset document size, the grid is used to delete the external solutions. The number of grid is set artificially at the start of MOAF. Grid must cover all the solutions, if new solutions go beyond the scope of the grid, the grid should be redrawn. Algorithm always deletes the solution in the grid with highest density to guarantee the uniformity of the Pareto front.

4 A Multi-objective Artificial Flora Optimization Algorithm

Some simulation results are presented in this section to show the performance of MOAF algorithm. We use 6 benchmark functions to text the convergence of MOAF algorithm, the functions and their bounds are shown in Table 1.

We use convergence and spacing to measure performance. The convergence can use generational distance (*GD*) to express [13]. *GD* shows the distance between the elements in the set of approximate solutions found by algorithm and the elements in the Pareto optimal set. *GD* can be defined as follow:

$$GD = \sqrt{\sum_{i=1}^{n} E_i^2 / n} \tag{13}$$

Where n is the number of elements included in the found approximate solutions set. is the Euclidean distance between the *i*th element in the found approximate solutions set and the nearest element in the Pareto optimal set. The smaller the *GD*, the closer the

Table 1. Benchmark functions

Functions	Expression formula	Bounds
ZDT1	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + (9/29) \cdot \sum_{i=2}^{30} x_i$ $h(f_1(x), g(x)) = 1 - \sqrt{f_1(x)/g(x)}$	$0 \le x_i \le 1$ $1 \le i \le 30$
ZDT2	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + (9/29) \cdot \sum_{i=2}^{30} x_i$ $h(f_1(x), g(x)) = 1 - (f_1(x)/g(x))^2$	$0 \le x_i \le 1$ $1 \le i \le 30$
ZDT3	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + (9/29) \cdot \sum_{i=2}^{30} x_i$ $h(f_1(x), g(x)) = 1 - \sqrt{\frac{f_1(x)}{g(x)}} - (\frac{f_1(x)}{g(x)}) \sin(10\pi f_1(x))$	$0 \le x_i \le 1$ $1 \le i \le 30$
Deb1	$f_1(x) = x_1$ $f_2(x) = g(x) \cdot h(x)$ $g(x) = 1 + x_2^2$ $h(x) = \begin{cases} 1 - (f_1(x)/g(x))^2, & if\ f_1 \le g \\ 0, & otherwise \end{cases}$	$0 \le x_i \le 1$ $i = 1, 2$
Deb2	$f_1(x) = x_1$ $f_2(x) = g(x) \cdot h(x)$ $g(x) = 1 + 10 \cdot x_2$ $h(x) = 1 - (f_1(x)/g(x))^2 - (f_1(x)/g(x)) \cdot \sin(12\pi f_1(x))$	$0 \le x_i \le 1$ $i = 1, 2$
Deb3	$f_1(x) = 1 - e^{(-4 \cdot x_1)} \cdot \sin^4(10 \cdot \pi \cdot x_1)$ $f_2(x) = g(x) \cdot h(x)$ $g(x) = 1 + x_2^2$ $h(x) = \begin{cases} 1 - (f_1(x)/g(x))^{10}, & if\ f_1 \le g \\ 0, & otherwise \end{cases}$	$0 \le x_i \le 1$ $i = 1, 2$

found solutions get to the optimal solutions. $GD = 0$ means that all the found solutions are in the Pareto optimal set.

The distance variance of neighboring elements is used to measure the spacing (SP), it can be defined as [14]:

$$SP = \sqrt{\sum_{i=1}^{n} (\overline{d} - d_i)^2 \Big/ (n-1)} \tag{14}$$

Where $d_i = \min_j(\left|f_1^i(\overrightarrow{x}) - f_1^j(\overrightarrow{x})\right| + \left|f_2^i(\overrightarrow{x}) - f_2^j(\overrightarrow{x})\right|), i,j = 1,2,\ldots,n, \overline{d}$ is the mean of d_i, n is the number of elements included in the found approximate solutions set. $SP = 0$ means that all the elements in approximate Pareto front are equidistantly distribute.

NSGA-II is an effective algorithm and widely applied [15]. We compare the proposed algorithm with NSGA-II algorithm. The simulation results are obtained through 50 runs and both algorithms are iterated 300 times. The default parameters are shown in Table 2.

