Soft Computing (2022) 26:2325–2356
https://doi.org/10.1007/s00500-021-06522-6

OPTIMIZATION

Search in forest optimizer: a bioinspired metaheuristic algorithm for global optimization problems

Amin Ahwazian[1] · Atefeh Amindoust[1] · Reza Tavakkoli-Moghaddam[2] · Mehrdad Nikbakht[1]

Accepted: 31 October 2021 / Published online: 30 January 2022

Abstract
The present research proposes a new particle swarm optimization-based metaheuristic algorithm entitled "search in forest optimizer (SIFO)" to solve the global optimization problems. The algorithm is designed based on the organized behavior of search teams looking for missing persons in a forest. According to SIFO optimizer, a number of teams each including several experts in the search field spread out across the forest and gradually move in the same direction by finding clues from the target until they find the missing person. This search structure was designed in a mathematical structure in the form of intragroup search operators and transferring the expert member to the top team. In addition, the efficiency of the algorithm was assessed by comparing the results to the standard representations and a problem with the genetic, grey wolf, salp swarm, and ant lion optimizers. According to the results, the proposed algorithm was efficient for solving many numerical representations, compared to the other algorithms.

Keywords Search in forest · Swarm intelligence · Metaheuristic · Intragroup search · Global optimization problems · Transfer of an expert member to the top team

1 Introduction

An optimization process includes finding the best solution from all feasible solutions for a specific problem. With regard to their nature, optimization algorithms can be widely divided into two groups: deterministic algorithms and stochastic intelligent algorithms (Yang 2008). Deterministic algorithms will produce the same output when the solutions to a problem have the same initial values. These methods are classified as gradient restricted methods that move exactly towards the optimal solution. In contrast, deterministic algorithms are generally recognized as gradient-free techniques used in random steps to find the best solution possible. In this method, the optimization process cannot be repeated in any situation (Brownlee 2011). In most cases, however, favourable final solutions can be produced by both techniques. Stochastic intelligent algorithms are divided into two general modes of heuristic and metaheuristic algorithms (Yang et al. 2012).

As the name implies, heuristic algorithms use trial and error to find the solution that works and most of them are applied in various optimization areas such as bat swarm optimization, hill climbing, and simulation annealing.

Many real-world machine learning and AI problems are generally continuous, discrete, finite, or infinite (Abbassi et al. 2019). Given these characteristics, some problems cannot be easily solved with ordinary mathematical programming approaches such as conjugate gradient, sequential quadratic programming, fast steepest, and quasi-Newton methods (Faris et al. 2019).

Meanwhile, several studies have confirmed the inefficiency or low efficiency of the metaheuristic algorithms in dealing with many large-scale, indirect, and indistinguishable polynomial problems in the real world (Wu et al. 2015). Accordingly, metaheuristic algorithms are designed

✉ Atefeh Amindoust
atefeh_amnidoust@yahoo.com

Amin Ahwazian
am_ahw@yahoo.com

Reza Tavakkoli-Moghaddam
tavakoli@ut.ac.ir

Mehrdad Nikbakht
nikbakht2020@yahoo.com

[1] Department of Industrial Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Iran

[2] School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran

as a competitive alternative solvent to solve many problems owing to their simplicity and easy implementation process. Moreover, the main operations of these methods do not rely on mathematical features or gradient information. Nevertheless, one of the major drawbacks of metaheuristic algorithms is their sensitivity to setting user-defined parameters. Another problem with these algorithms is their lack of ability to always reach a global optimal solution (Dréo et al. 2006).

Metaheuristic methods are used to solve complicated issues such as scheduling (Wang and Zheng 2018), shape design (Rizk-Allah et al. 2017), economic load dispatch (Zou et al. 2016a, b), large-scale data optimization (Yi et al. 2018, 2020), infinite impulse response (IIR) systems identification (Zou et al. 2018), malware code detection (Cui et al. 2018), error detection (Li et al. 2013; Yi et al. 2018), forecasting promotion places (Nan et al. 2017), unit commitment (Srikanth et al. 2018), classification (Zou et al. 2016a, b, 2017), path planning (Wang et al. 2012a, b, 2016a, b), vehicle navigation (Chen et al. 2018), knapsack problems (Feng et al. 2018a, b), and cyber-physical systems (Cui et al. 2017), neural network (Pandey et al. 2020), trajectory tracking control of unmanned aerial vehicle (Selma et al. 2020).

Metaheuristic algorithms designed are properly implemented for most optimization issues and can successfully provide a suitable and satisfactory situation. In the area of computer sciences, researchers have proposed a new type of gradient-free technique known as optimization techniques called "genetic algorithm (GA)" by the conceptualization of evolution (Goldberg and Holland 1998). After that, several other techniques have been suggested in the field of optimization, including ant colony optimization (Dorigo and Birattari 2010), monarch butterfly optimization (MBO) (Feng et al. 2018a, b), the ant lion optimizer (Mirjalili 2018), moth search (Wang et al. 2018a, b, c), artificial bee colony (Wang et al. 2018a, b, c), evolutionary strategy (Back 1996), harmony search (Geem et al. 2001), imperialist competitive algorithm (Talataheri et al. 2012), earthworm optimization algorithm (Wang et al. 2015), cuckoo search (Wang et al. 2016a, b), biogeography-based optimization (Wang et al. 2014), elephant herding optimization (Wang et al. 2016a, b), and differential evolution (Wang et al. 2012a, b).

In general, metaheuristic algorithms are divided into two main classes (Talbi 2009), single-solution-based (e.g. simulated annealing) and population-based (e.g. GA). In the first class, only one solution is processed in the optimization stage, as the name implies. On the other hand, a set of solutions (i.e. population) evolves with each iteration of the optimization process in the second class. The population-based techniques can often find an optimal or near-optimal solution in the neighbourhood. In addition, population-based metaheuristic techniques are usually inspired by natural phenomena and start the optimization process by producing a set (population) of people, where each person in the population represents a candidate solution to the optimization problem. The population repeatedly evolves with replacing the current population with a new population using some random operators. The optimization process continues until meeting the stopping criteria (i.e. the maximum number of iterations).

2 Literature review

Particle swarm optimization (PSO) is a spirit-based stochastic optimization method proposed by Eberhart and Kennedy (1995) and by Kennedy and Eberhart (1995). Given the inefficiency of the first version of PSO in optimization issues, another version was quickly proposed by shi and eberhart (1998). Afterwards, a reproduction operator was incorporated into the algorithm following presenting the concept of a sub-population GA (SPGA) (Suganthan 1999). Another local PSO version was suggested based on the k-means clustering approach known as the social convergence method with hybrid spatial neighbourhood and ring topology by Kennedy and Stereotyping (2000). Van Den Bergh (2001) analyzed the definitive version of the PSO algorithm followed by presenting a version of the PSO algorithm with reduction factor after analyzing the convergence behavior of PSO algorithm, which guaranteed convergence and improved its rate (Clerc and Kennedy 2002). The bare-bones PSO (BBPSO) was proposed in as a dynamic model of PSO (Kennedy 2003). In 2004, Emara and Fattah (2004) analyzed the continuous version of PSO algorithm and theoretical analysis was performed on the PSO algorithm for symmetric spherical local neighbourhood functions, and numerical tests confirmed the speed characteristics along with reducing variability in the PSO algorithm (Blackwell 2005). Kadirkamanathan et al. (2006) analyzed the dynamic stability of particles using Lyapunov stability analysis and the concept of passive system (Kadirkamanathan 2006). A marine model was studied to promote search diversity in PSO (Poli et al. 2007). In addition, the social and fully aware particle version of the PSO algorithm was also introduced by Poli (2008) and the stable stochastic analysis encompassed higher-ranking moments, proving its suitability for comprehension of dynamics of particle growth and clarifying the convergence features of PSO (Poli 2009). The modified algorithm presented by Park et al. (2009) introduced the chaotic inertia weight that suddenly falls and simultaneously inclines downwards. Zhan et al. (2010) introduced orthogonal learning, in which an

orthogonal learning model was applied for efficient sampling.

Li et al. (2011) presented a self-learning PSO algorithm, in which the speed update program can change automatically during the evolution process. Approaches such as search space compression have been proposed dynamically to determine the search space (Barisal et al. 2013). Garcia-Gonzalo and Fernandez-Martinez (2014) presented a convergent reliability and stability analysis of a series of PSO types, and their research was different from the classic PSO in terms of the statistical distribution of PSO parameters. Moreover, Peng and Chen (2015) introduced a coexistence particle optimization algorithm to optimize fuzzy neural networks. Eddaly et al. (2016) presented a hybrid combinatorial particle swarm optimization algorithm (HCPSO) as a resolution technique. An iterated local search algorithm based on probabilistic perturbation is sequentially introduced to the particle swarm optimization algorithm for improving the quality of solution. The computational results showed that their approach was able to improve several best known solutions of the literature. Tanweer et al. (2016) presented a new dynamic particle optimization algorithm and self-resemblance algorithm. Moreover, Wang et al. (2018a, b) proposed a Krill herd algorithm, which is from the category of PSO algorithm. Koyuncu and Ceylan (2018) added a scout bee phase to standard PSO and formed the Scout Particle Swarm Optimization (ScPSO) to design an efficient technique for continuous function optimization; consequently, a robust optimization algorithm was obtained.

3 Search in forest theory

Every year, thousands of people lose their way in forests and mountain areas, and search-and-rescue operations are carried out to find them and bring them to their destination. To this end, some "search agents" are assigned to find the missing people in the forest after passing special education courses in this regard. These individuals are often grouped in "search teams" during operations and attempt to find the target in the depth of forests. It should be noted that in order to perform search operations, different methods can be implemented according to the opinion of the commander. However, given the standard content presented in educational courses held for police and fire brigade, who are among the main forces searching in the forest, multi-member teams are formed at first to search different parts of the forest. Afterwards, each team that finds a clue of the target informs other teams to adjust their direction. Ultimately, all teams search the direction in which most clues are found. Figure 1 graphically shows how to search the forest in accordance with the above-mentioned issues.

Two techniques are applied to organize teams to conduct effective search operations in all forest areas. The first technique is a local search inside each team, in which the commander of each team allocates parts to each member and gives them the responsibility of searching that part. The commander is informed of any clue found by team members, and they will send another team member to that area for more effective search in order to deepen the search. Therefore, all team members attempt to conduct an effective search in the parts allocated to them. In the second technique, the team that finds most clues is enforced by other teams. In fact, commanders of teams send one of their team members to the team with the most clues in order to help conduct an effective search in the forest. Meanwhile, some members remain in the teams that find the least clues in order to continue the search in the forest in case of any mistake made by the top team. In fact, a team may find several clues that are false and cannot lead to finding the target. In this case, other teams continue the search in the forest. However, their search path is in line with the search path of the top team since most clues have been found on this route. The mathematical structure inspired by this behavior will be explained in the next section to solve global optimization problems.

4 Mathematical model of search in forest optimizer

In this section, the steps related to the implementation of the mathematical model of the forest search (SIFO) optimizer are described separately.

5 Research team production

First, M teams were determined, each with N members, in order to establish initial order in the formation of search teams. In fact, $(P_n)_m$ population is created, in which m shows the number of search teams and n shows the number of team members. Since team members search the forest space in a stochastic manner and they might find a clue at any time, their movement structure is designed in two sections based on a mathematical model. In the first section, a stochastic step is designed for each m search team in the form of Model 1.

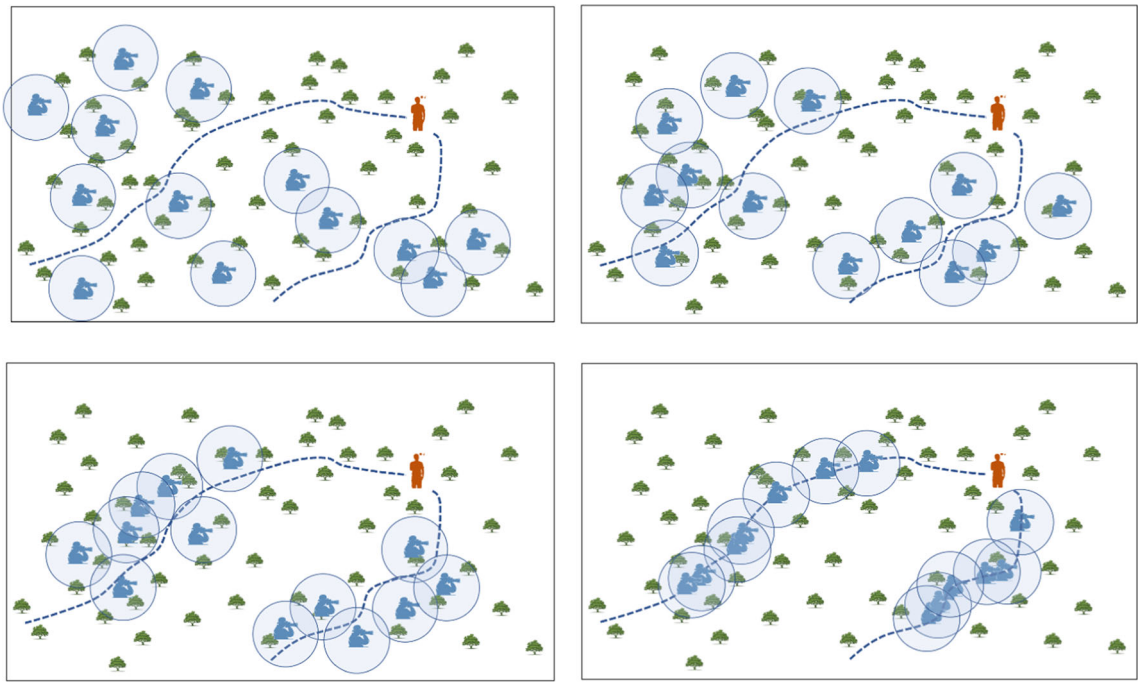


Fig. 1 Convergence of search teams to find targets in the forest

$$X_m^t = \sum_{i=1}^{Max\_Itr} \sum_{j=1}^{i} \text{Sign}_j \times \alpha_j,$$

$$s.t.$$

$$\alpha_j = \begin{cases} 0, & r1 < 0 \\ 1, & r1 \geq 0 \end{cases}, \quad \forall j \in \{1, \ldots, Max\_Itr\}$$

$$\text{sign}_j = \begin{cases} 1 & 0 \leq r2 < 0.33 \\ 0 & 0.33 \leq r2 < 0.66 \\ -1 & r2 \geq 0.66 \end{cases}, \quad \forall j \in \{1, \ldots, Max\_Itr\} \tag{1}$$

where r1 and r2 are random numbers in the [0, 1] range. Therefore, each search group (population) can search for the missing person stochastic in the forest space. Model 1 produces various stochastic search structures, which are shown in Fig. 2.

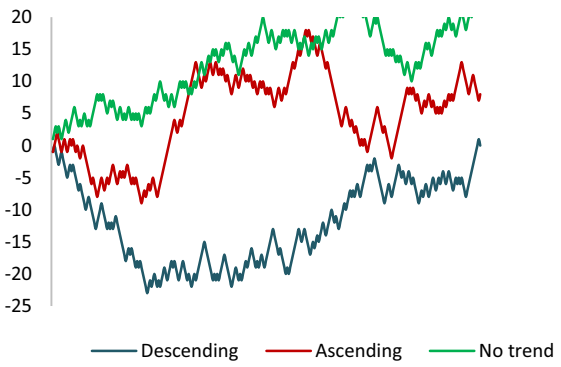


Fig. 2 Movement structure of search teams in the solution space

5.1 Search space division scheme

Given the fact that each search team searches a specific area of the solution space, a specific movement structure must be determined for each member of each team to ensure a thorough search of the entire solution space. In other words, members of each team must cover the entire solution space based on the angle assigned to them to search using stochastic movements. The search area is assumed to be enclosed in a circle in order to formulate this structure. In addition, it is assumed that all search teams are in the centre of the circle at a zero moment. As such, a proper angle of the circle is allocated to each team n based on the number of search teams (M).

In Fig. 3, the solution space is divided into eight equal parts based on the enclosed circle and space is allocated to each of the eight teams based on θ angle. Therefore, the entire solution space will be inspected by the search team members. According to Model 1, all search teams make stochastic movements in the allocated space. In addition, the surrounding space is inspected by the research team while searching the main route of their team's direction. Since the opposite view of θ angle is the sine of the angle, sine function, as shown in Model 2, can be used to mathematically define the movement structure of each search member.

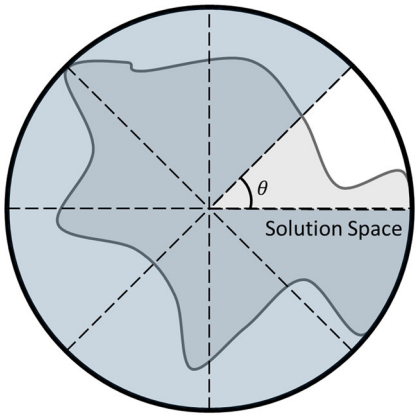


Fig. 3 Division of solution space based on the number of search teams

$$Y_{mn}^{t} = \sum_{i=1}^{Max\_Itr} \sum_{j=1}^{i} \text{Sign}_j \times \beta_j,$$

$$s.t.$$

$$\beta_j = \begin{cases} \sin\theta & 0 < \theta \le 90 \\ 1 + \sin\theta & 90 < \theta < 180 \\ 1 & \theta = \{180, 360\} \end{cases}, \quad \forall j \in \{1, \ldots, Max\_Itr\}$$

$$\theta = \frac{360}{M}, \quad \forall j \in \{1, \ldots, Max\_Itr\}$$

$$\text{sign}_j = \begin{cases} 1, & r < 0.5 \\ -1, & r \ge 0.5 \end{cases}, \quad \forall j \in \{1, \ldots, Max\_Itr\} \tag{2}$$

where r is a random number in (0, 1) range. Similar to Fig. 2, the movement structure produced in Model 2 can create various procedures, which ensures the stochastic movement of the entire members of all search teams. The position of each search team in the solution space is determined in a random matrix, in which columns are indicative of the number of search teams, while rows demonstrate the number of members in each team in the form of matrix (3).

$$\text{Position}_{\text{Team}} = \begin{bmatrix} T_{11} & \cdots & T_{1n} \\ \vdots & \ddots & \vdots \\ T_{m1} & \cdots & T_{mn} \end{bmatrix} \tag{3}$$

where T_{mn} indicates the nth member of the mth team. Clearly, each team member has a set of decision-making variables, which is shown in a matrix in the form of Eq. (4).

$$T_{mn} = [M_1, M_2, \ldots, M_v] \tag{4}$$

where M_v is the amount of the vth variable in the nth member of the mth population. The fitness of each team member can be estimated based on the problem's objective function and be considered as Eq. (5).

$$\text{Fitness}_{\text{Team}} = \begin{bmatrix} f(T_{11}) & \cdots & f(T_{1n}) \\ \vdots & \ddots & \vdots \\ f(T_{m1}) & \cdots & f(T_{mn}) \end{bmatrix} \tag{5}$$

Obviously, $f(T_{mn}) = f(T_{mn}(M_1, M_2, \ldots, M_v))$. According to the structure created, it could be expressed that team members have the same role as particles in the PSO algorithm, and the created demographic structure is based on GA.

5.2 Integration of search teams

In algorithm iterations, a number of search teams are dissolved due to failure to find proper clues of the target, and their members are added to other teams. In this regard, each team can attract the members of the dissolved team based on their performance. In fact, the allocation of the members of the dissolved team to other teams depends on the teams' fitness. However, the teams' performance must be assessed and the improper teams must be dissolved at the right time. It is best not to eliminate any team at the beginning of the search due to the need for several teams to inspect the entire solution space. As search operations expand, the search teams can be assessed and dissolved in case of poor performance. In this algorithm, it is assumed that the teams have more time to show their performance in primary iterations in order to formulate this behavior mathematically. In final iterations, however, assessments are carried out in shorter periods. This mathematical structure is shown in Model 6.

$$\begin{aligned} &\text{If}\left(\frac{\text{Best}^t - \underset{m}{\text{Worst}}\left(\text{Fitness}_m^t\right)}{\text{Best}^t}\right) \\ &\le 0.4 \rightarrow \text{remove team with the worst fitness} \\ &\text{Best}^t = \frac{\underset{m}{\text{Best}}\left(\text{Fitness}_m^t\right) + \text{Average}(\text{Fitness}^t)}{2} \\ &\text{Average}(\text{Fitness}^t) = \frac{\sum_{m=1} \text{Fitness}_m^t}{M} \end{aligned} \tag{6}$$

When the measure of the best fitness increases from the worst fitness related to the best fitness for all teams lower than the empirical value of 0.4 for a team, that team is recognized as the weaker team. It means that the search in the heuristic path would not be productive. Accordingly, before starting the next iteration, the elite members of that group will be transferred to stronger groups. Best^t indicates the fitness of the best member in the tth iteration, Worst is indicative of the fitness of the weakest population, and Average function is the mean fitness of the existing

populations in that iteration. Following dissolving the team with the weakest performance, its members are transferred to other teams. However, the number of members allocated to each of the remaining teams depends on their fitness based on Eq. 7. In order to improve the stronger teams by transferring the determined elite members to the weaker teams, we should also transfer the elite of these individuals, meaning their ability to discover the best optimal answer. It is obtained via calculating the measure of the best optimal answer of each team related to the sum of all teams' answers and multiplying the result by the number of determined individuals.

$$I_m^t = \frac{\text{Fitness}_m^t}{\sum_m \text{Fitness}_m^t} \times N_r^t \tag{7}$$

where N_r^t the number of members and r is the index of the dissolved team. The elite members of the weaker teams are determined so far. Thereby, the stronger team would have $n + I_m^t$ members. Accordingly, transferring the elite member of the weaker teams to the stronger teams was performed until the final iteration of the algorithm. Therefore, the teams with a weak performance can be eliminated in various algorithm iterations and the remaining teams' ability to search can be strengthened through increasing their members.