Table 2. The default parameters in NSGA-II and MOAF algorithm

Algorithm	Parameter values
NSGA-II	N = 200
MOAF	N = 200, B = 10, c1 = 1, c2 = 2

Table 3 shows the generational distance. According to the results in Table 3, MOAF can find a set of Pareto optimal solutions with better convergence compare with NSGA-II. For function ZDT1 and ZDT3, average generational distance of MOAF is 10 times less than the generational distance of NSGA-II. Average generational distance of MOAF reduces two orders of magnitude compared with NSGA-II for function ZDT2. For function Deb1-3, average generational distance of MOAF is smaller than NSGA-II. Table 4 shows the spacing results. For the spacing results, NSGA-II is better than MOAF.

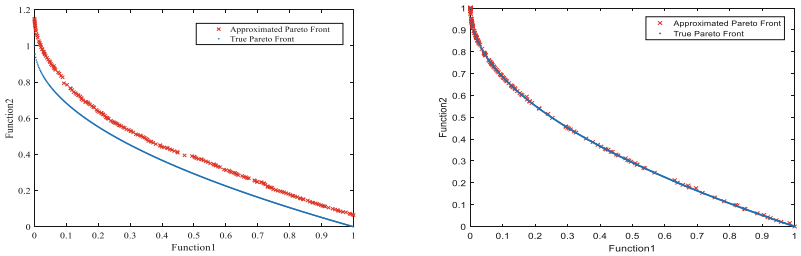
Table 3. Comparison of GD results obtained by NSGA-II and MOAF.

Functions	Algorithm	Best	Worst	Mean
ZDT1	NSGA-II	0.031441	0.090971	0.067403
	MOAF	0.001348	0.002232	0.001816
ZDT2	NSGA-II	0.045386	0.160601	0.098721
	MOAF	0.000550	0.001163	0.000817
ZDT3	NSGA-II	0.021018	0.073688	0.041069
	MOAF	0.001068	0.002386	0.001485
Deb1	NSGA-II	0.000193	0.000268	0.000218
	MOAF	0.000062	0.000173	0.000097
Deb2	NSGA-II	0.001629	0.002109	0.001887
	MOAF	0.000202	0.000574	0.000399
Deb3	NSGA-II	0.000470	0.000704	0.000594
	MOAF	0.000250	0.000525	0.000351

Table 4. Comparison of SP results obtained by NSGA-II and MOAF.

Functions	Algorithm	Best	Worst	Mean
ZDT1	NSGA-II	0.002638	0.004918	0.003348
	MOAF	0.021058	0.056519	0.033130
ZDT2	NSGA-II	0.000008	0.011005	0.003769
	MOAF	0.022346	0.045818	0.031371
ZDT3	NSGA-II	0.002333	0.008480	0.003705
	MOAF	0.011716	0.034163	0.020015
Deb1	NSGA-II	0.003066	0.004088	0.003559
	MOAF	0.224659	0.526650	0.321211
Deb2	NSGA-II	0.002988	0.004265	0.003566
	MOAF	0.031875	0.077694	0.048413
Deb3	NSGA-II	0.002715	0.003754	0.003191
	MOAF	0.367159	1.428707	0.646577

Figures 1, 2, 3, 4, 5 and 6 show the approximate Pareto front found by NSGA-II and MOAF, and the true Pareto front is shown in every figure. From Figs. 1, 2 and 3, it can be seen that NSGA-II cannot cover the true Pareto front for ZDT1-ZDT3, but MOAF can cover the true Pareto front. Although the spacing results of NSGA-II is smaller than MOAF in Table 3, it has no meaning for the approximated Pareto front cannot cover the true Pareto front. From Figs. 4, 5 and 6, the approximated Pareto front found by MOAF and NSGA-II is close to the true Pareto front for Deb1-Deb3, but it can be seen from Table 3 that the generational distance of MOAF is smaller than NSGA-II. It is easy to know that MOAF can find a set of Pareto optimal solutions with better convergence compare with NSGA-II.