5.3 Intragroup search

In this section, we describe the local search structure conducted inside each group to improve each of its members and ultimately enhance the group's ability to search. As explained, an intragroup search occurs when one of the members of the group finds a clue. For a more detailed search, the commander of the group sends other members to the direction of the mentioned member to reinforce the local search to a certain extent. To this end, $S_m(t)$ is defined as a set of neighbours on the tth variable in $X_m(t)$ solution.

$$S_m(t) = \left\{W_m^+(t), W_m^+(t)\right\} \tag{8}$$

$$W_m^+(t) = X_m^t + c \times (u_m(t) - v_m(t)) \tag{9}$$

$$W_m^-(t) = X_m^t + c \times (u_m(t) - v_m(t)) \tag{10}$$

where $W_m^+(t)$ and $W_m^-(t)$ indicate two neighbours of X_m^t solution, and $u_m(t) = (u_1(t), u_2(t), \ldots, u_M(t)), \quad m \in \{1, \ldots, M\}$ and $v_m(t) = (v_1(t), v_2(t), \ldots, v_M(t)), \quad m \in \{1, \ldots, M\}$ refer to two random solutions selected from P_m population. Moreover, c shows the cluttered factor obtained based on the normal distribution of $N(\mu, \sigma^2)$. While the application of the cluttered factor in the local search algorithm is not considered as a decision-making criterion, it has a slight impact on the improvement process of algorithm implementation. Meanwhile, careful adjustments must be made for u and v factors. However, it is easily justified that u factor must be greater than zero ($u > 0$). The value of the factor must be determined accurately since very large amounts lead to extremely large values for $(u_m(t) - v_m(t))$, which intensifies the cluttered br in the algorithm. On the other hand, the value of the mentioned equation will be very small if very low amounts of the factor are used. This leads to very small values obtained from multiplying the c factor into the factor's value, very low cluttered factor, and low algorithm convergence. As such, the parameter should be set with great care. In addition, σ parameter's value should be greater than zero. Notably, very large or very low σ values affect the c parameter, and consequently, the convergence of the algorithm. It is suggested that the values of σ and μ be in the (0, 1) range.

The stochastic step created in models 1 and 2 cannot be used directly since the solutions produced by algorithm iterations for each member of the population should be in the permissible range of problem variables. Therefore, necessary numerical conversions should be made using standardization functions.

$$X_m^t = Z_{X_m^t} \tag{11}$$

$$Y_m^t = Z_{Y_m^t} \tag{12}$$

5.4 Transfer of elite members to the top team

The elite members are the individuals of each team providing the best optimal answer in each search. The superior teams are the ones providing more optimal answers in each search compared to the other teams. Indeed, the teams with more fitness are in the right heuristic path of the total optimal answer compared to the other teams. Transferring the elite individuals to the superior teams leads to their strength in accurately searching the neighbourhood through the heuristic path to find the best fitness, and it is one of the tools for exiting the local optimizations, preventing the algorithm from getting fall in them.

In this research, the final solution was improved using the random walk method (Levy flight) based on the equation below:

$$X_m^{t+1} = X_m^t + \text{Levy}(d) \times X_m^t \tag{13}$$

where t is the number of algorithm iteration and d is the dimension of the team members' position vector. The Levy flight value is estimated using the equation below:

$$\text{Levy}\left(X_m^t\right) = 0.01 \times \frac{r_1 \times \sigma}{|r_2|^{\frac{1}{\beta}}} \tag{14}$$

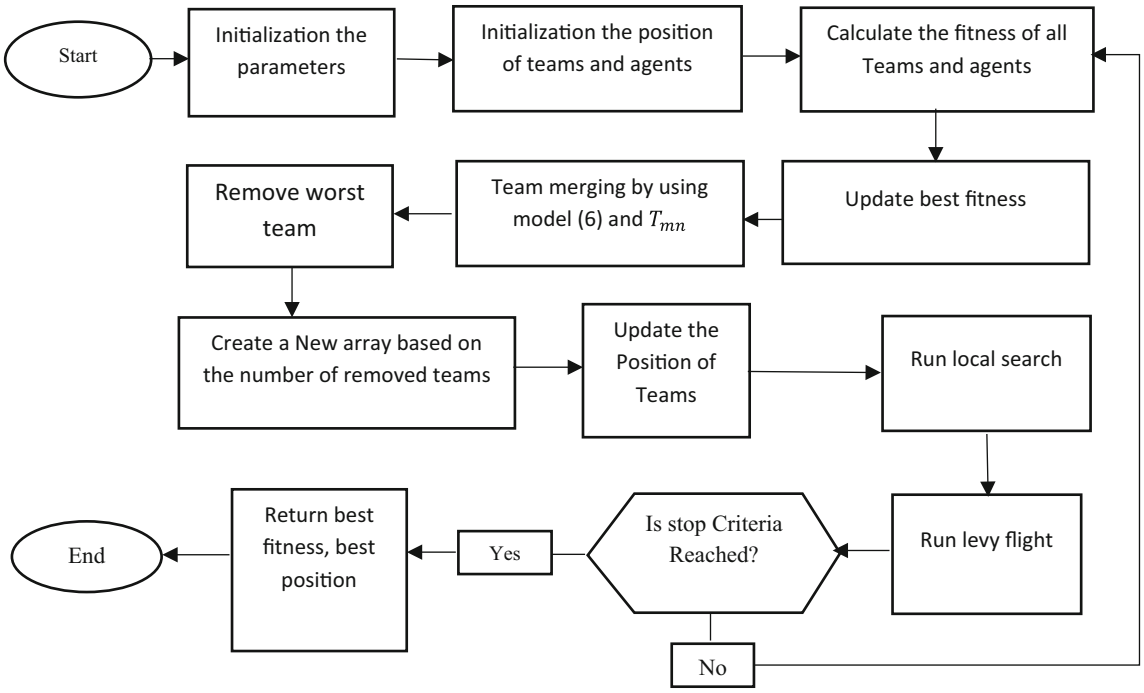


Fig. 4 Flowchart of the SIFO algorithm

where r_1 and r_2 are random values in the (0, 1) range. In addition, β is a constant number (e.g. 1.5). Finally, σ is calculated based on the equation below:

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right) \tag{15}$$

where $\Gamma\left(X_m^t\right) = \left(X_m^t - 1\right)!$ Therefore, the solutions generated are improved with each algorithm iteration, and the best algorithm solution is considered as the final solution.

5.5 Flowchart and the pseudo-code of the SIFO algorithm

Figure 4 shows flowchart of the SIFO algorithm.

Algorithm 1 Pseudo-code of SIFO

Initialize the parameters *M, N, Pr, c, Max_iteration,* ;
Initialize the positions of Teams and agents using T_{mn};
While ($t \leq Max\_iteraition$)
  Calculate the fitness of all *Teams and agents*;
  $Update\ bestFitness, T_{mn}$
  Team merging using **Model** (6) and *Pr*;
  For *each search agent*
    Run Local search;
    Run Levy flight;
  End For
  $t = t + 1$;
End While
Return *bestFitness, best position*;

6 Results and discussion

In this section, we evaluate the performance of the proposed algorithm in solving two general categories of optimization problems, including constrained and unconstrained problems. Figure 5 shows diagram block was provided to demonstrate the road map in different steps of this part. All problems have been run in a 3.2 GHz system with 32 GB random access memory in Windows 10.

6.1 Introducing standard unconstrained optimization and parameter regulation tests

In this research, we use four popular standardized tests, including classic unimodal (Yao et al. 1999; Digalakis and Margaritis 2001) and multimodal functions (Molga and Smutnicki 2005), unimodal and multimodal CEC2014 functions and CEC2014 combined functions. The mathematical structure, dimensions, and the range of values of these functions are described in Tables 1, 2, 3, 4.

A very important issue in the implementation of a metaheuristic algorithm is the accurate adjustment of parameters to increase their efficiency. It is worth noting that there is still no comprehensive approach to the assessment of the performance of metaheuristic algorithms (Yang 2010). Therefore, the solutions provided by the proposed algorithm should be compared to the results obtained from other algorithms. Therefore, we considered the GA (Davis 1991), grey wolf optimizer (GWO)

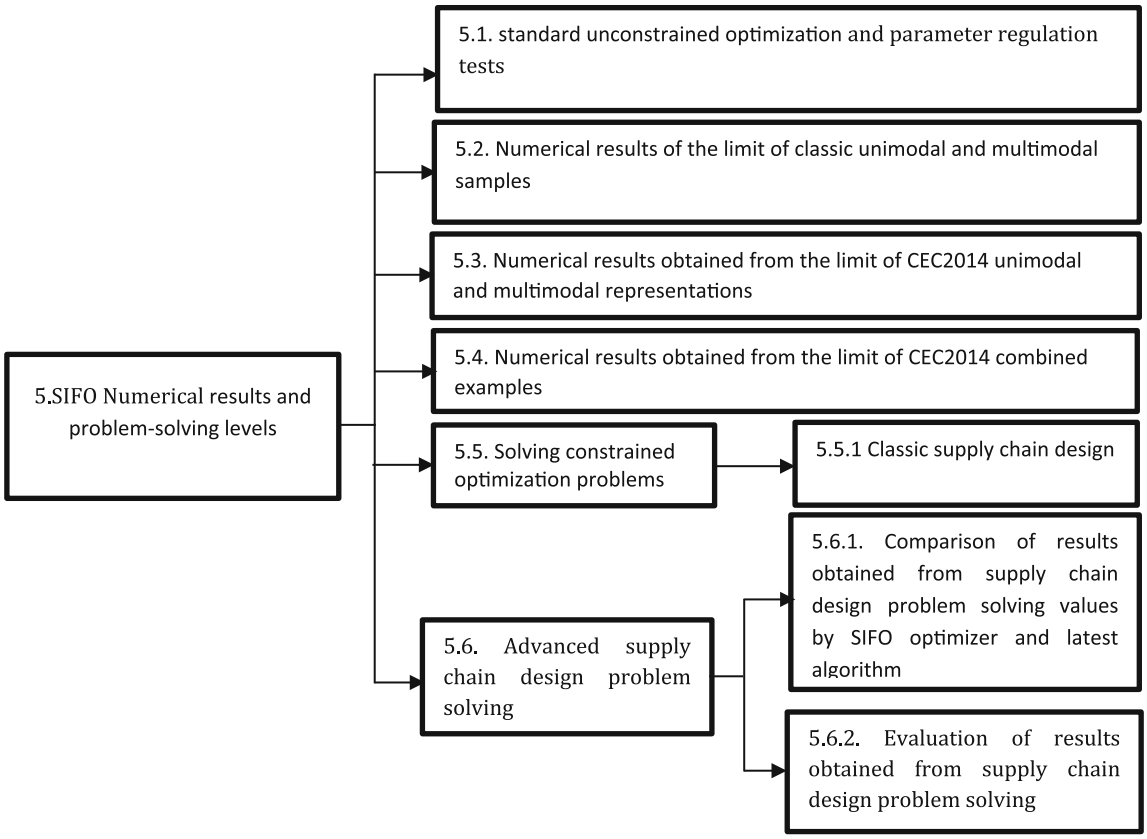


Fig. 5 Road map of result and discussion

(Mirjalili et al.2014), salp swarm (SSA) (Mirjalili et al. 2017), ant lion optimizer (ALO) (Mirjalili 2015), gravitational search algorithm (GSA) (Rashidi and Cook 2009), differential evolution (DE) (Storn and Price 1997), and PSO (Zhou et al. 2003) algorithms as the base algorithms to compare the results. It is thereby necessary to adjust the parameters of each of the mentioned algorithms based on Table 4.

All algorithms are implemented to achieve final results in similar situations. In all algorithms, the population size is estimated at 50 and a total of 1000 iterations are considered. In order to control the performance of all algorithms, the best results are considered among the 30 independent performances. In addition, a number of diagrams are presented below to evaluate the results of the proposed algorithm in comparison with other metaheuristic algorithms in terms of solving standard numerical representations.

- *Diagram of search structure* The diagram shows the location of each search agent in the forest space during the implementation of the algorithm. The diagram helps determine the level of success of search agents in the solution space.
- *Improved trajectory* The diagram shows the amount of target function in the best solution obtained in each implementation of the algorithm. In fact, the diagram shows the improved trajectory of the best member of the search agent population.
- *Mean solutions* The diagram demonstrates the mean of a fitness function for all members in each iteration. The diagram indicates the algorithm's ability in converging all population members.
- *Convergence diagram* The diagram shows the convergence of the algorithm towards the best solution in each iteration.

6.2 Numerical results of the limit of classic unimodal and multimodal samples

The diagrams related to the comparison criteria are presented in Fig. 6 following solving the classic unimodal and multimodal numerical representations with the help of the proposed algorithm.

According to the results, the population members are scattered in the space during the optimization process and are concentrated on the optimal solution. This shows that the algorithm performance has balance in diversification

Table 1 Standard classic unimodal and multimodal numerical representations

	Function	Dim	Range	f_{min}
Unimodal benchmark functions	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_i \right)^2$	30	$[-100, 100]$	0
	$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	$[-100, 100]$	0
	$f_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	$[-30, 30]$	0
	$f_6(x) = \sum_{i=1}^{n} [x_i + 0.5]^2$	30	$[-100, 100]$	0
	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0
Multimodal benchmark functions	$f_8(x) = \sum_{i=1}^{n} -x_i sin\left(\sqrt{\lvert x_i^2 \rvert} \right)$	30	$[-500, 500]$	-418.9829×5
	$f_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	30	$[-5.12, 5.12]$	0
	$f_{10}(x) = -20\exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \cos(2\pi x_i) \right) + 20 + e$	30	$[-32, 32]$	0
	$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	$[-600, 600]$	0
	$f_{12}(x) = \frac{\pi}{2} \left\{ 10\sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \left[10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	30	$[-50, 50]$	0
	$y_1 = 1 + \frac{x_i + 1}{4}$			
	$u(x_i, 10, 100, 4) = \begin{cases} lk(x_i - a)^2 & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^2 & x_i < -a \end{cases}$			
	$f_{13}(x) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 \left[\sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 \left[\sin^2(2\pi x_n) \right] \right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

Table 2 CEC2014 unimodal and multimodal standard numerical representations

Function	Dim	Range	f_{min}
$f_{14}(x)$ = Rotated High Conditioned Elliptic Function	30	$[-100, 100]$	100
$f_{15}(x)$ = Rotated Cigar Function	30	$[-100, 100]$	200
$f_{16}(x)$ = Shifted and Rotated Ackley's Function	30	$[-100, 100]$	500
$f_{17}(x)$ = Rotated Weierstrass Function	30	$[-100, 100]$	600
$f_{18}(x)$ = Rotated HappyCat Function	30	$[-100, 100]$	1300
$f_{19}(x)$ = Rotated HGBat Function	30	$[-100, 100]$	1400
$f_{20}(x)$ = Rotated Expanded Griewank's plus Rosenbrock's Function	30	$[-100, 100]$	1500
$f_{21}(x)$ = Rotated Expanded Scaffer's F6 Function	30	$[-100, 100]$	1600

Table 3 CEC2014 combined standard numerical representations

Function	Dim	Range	f_{min}
$f_{22}(x)$ = Composite function 1	30	$[-100, 100]$	2300
$f_{23}(x)$ = Composite function 2	30	$[-100, 100]$	2400
$f_{24}(x)$ = Composite function 3	30	$[-100, 100]$	2500
$f_{25}(x)$ = Composite function 4	30	$[-100, 100]$	2600
$f_{26}(x)$ = Composite function 5	30	$[-100, 100]$	2700
$f_{27}(x)$ = Composite function 6	30	$[-100, 100]$	2800

Table 4 Parameters of algorithms

Algorithms	Parameters
SIFO	$M = 10, N = 25, c = 0.3$
GA	crossover $= 0.7$, mutation $= 0.3$
GWO	$a = [0, 2]$
SSA	$c_1 = [0, 1], c_2 = [0, 1]$
ALO	$k = 500$
PSO	$c_1 = 2, c_2 = 2, v\text{Max} = 6$
DE	crossover $= 0.5$
GSA	$g = 0.2$

and contemplation phases. In addition, the trajectory of the best member of the population in each iteration shows many numerical changes in the first iterations and then convergence on the final solution. In fact, this confirms a balance in the discovery of proper solutions in the initial iterations (diversification) and focusing on the best solution for local search to make an optimized global finding. Evaluation of the convergent diagram shows that the convergence process is carried out with less slope or is discontinued completely in some iterations. This means that the algorithm has failed to find a solution better than the existing solution. However, a sudden significant improvement is observed and the improvement slope has a suitable state again. This shows that the proposed algorithm can properly pass the trap of local optimization solutions and continue the search in other parts of the space using the operators designed in order to pass local optimization and move towards global optimization. Therefore, the proposed algorithm is able to discover global optimization solutions in all standard representations in a shorter period. The results obtained from solving the classic unimodal and multimodal numerical representations are presented in Table 5 with the help of the base algorithms and proposed algorithm.

As observed, the SIFO optimizer produced better solutions in all numerical representations, compared to SSA, ALO, PSO, and GSA algorithms. Moreover, SIFO shows very competitive results compared to DE in F4, F5, F6, F11, F12, and F13 representations and compared to GWA in F8 and F11 representations. Notably, the mean and standard deviation of solutions are considered in Table 5. Meanwhile, SIFO optimizer produced better solutions in all numerical representations, compared to the other algorithms in independent implementations. In other words, SIFO was able to provide solutions with higher quality in independent implementations. The best and worst values for final solutions obtained from the independent implementation of different algorithms are presented in Table 6 to better show this superiority. In addition, the number of times that the SIFO optimizer has been able to generate better solutions, compared to other algorithms, is shown under the heading NBS.

As observed, SIFO optimizer produced better independent solutions in standard representations, compared to several algorithms, which shows its capacity to be used as an efficient algorithm in solving optimization problems.

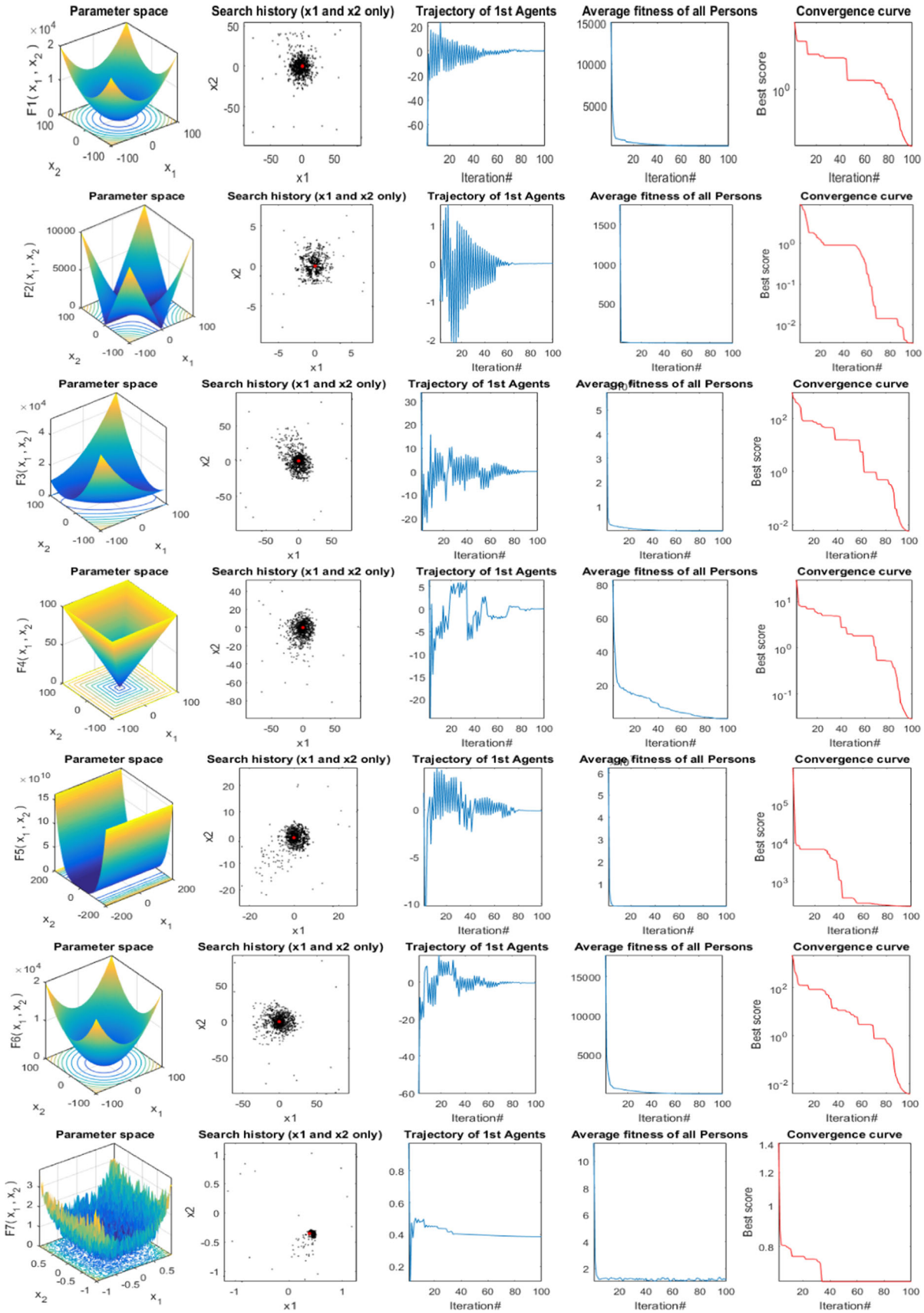


Fig. 6 Behavior of the proposed algorithm in solving the standard numerical representation