(a) (b)

Fig. 1. Pareto fronts of ZDT1 produced by (a) NSGA-II (b) MOAF

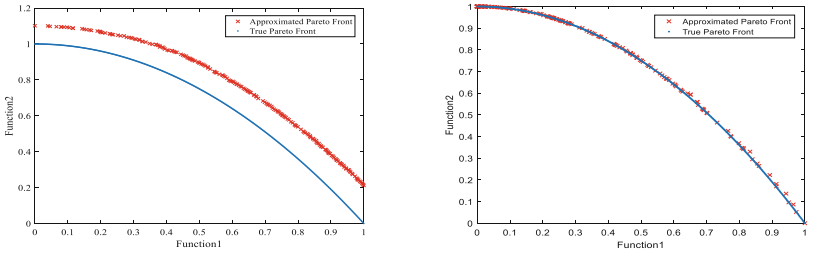


Fig. 2. Pareto fronts of ZDT2 produced by (a) NSGA-II (b) MOAF

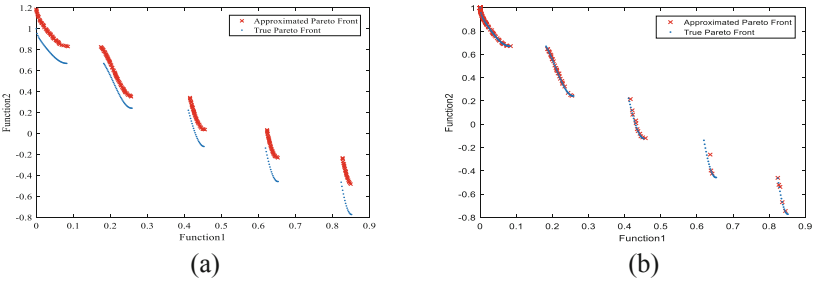


Fig. 3. Pareto fronts of ZDT3 produced by (a) NSGA-II (b) MOAF

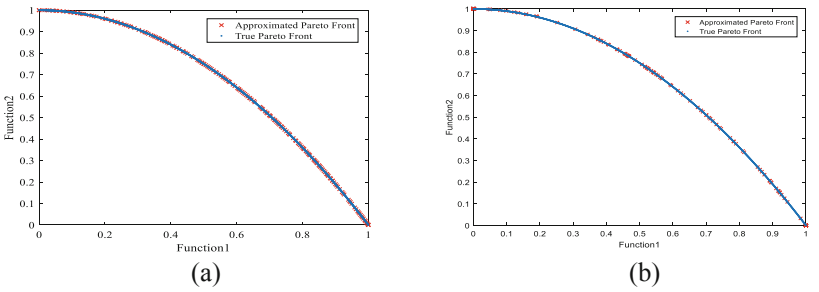


Fig. 4. Pareto fronts of Deb1 produced by (a) NSGA-II (b) MOAF

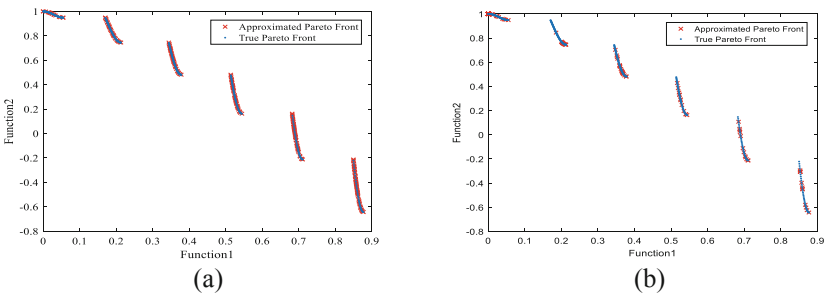


Fig. 5. Pareto fronts of Deb2 produced by (a) NSGA-II (b) MOAF

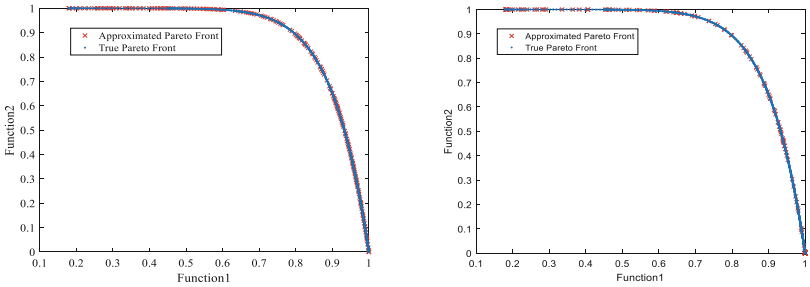


Fig. 6. Pareto fronts of Deb3 produced by (a) NSGA-II (b) MOAF

5 Conclusion

When dealing with practical problems in the real world, it is inadequate to consider only a single objective optimization, this kind of problems ask people to optimize multiple objectives equally at the same time. Multi-objective swarm intelligence optimization algorithm is an important way to solve the multi-objective problems. AF optimization algorithm is a recently proposed swarm intelligence optimization algorithm, the process of plant breeding and migration is used to find the optimal solution. Based on the standard AF optimization algorithm, this paper proposes a new multi-objective swarm intelligence optimization algorithm, called multi-objective artificial flora optimization algorithm. Algorithm uses the four basic elements and three main behavior patterns in the process of plant breeding and migration and adds external document to apply standard AF optimization algorithm to multi-objective optimization problems. The simulation results show that the approximated Pareto front found by MOAF algorithm can cover the true Pareto front and the solutions are more satisfactory than the solutions found by NSGA-II algorithm.

References

1. Sun, Y., Ng, D., Zhu, J.: Multi-objective optimization for robust power efficient and secure full-duplex wireless communication systems. IEEE Trans. Wireless Commun. **15**(8), 5511–5526 (2016)
2. Delgarm, N., Sajadi, B., Kowsary, F.: Multi-objective optimization of the building energy performance: a simulation-based approach by means of particle swarm optimization (PSO). Appl. Energy **170**, 293–303 (2016)