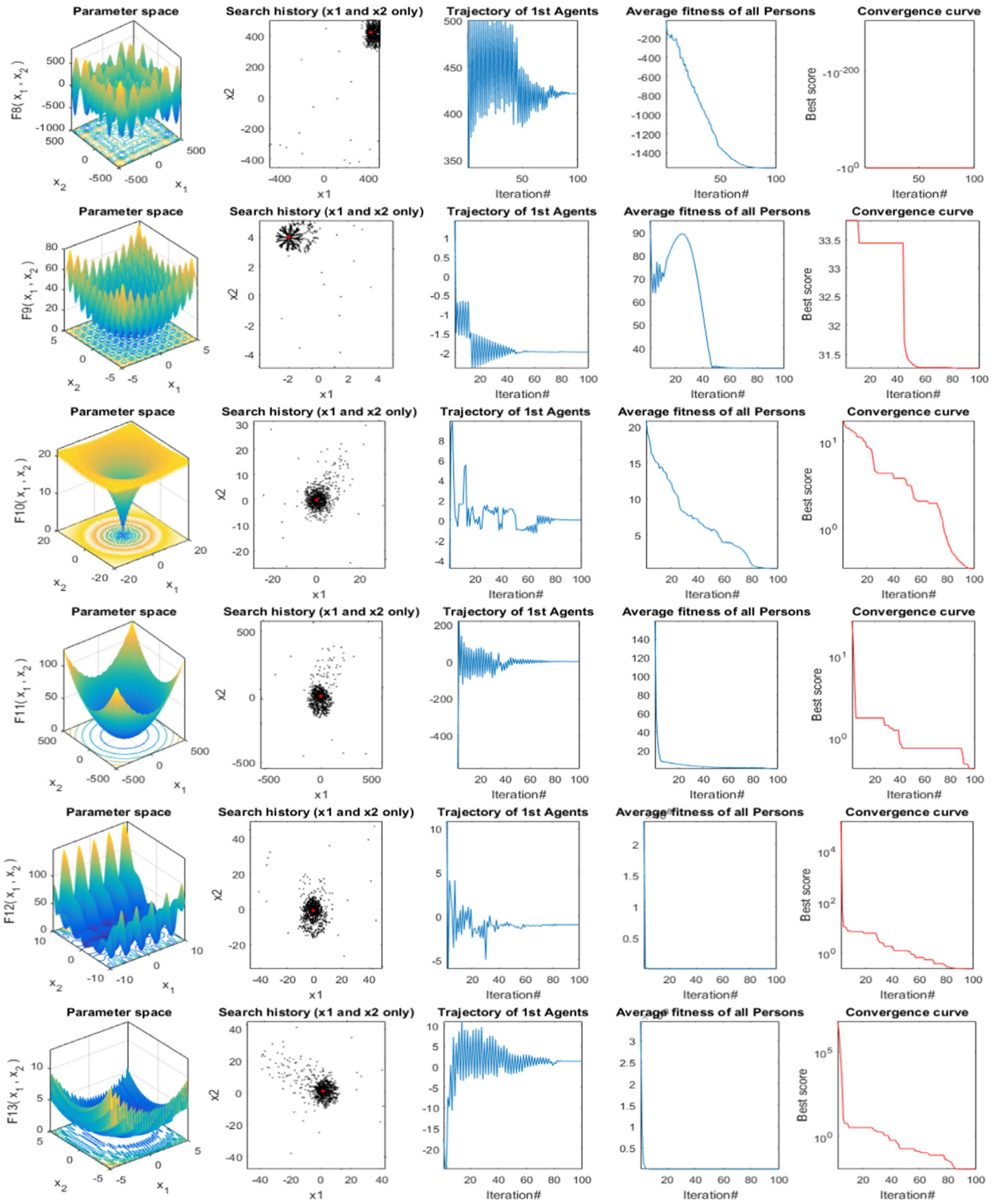


Fig. 6 continued

Table 5 Numerical results obtained from solving the base algorithms and SIFO optimizer

Standard representation	SIFO		GA		GWO		SSA	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std
F1	**2.41E−107**	7.91E−05	0.00057	0.00013	6.59E−28	**6.34E−05**	1.8531E−08	8.85E−05
F2	**1.922E−52**	5.39E−02	0.0081	0.00077	7.18E−17	0.029014	9.2792E−06	8.95E−02
F3	**5.11E−29**	8.40E+01	0.016	0.014	3.29E−06	79.14958	5.8652E−10	1.40E+02
F4	2.3189E−25	7.75E−01	0.3	0.5	5.61E−07	1.315088	1.3316E−05	1.12E+00
F5	0.40944	3.96E+01	5.06	5.87	26.81258	69.90499	7.3879	6.41E+01
F6	1.9265E−05	4.18E−05	**0**	0	0.816579	0.000126	5.1691E−10	6.96E−05
F7	**0.00077283**	1.08E−01	0.1415	0.3522	0.002213	0.100286	0.011272	6.29E−02
F8	− 1855.0007	− 3.63E+02	− 154.5	52.6	**− 6123.1**	**− 4087.44**	− 2923.4471	− 8.14E+02
F9	**0**	2.11E+01	0.046	0.012	0.310521	47.35612	12.9344	2.22E+01
F10	**8.8818E−16**	1.65E−01	1.80E−02	2.10E−03	1.06E−13	7.78E−02	1.6462	2.74E−01
F11	0.012453	1.02E+00	1.60E−02	2.20E−02	**4.49E−03**	**6.66E−03**	0.1108	1.68E+00
F12	2.033E−07	2.00E−01	9.2E−01	3.6E−01	5.34E−02	2.07E−02	1.0168E−11	3.33E−01
F13	2.0979E−05	1.43E+00	1.60E−04	7.30E−05	6.54E−01	4.47E−03	0.010987	2.38E+00

Standard representation	ALO		PSO		DE		GSA	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std
F1	2.59E−10	1.65E−10	0.000136	0.000202	8.20E−14	5.90E−14	2.53E−16	9.67E−17
F2	1.84E−06	6.58E−07	0.042144	0.045421	1.50E−09	9.90E−10	0.055655	0.194074
F3	6.06847E−10	6.34E−10	70.12562	22.11924	6.80E−11	7.40E−11	896.5347	318.9559
F4	1.36E−08	1.81E−09	1.086481	0.317039	**0**	0	7.35487	1.741452
F5	0.34677239	0.109584	96.71832	60.11559	**0**	0	67.54309	62.22534
F6	2.56E−10	1.09E−10	0.000102	8.28E−05	**0**	0	2.50E−16	1.74E−16
F7	0.00429249	0.005089	0.122854	0.044957	0.00463	0.0012	0.089441	0.04339
F8	− 1606.2764	314.4302	− 4841.29	1152.814	− 1108.1	574.7	− 2821.07	493.0375
F9	7.71E−06	8.45E−06	46.70423	11.62938	69.2	38.8	25.96841	7.470068
F10	3.73E−15	1.50E−15	2.76E−01	5.09E−01	9.7E08	4.2E−08	6.21E−02	2.36E−01
F11	0.01860449	0.009545	9.22E−03	7.72E−03	0.00E+00	0.00E+00	2.77E+01	5.04E+00
F12	9.75E−12	9.33E−12	6.92E−03	2.63E−02	**7.9E−15**	8.00E−15	1.80E+00	9.51E−01
F13	2.00E−11	1.13E−11	6.68E−03	8.91E−03	**5.1E−14**	4.8E−14	8.90E+00	7.13E+00

Bold numbers represent the best values

6.3 Numerical results obtained from the limit of CEC2014 unimodal and multimodal representations

In this section, CEC2014 standard representations presented in two unimodal and multimodal versions are solved using SIFO optimizer and other comparative algorithms, and the results are presented in the table. It should be pointed out that owing to higher computational complexity, compared to F1–F13 functions, the samples can challenge the solution discovery power of the proposed algorithm in the exploration phase. This is mainly due to the fact that discovering the accurate location of the global optimization solution in the complicated computational space of these algorithms is extremely difficult, and the algorithm may get stuck in the local optimization trap. In fact, the operators presented in different algorithms can be well evaluated in this category of standard functions. The necessary comparative results are presented in Tables 7 and 8. Notably, solving the two standard samples was assessed in two modes of 1000 and 10,000 iterations to evaluate the effect of an increased number of iterations on the final solutions.

As observed, the proposed algorithm was able to provide more efficient solutions in most numerical examples, compared to the other algorithms. This level of superiority is due to the presence of two different populations, which extremely increases search power in the exploration phase.

As observed, the final solutions significantly improved with an increase in the number of iterations, which confirmed the proper implementation of the algorithm's operators to escape the local optimization trap. In fact, an increase in the number of iterations enabled the algorithms

Table 6 Comparison of the best, worst, and number of times of superiority of SIFO optimizer over other algorithms

Standard representation	SIFO		GA			GWO			SSA		
	Best	Worst	Best	Worst	NBS	Best	Worst	NBS	Best	Worst	NBS
F1	**2.31E−117**	**2.87E−94**	0/0005586	0/0006099	10	6.3923E−28	7.1172E−28	10	1/81604E−08	2/00135E−08	10
F2	**1.844E−59**	**2.019E−51**	0/007938	0/008667	10	6.821E−17	7.7544E−17	10	9/09362E−06	9/74316E−06	10
F3	**5.1728E−31**	**5.3324E−26**	0/0152	0/0168	10	3.1584E−06	3.5532E−06	10	5/57194E−10	6/27576E−10	10
F4	2.3189E−25	2.3189E−21	0/285	0/318	10	5.3856E−07	6.0027E−07	10	1/27834E−05	1/4115E−05	10
F5	0.40944	0.57441	4/807	5/313	10	26.2763284	28.153209	10	7/166263	7/905053	10
F6	1.9265E−05	2.0174E−05	**0**	**0**	0	0.79208163	0.88190532	10	5/06572E−10	5/47925E−10	3
F7	**0.00055745**	**0.00087429**	0/13867	0/14999	10	0.00210235	0.00234578	6	0/01082112	0/01194832	7
F8	− 3845.0009	− 1387.0001	− 151/41	− 162/225	10	**− 6000.638**	**− 6429.255**	3	− 2806/50922	− 3069/61946	1
F9	**0**	**0**	0/04462	0/04968	10	0.30120537	0.33225747	10	12/417024	13/710464	10
F10	**8.0017E−21**	**8.9857E−15**	0/0171	0/01944	10	1.0282E−13	1.113E−13	7	1/613276	1/744972	10
F11	0.008657	0.014687	0/01552	0/01712	10	0.0043553	0.0048492	4	0/10526	0/119664	7
F12	2.027E−09	2.114E−07	0/8832	0/9752	10	0.051798	0.057672	5	9/76128E−12	1/08798E−11	6
F13	1.8729E−05	2.1187E−05	0/0001568	0/0001728	4	0.62784	0.6867	7	0/01076726	0/01164622	10

Standard representation	ALO			PSO			DE			GSA		
	Best	Worst	NBS	Best	Worst	NBS	Best	Worst	NBS	Best	Worst	NBS
F1	2.4605E−10	2.7195E−10	10	0.00013192	0.00014688	10	8.036E−14	8.61E−14	10	2.4541E−16	2.7324E−16	10
F2	1.7848E−06	1.9504E−06	10	0.04087968	0.04551552	10	1.47E−09	1.59E−09	10	0.05287225	0.0589943	10
F3	5.76505E−10	6.55395E−10	10	67.3205952	74.3331572	10	6.596E−11	7.14E−11	10	878.604006	968.257476	10
F4	1.3192E−08	1.4552E−08	10	1.03215695	1.16253467	10	**0**	**0**	0	7.1342239	7.7226135	10
F5	0.332901494	0.36411101	0	91.882404	102.5214192	10	**0**	**0**	0	64.1659355	70.9202445	10
F6	2.4576E−10	2.7648E−10	3	0.00009792	0.00011016	10	**0**	**0**	0	2.45E−16	2.625E−16	1
F7	0.004077866	0.004550039	10	0.11793984	0.13145378	10	0.0045374	0.0049541	10	0.08586336	0.09570187	10
F8	− 1574.15087	− 1686.59022	4	**− 4599.2255**	**− 5180.1803**	1	− 1085.938	− 1163.505	2	− 2708.2272	− 2962.1235	2
F9	7.4016E−06	8.2497E−06	10	45.3031031	50.4405684	10	67.124	73.352	10	24.6699895	28.0458828	10
F10	3.5435E−15	3.9538E−15	10	0.26496	0.29808	10	9.215E−08	1.0282E−07	10	0.060237	0.065826	10
F11	0.017674266	0.020092849	3	0.0089434	0.009681	2	**0**	**0**	4	27.146	29.362	6
F12	9.4575E−12	1.02375E−11	3	0.0066432	0.007266	10	**7.505E−15**	**8.374E−15**	0	1.746	1.89	10
F13	1.96E−11	2.1E−11	0	0.0065464	0.0072144	7	**4.896E−14**	**5.406E−14**	0	8.633	9.612	10

Bold numbers represent the best values

Table 7 Results of solving CEC2014 unimodal and multimodal standard representations with 1000 iterations

	F14			F15			F16			F17		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	**8,522,985**	**6,205,988.4**	**28,681,290**	**21,326.684**	**13,167.541**	**55,011,790**	565.47887	0.0989728	552.8698	657.77668	3.1269863	667.6992
GA	405,411,981	101,193,246	716,307,600	2.728E+10	5.011E+09	3.543E+10	583.82725	0.0524357	549.4335	626.97282	**2.1282005**	661.32
GWO	18,348,593	7,692,844.2	59,495,100	10,030.949	10,887.71	325,542,000	564.9344	0.1030831	565.845	704.62436	4.5871848	716.9688
SSA	83,737,387	65,199,640	124,227,000	2.074E+09	1.624E+09	3.902E+09	565.88375	0.0534223	535.48	**597.117**	2.3918105	720.024
ALO	83,634,732	124,176,671	95,200,000	1.141E+10	7.71E+09	1.218E+10	552.67478	0.1722753	**520.672**	667.93883	2.5326636	**655.504**
PSO	153,051,592	64,485,240	156,816,000	2.038E+09	932,439,600	2.058E+09	547.44849	0.1107506	602.4672	673.36846	2.7495863	701.5032
DE	32,796,656	13,569,179	118,886,400	16,515,654	10,524,962	2.274E+09	**539.58343**	**0.0765576**	565.1316	687.5771	3.9033215	720.048
GSA	281,972,016	110,300,212	315,675,000	5.746E+09	2.473E+09	6.01E+09	570.35396	0.0861795	560.3745	700.14832	2.6391546	732.4695

	F18			F19			F20			F21		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	**1434.7418**	**0.1052008**	1545.12	1507.131	0.3452971	1729.6	**1644.8209**	**2.9013992**	1672.275	1788.9315	0.5615324	1828.3769
GA	1437.5537	0.3482953	**1474**	1769.3398	13.533709	1753.548	1729.7023	12,444.043	1715.6475	1815.0498	**0.2234301**	1945.5072
GWO	1661.564	0.1481248	1617.27	1624.4851	**0.2176479**	1639.36	1667.5496	3.9556884	1856.353	1835.1684	0.4995384	1780.1847
SSA	1632.5283	0.4959177	1661.1	1627.0606	7.5472792	**1609.722**	1721.1485	893.07076	1836.0555	1845.9576	0.6010072	1816.6288
ALO	1449.142	1.0251744	1588.4	1726.4993	20.236166	1761.424	1717.8582	371,047.57	1713.8128	**1763.7048**	0.5270802	1851.3824
PSO	1530.3865	0.2504283	1475.52	1566.4125	5.7621958	1756.512	1677.599	116.06209	1720.8431	1794.4435	0.4085195	1822.744
DE	1531.1723	0.1393287	1631.34	**1506.6699**	0.278098	1638.76	1769.6817	6.0667665	**1655.8626**	1880.1392	0.4690218	**1771.385**
GSA	1633.192	0.9657844	1601.7	1612.0919	9.4447633	1814.7	1754.6144	10,558.19	1857.411	1892.0438	0.2593371	1839.1146

Bold numbers represent the best values

Table 8 Results of solving CEC2014 unimodal and multimodal standard representations with 10,000 iterations

	F14			F15			F16			F17		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	8,117,128.6	**5,746,285.6**	**25,839,000**	19,565.766	12,306.113	**48,683,000**	**503.484928**	0.0916415	507.22	**607.45398**	2.7429704	618.24
GA	378,889,702	94,573,127	628,340,000	2.393E+10	4.396E+09	3.375E+10	507.675868	**0.0459962**	523.27	585.95591	**1.9524775**	**601.2**
GWO	**1,627,693**	6,930,490.3	56,662,000	**9202.7059**	**9721.1697**	283,080,000	491.247304	0.0896375	538.9	618.09154	4.2084264	628.92
SSA	75,439,088	56,695,339	116,100,000	1.938E+09	1.425E+09	3.423E+09	538.9369	0.0464542	**486.8**	563.31792	2.1355451	631.6
ALO	77,439,567	109,890,859	85,000,000	1.086E+10	6.763E+09	1.107E+10	507.04108	0.1538172	491.2	624.24189	2.3235446	618.4
PSO	144,388,294	60,266,579	142,560,000	1.788E+09	879,660,000	1.888E+09	502.246321	0.1054768	528.48	629.31632	2.5697068	649.54
DE	30,088,675	12,448,788	110,080,000	14,487,416	9,929,209.5	2.106E+09	486.1112	0.0702364	523.27	613.90813	3.717449	642.9
GSA	268,544,777	101,192,855	274,500,000	5.472E+09	2.311E+09	5.415E+09	523.260517	0.0797958	533.69	630.76425	2.3992314	636.93

	F18			F19			F20			F21		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	**1353.53**	**0.1001912**	**1392**	**1408.5336**	0.3002583	**1504**	**1509.01**	**2.590535**	1520.25	**1611.65**	0.4882891	1677.41
GA	1356.1827	0.3255096	1340	1552.0525	12.416247	1538.2	1530.71	10,915.827	1633.95	1635.18	**0.2049818**	1737.06
GWO	1457.5123	0.1310839	1457	1490.3533	**0.1909192**	**1504**	1573.16	3.6290719	1614.22	1699.23	0.4625355	**1603.77**
SSA	1457.6145	0.4723025	1470	1439.8766	7.053532	1450.2	1608.55	819.33097	1596.57	1709.22	0.5513827	1621.99
ALO	1380.1353	0.907234	1444	1501.3037	18.23078	1572.7	1547.62	350,044.88	1530.19	1633.06	0.4664427	1653.02
PSO	1457.5109	0.221618	**1392**	1424.0114	5.3852297	1540.8	1525.09	107.4649	1522.87	1677.05	0.3890662	1657.04
DE	1444.5022	0.1302138	1431	1421.3867	0.2648552	1546	1566.09	5.7233646	**1519.14**	1663.84	0.4342795	1610.35
GSA	1471.3442	0.870076	1405	1452.3351	8.7451512	1578	1566.62	9511.883	1615.14	1735.82	0.2446577	1656.86

Bold numbers represent the best values

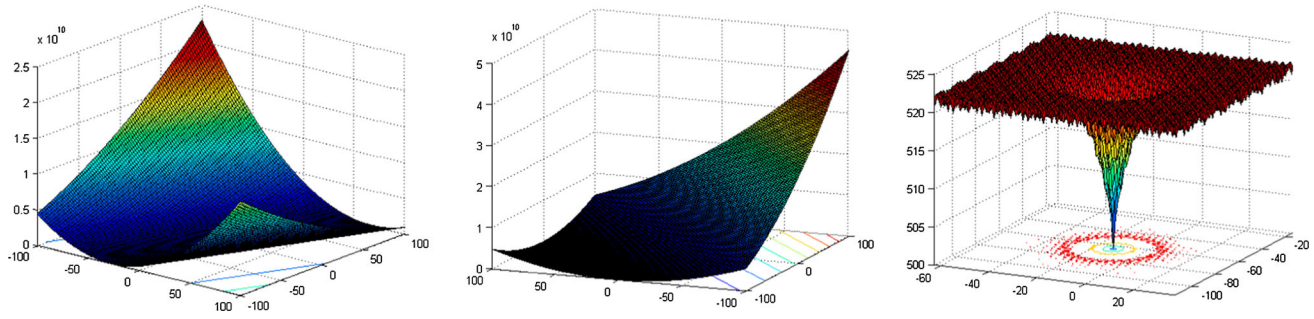


Fig. 7 3D structure of F14, F15, and F16 functions

Table 9 Results obtained from solving CEC2014 combined standard representations

	F22			F23			F24		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	**2500**	**0**	**2500**	**2600**	**0**	**2600**	**2700**	**0**	**2700**
GA	2748.3148	5.8429272	2787.2	2721.0841	2.3848824	2760.4	2811.0338	3.9165984	2849.6
GWO	2746.6802	6.3599307	2811.9	2766.9163	3.1528094	2797.6	2805.5017	4.9085358	2849.6
SSA	2670.9288	2.1432852	2698.6	2752.9106	33.171974	2776.8	2793.6216	16.93302	2822.2
ALO	2721.2069	1.6228404	2672.4	2725.2286	32.809296	2750.1	2788.7129	13.175748	2811.9
PSO	2672.2238	6.148272	2735.2	2782.7946	42.195054	2781	2806.3121	12.592632	2784.6
DE	2710.7562	19.140402	2808	2678.0084	0.005512	2678	2826.635	3.877847	2822.2
GSA	2759.2115	17.331604	2766.4	2659.4071	5.4450435	2688.3	2768.5318	17.368581	2828.8
	F25			F26			F27		
	AVG	STD	MED	AVG	STD	MED	AVG	STD	MED
SIFO	**2800.769**	**100.14584**	**2800**	**2900**	**0**	**2900**	**3000**	**0**	**3030**
GA	2881.7921	100.1502152	2881	2958	0	2987	3090	**0**	3151.2
GWO	2901.4135	141.570712	2884.6	3712.5562	106.92007	3855.6	4877.569	361.74373	5678.4
SSA	2884.2828	100.881382	2918.4	3385.5871	60.485184	3519	5610.5743	513.74952	7065.8
ALO	2923.3572	161.953984	2901.6	4072.2828	480.03356	4160	5922.2012	1120.4886	6000.8
PSO	2940.3414	169.563316	2932.5	4211.2077	414.9055	4274.4	5936.6541	1097.5672	6035.8
DE	2924.4071	157.992296	2928.8	4059.0527	382.53984	4066.4	5453.9283	870.50891	5520.8
GSA	2882.3581	100.8285664	2854	3391.3789	260.02034	3590.4	4409.4127	275.50292	5064.8