3. Coello, C.A.: Evolutionary multi-objective optimization: a historical view of the field. IEEE Comput. Intell. Mag. **1**(1), 28–36 (2006)
4. Asrari, A., Lotfifard, S., Payam, M.S.: Pareto dominance-based multiobjective optimization method for distribution network reconfiguration. IEEE Trans. Smart Grid **7**(3), 1401–1410 (2016)

5. Zanchetta, P.: Heuristic multi-objective optimization for cost function weights selection in finite states model predictive control. In: Predictive Control of Electrical Drives and Power Electronics, pp. 70–75. IEEE (2011)
6. Pandit, D., Zhang, L., Chottopadhyay, S.: A scattering and repulsive swarm intelligence algorithm for solving global optimization problems. Knowl.-Based Syst. **156**, 12–42 (2018)
7. Long, C., Xuehan, W., Yan, W.: Artificial Flora (AF) optimization algorithm. Appl. Sci. **8** (3), 329 (2018)
8. Coello, C.A., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. IEEE Trans. Evol. Comput. **8**(3), 256–279 (2004)
9. Ojha, M., Singh, K.P., Chakraborty, P.: An empirical study of aggregation operators with Pareto dominance in multiobjective genetic algorithm. IETE J. Res. **63**(4), 1–11 (2017)
10. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., et al. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45356-3_83
11. Zhang, Q., Xue, S.: An improved multi-objective particle swarm optimization algorithm. Eng. J. Wuhan Univ. **186**(3), 33–36 (2010)
12. Akbari, R., Hedayatzadeh, R., Ziarati, K., Hassanizadeh, B.: A multi-objective artificial bee colony algorithm. Swarm Evol. Comput. **2**(1), 39–52 (2012)
13. Zeng, G.Q., Chen, J., Li, L.M., Chen, M.R., Wu, L.: An improved multi-objective population-based extremal optimization algorithm with polynomial mutation. Inf. Sci. **330** (C), 49–73 (2016)
14. Tan, K.C., Goh, C.K., Yang, Y.J.: Evolving better population distribution and exploration in evolutionary multi-objective optimization. Eur. J. Oper. Res. **171**(2), 463–495 (2006)
15. Kannan, S., Baskar, S., Mccalley, J.D., Murugan, P.: Application of NSGA-II algorithm to generation expansion planning. IEEE Trans. Power Syst. **24**(1), 454–461 (2009)