Bold numbers represent the best values

to leave the local optimization space and enter the global optimization space by using operators that have the property of causing mutations in the production responses. A 3D structure can be drawn for other functions (see Fig. 7).

6.4 Numerical results obtained from the limit of CEC2014 combined examples

In this section, six combined numerical examples presented at CEC2014 are considered as evaluation functions and the results of the solution are described as the last numerical analysis presented to test the proposed algorithm using standard examples. These examples can assess the exploitation phase of the proposed algorithm since the space of solutions is designed in a way that it has made the discovery of a global optimization solution in the solution space extremely complicated. In other words, the extend of the solution space leads to the appropriate implementation of the exploration phase during an acceptable period. However, finding a solution to the exploitation phase faces

Table 10 Results obtained from the comparison of GWO, SSA, and SIFO results

Instance	Size			Results								
	Supplier	Manufacturer	Distribution centre	SIFO			GWO			SSA		
				Objective function value	Deviation	Require number of iterations	Objective function value	Deviation	Require number of iterations	Objective function value	Deviation	Require a number of iterations
Medium	15	9	20	**900,646**	5447	71	1,350,969	7625.8	56	2,026,454	9150.96	88
	15	9	25	**684,040**	5823	101	889,252	7569.9	81	889,252	9840.87	145
	18	9	27	**601,712**	5476	46	782,225.6	8214	36	938,670.7	12,321	52
	18	9	32	**235,710**	6213	40	**235,710**	6213	30	329,994	9319.5	47
	21	10	34	**642,520**	7444	87	899,528	11,166	66	1,349,292	16,749	100
	21	10	39	**84,891**	9459	52	110,358.3	14,188.5	46	110,358.3	14,188.5	61
	27	10	48	**993,107**	7292	88	1,191,728	10,208.8	65	1,787,593	11,229.68	99
	24	10	41	**129,627**	9796	62	168,515.1	13,714.4	55	168,515.1	20,571.6	85
	24	10	11	**308,880**	7871	53	401,544	9445.2	39	401,544	10,389.72	70
	30	10	50	**367,890**	6124	32	515,046	8573.6	26	618,055.2	9430.96	43
Large	33	10	60	**312,993**	5956	43	**312,993**	5956	34	375,591.6	6551.6	61
	33	10	65	**290,586**	6970	78	348,703.2	9061	67	488,184.5	12,685.4	97
	39	10	80	**668,889**	6095	95	802,666.8	6095	80	1,043,467	9142.5	109
	36	10	70	**342,566**	5714	73	479,592.4	6856.8	61	479,592.4	8228.16	84
	36	10	75	**916,787**	9059	90	1,100,144	9059	63	1,650,217	9059	119
	39	10	85	**789,653**	6526	116	868,618.3	9136.4	95	955,480.1	10,050.04	136
	41	10	90	**641,945**	9729	104	**641,945**	13,620.6	75	834,528.5	16,344.72	116
	45	10	110	**643,529**	8533	100	900,940.6	11,946.2	75	1,171,223	14,335.44	136
	45	10	120	**480,899**	9413	67	673,258.6	10,354.3	59	875,236.2	14,496.02	84
	41	15	95	**881,236**	9629	70	1,145,607	14,443.5	60	1,489,289	15,887.85	101

Bold numbers represent the best values

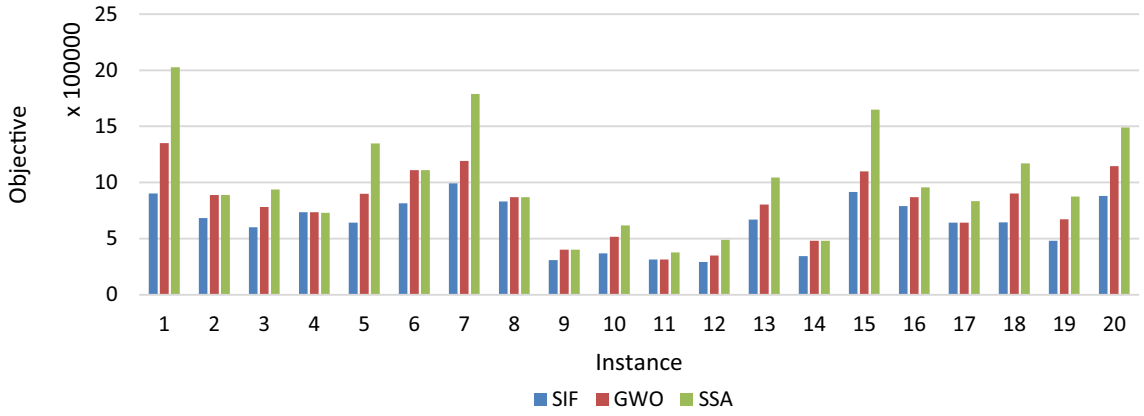


Fig. 8 Comparison of target function values of GWA, SSA, and SIFO

Fig. 9 Comparison of deviation amounts of GWA, SSA, and SIFO

serious challenges. Table 9 presents the numerical results obtained from the implementation of these standard examples.

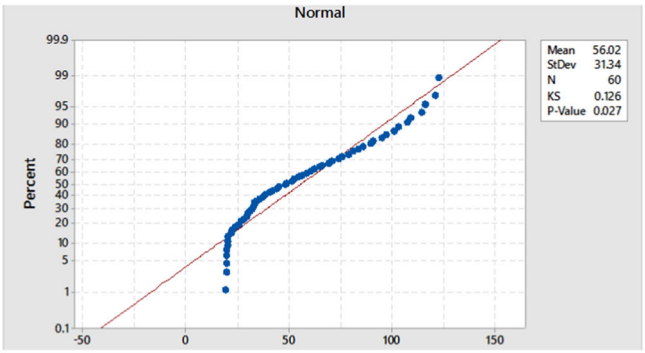
As observed, the proposed algorithm provided better solutions in solving the combined representations, compared to the other algorithms presented in the research. Therefore, it could be expressed that the application of the proposed algorithm can create a suitable condition for finding the final solution in the global optimization solution space. To sum up, the SIFO optimizer was able to create a proper situation for a general assessment of the solution space using the mechanism of production of multiple populations in the form of search teams and applying various search factors in each team. Moreover, the algorithm reinforced the discovery phase by integrating search teams in various iterations. In addition, using the Levy flight operator to transfer the elite members resulted in a suitable convergence in the problem-solving procedure. A numerical example corresponding to real-world conditions is solved using this algorithm, and the numerical results are fully described to examine the application of the proposed algorithm in solving engineering problems more precisely.

6.5 Solving constrained optimization problems

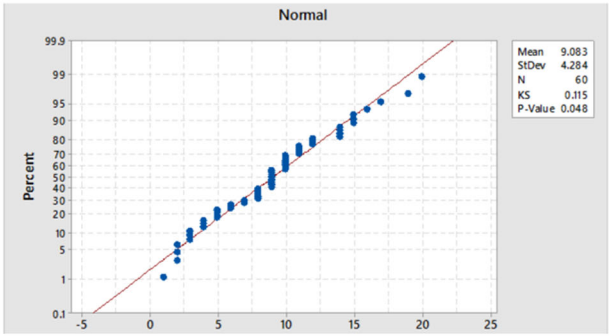
As mentioned at the beginning of this section, two numerical examples in the area of supply field design are evaluated as constrained optimization problems in order to more accurately evaluate the performance of the proposed algorithm.

6.5.1 Classic supply chain design

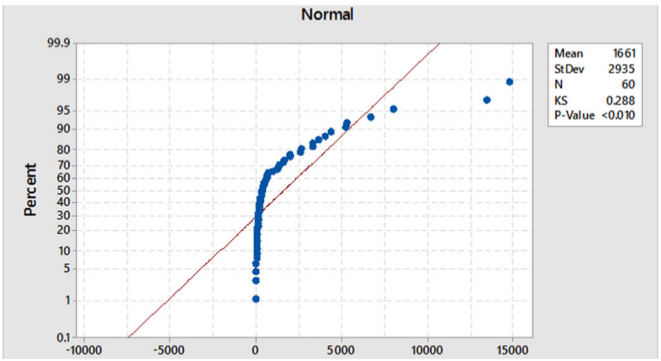
In brief, supply chain management is one of the basic issues in the fields of management, economics, industrial engineering, planning, and other related fields. A part of issues in the area of supply chain management is defined as optimization problems and in the field of operations research (Zou et al. 2016a). In this set of problems, the flow of goods, from the beginning of the production path, which



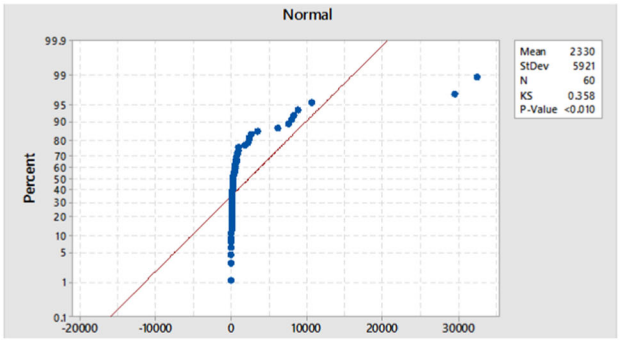
K-S test for the time of solution



K-S test for standard deviation criterion



K-S test for the required number of iterations



K-S test for the amount of target function

Fig. 10 K-S test diagrams for various criteria

Table 11 Final results of testing the normal distribution of the criteria

Criterion	Time	Number of iterations	Target function value	Standard deviation
P-Value	< 0.027	0.048	< 0.010	< 0.010
Result	Rejection of the null hypothesis	Rejection of the null hypothesis	Rejection of the null hypothesis	Rejection of the null hypothesis

Table12 Kruskal–Wallis test results

Criterion	Time	Number of iterations	Target function value	Standard deviation
P-Value	< 0.012	0.041	< 0.016	< 0.126
Result	Rejection of the null hypothesis	Rejection of the null hypothesis	Rejection of the null hypothesis	Rejection of the null hypothesis

includes the supply of raw materials, manufacturing in production centres, and distribution of the final product, is managed in a way that the final cost of the system is minimized. In this research, the data extracted from the research by (Beamon 1998) are used, which is recognized as one of the most complete classic models in this field. In this research, 20 numerical examples were generated according to the structure of the basic article to more accurately evaluate the behaviour of the SIFO optimizer. In Table 10 the numerical results obtained from comparison with GWA, SSA, and SIFO optimizer were described. Notably, the comparison criteria included the value of the objective function, the amount of deviation of solutions, and the number of iterations required to achieve the final solution.

Table 13 Indices and sets

I: Index of the set of suppliers
J: Index of the set of producers
K: Index of the set of retailers
L: Index of the set of assembly centres
M: Index of the set of destruction centres
N: Index of the set of capacity level points for all facilities
E: Index of the set of common points between points of production centres (*J*) and points of assembly centres (*L*)
F: Index of the set of modules used (optional or mandatory)

Table 14 Input parameters

D_k: Demand of retailer k
R_k: Average deduction of returned product quantity from retailer k (%)
S: Average deduction of the returned product in collection centres (%)
f_i^n: Fixed cost of activation of supplier i with capacity level n
g_j^n: Fixed cost of activation of producer j with capacity level n
h_l^n: Fixed cost of activation of a collection centre at location l with capacity n
b_m^n: Fixed cost of activation of destruction centre at location m with capacity n
$f_{et}^{nn'}$: A fixed amount of savings from the integration of the production centre with capacity n with the collection centre with capacity n' in location e
cx_{ij}: Total costs of transferring a product unit from the supplier centre I to collection centre j
cu_{jk}: Total costs of transferring a product unit from the production centre j to retailer k
cq_{kl}: Total costs of transferring a returned product unit from retailer k to collection centre l
cp_{li}: Total costs of displacement of a returned product unit from collection centre l to supplier i
ct_{lm}: Total costs of transferring a returned product unit from collection centre l to destruction centre m
caw_i^n: Capacity level n of potential supply centre I (direct flow)
cay_j^n: Capacity level n of potential production centre j
caz_l^n: Capacity level n of potential collection centre l
cav_m^n: Capacity level n of potential destruction centre m
car_i^n: Capacity level n of potential recovery centre I (reverse flow)
$cost_f$: Module production cost f
$value_f$: Value created for the customer if module f is provided
S_{kf}: Value 1 if module *f* is mandatory for client *k* and equal to zero if module f is optional
B_j^f: an input parameter that has the ability to produce module f if the producer *j* is one
M: A large enough integer

Table 15 Decision variables

K_{kf}: Equal to one if module f is used for customer *k*; otherwise, it is zero
X_{ij}: Raw material flow rate from supplier i to producer *j*
U_{jk}: Product flow rate from producer *j* to retailer *k*
W_{kl}: Product flow rate from retailer *k* to collection centre *l*
p_{li}: Product flow rate from collection centre *l* to supplier *i*
t_{lm}: Product flow rate from collection centre *l* to destruction centre *m*
w_i^n: A binary variable that is activated in case of one value for a supplier *I* at the capacity level *n*
y_j^n: A binary variable that is activated in case of one value for producer *j* at the capacity level *n*
z_l^n: A binary variable that is activated in case of one value for collection centre *I* at the capacity level n
v_m^n: A binary variable that is activated in case of one value for destruction centre *m* at the capacity level n

According to Table 10, the SIFO optimizer had lower objective function values, compared to the other algorithms, which is also observed in Fig. 8. In fact, the algorithm is able to conduct a better search in the solution space and report more appropriate results. This is mainly due to the presence of robust operators based on the crossover and mutation in the algorithm.

Regarding the amount of deviation, the SIFO optimizer had the lowest value in this respect, whereas SSA had the highest value, which showed the higher efficiency of the proposed algorithm in solving problems in this field.

As observed, in Fig. 9 there was a large gap between the solutions provided by the SIFO optimizer and SSA, while GWO produced solutions similar to SIFO optimizer in some cases. Therefore, it is necessary to more analyse the comparison of the GWA and SIFO optimizer, for which statistical tests are used to make a more accurate comparison. Since the tests compared two different societies, the GWO and SIFO optimizer, which had the highest efficiency, was selected as the criterion algorithms and the necessary estimations were carried out. In choosing a statistical test for research, it is necessary to decide on the use of parametric and nonparametric tests in a specific way. In this regard, one of the main criteria for selection is conducting the Kolmogorov–Smirnov test, which shows the normal or abnormal distribution of the data. In general, data have a normal distribution if the mentioned test is rejected, which means that the parametrical statistical tests can be used for research. On the other hand, confirmation of the Kolmogorov–Smirnov test is interpreted as an abnormal distribution of the data, which means that nonparametrical tests should be used in the research. Meanwhile, significant results of the Kolmogorov–Smirnov test

Table16 Multi-objective function

15	$MinW = \left(\sum_{i\epsilon I}\sum_{n\epsilon N} f_i^n W_i^n + \sum_{j\epsilon J}\sum_{n\epsilon N} g_j^n Y_j^n + \sum_{l\epsilon L}\sum_{n\epsilon N} h_l^n Z_l^n + \sum_{m\epsilon M}\sum_{n\epsilon N} b_m^n V_m^n - \sum_{e\epsilon E}\sum_{n'\epsilon N}\sum_{n\epsilon N} f_e^{nn'} z_e^{n'} y_e^n + \sum_{i\epsilon I}\sum_{j\epsilon J} cx_{ij} X_{ij} + \sum_{j\epsilon J}\sum_{k\epsilon K} cu_{jk} U_{jk} + \sum_{k\epsilon K}\sum_{l\epsilon L} cq_{kl} Q_{kl} + \sum_{l\epsilon L}\sum_{m\epsilon M} ct_{lm} T_{lm} + \sum_{l\epsilon L}\sum_{i\epsilon I} cp_{li} P_{li} + \sum_{f\epsilon F}\sum_{k\epsilon K} cost_f K_{kf}\right) - \sum_{f\epsilon F}\sum_{k\epsilon K} value_f K_{kf} \sum_{j\epsilon J} u_{jk}$	
	s.t	
16	$\forall k\epsilon K$	$\sum_{jJ} U_{jk} = d_k$
17	$\forall k\epsilon K$	$\sum_{lL} Q_{kl} = r_k d_k$
18	$\forall j\epsilon J$	$\sum_{iI} X_{ij} = \sum_{kK} U_{jk}$
19	$\forall l\epsilon L$	$\sum_{mM} T_{lm} = S \sum_{kK} Q_{kl}$
20	$\forall l\epsilon L$	$\sum_{iI} P_{li} = (1-S) \sum_{kK} Q_{kl}$
21	$\forall i\epsilon I$	$\sum_{jJ} X_{ij} \le \sum_{nN} W_i^n caw_i^n$
22	$\forall j\epsilon J$	$\sum_{iI} X_{ij} \le \sum_{nN} Y_j^n cay_j^n$
23	$\forall j\epsilon J$	$\sum_{kK} U_{jk} \le \sum_{nN} Y_j^n cay_j^n$
24	$\forall l\epsilon L$	$\sum_{kK} Q_{kl} \le \sum_{nN} Z_l^n caz_l^n$
25	$\forall m\epsilon M$	$\sum_{lL} T_{lm} \le \sum_{nN} V_m^n cav_m^n$
26	$\forall i\epsilon I$	$\sum_{lL} P_{li} \le \sum_{nN} W_i^n car_i^n$
27	$\forall l\epsilon L$	$\sum_{mM} T_{lm} + \sum_{i} P_{li} \le \sum_{nN} Z_l^n caz_l^n$
28	$\forall i\epsilon I$	$\sum_{lL} P_{li} \le M \sum_{jJ} X_{ij}$
29	$\forall i\epsilon I$	$\sum_{nN} W_i^n \le 1$
30	$\forall j\epsilon J$	$\sum_{nN} Y_j^n \le 1$
31	$\forall l\epsilon L$	$\sum_{nN} Z_l^n \le 1$
32	$\forall m\epsilon M$	$\sum_{nN} V_m^n \le 1$
33	$\forall k\epsilon K, f\epsilon F, S_{kf} = 1$	$K_{kf} = 1$
34	$\forall j\epsilon J, f\epsilon F$	$\sum_{kK} U_{jk} S_{kf} \le M \times B_j^f$
35	$\forall i\epsilon I,\ \forall j\epsilon J,\ \forall l\epsilon L,\ \forall m\epsilon M$	$W_i^n, Y_j^n,\ Z_l^n,\ V_m^n \in \{0,1\}$
36	$\forall i\epsilon I,\ \forall j\epsilon J,\ \forall k\epsilon K,\ \forall l\epsilon L,\ \forall m\epsilon M$	$X_{ij},\ U_{jk},\ Q_{kl},\ T_{lm},\ P_{li} \ge 0$

Table 17 Parameters and their values of the proposed SIFO algorithm with other algorithms

Algorithms	Parameters	Optimal value
Search in forest optimizer	$M = 10,\ N = 40,\ c = 0.3$	**6104.0226**
Equilibrium Optimizer (2019)	$a1 = 2;\ a2 = 1;$ GP = 0.5	12,000.41
Marine Predators Algorithm (2020)	P = 0.5; fads = 0.2	8602.74824
Slime mould algorithm (2020)	z = 0.03	9147.5764
Horse herd optimization algorithm (2021)	W = 0.95	7881.1196
Wild horse optimizer (2021)	PS = 0.5, PC = 0.33	7230.7444

Table18 Customer demand

Customer	Demand	Customer	Demand	Customer	Demand	Customer	Demand	Customer	Demand
1	120	5	95	9	114	13	88	17	84
2	85	6	56	10	121	14	75	18	95
3	65	7	74	11	78	15	109	19	100
4	110	8	91	12	68	16	118	20	110

Table19 Average deduction of returned product from customers (%)

Customer	Demand	Customer	Demand	Customer	Demand	Customer	Demand	Customer	Demand
1	5	5	8	9	6	13	3	17	5
2	7	6	4	10	8	14	8	18	5
3	10	7	12	11	5	15	10	19	4
4	5	8	4	12	4	16	11	20	9

Table 20 Fixed cost of using suppler with different capacity levels (million Rials)

	Capacity level 1	Capacity level 2	Capacity level 3
Supplier 1	1831	1651	1350
Supplier 2	1253	1142	1857
Supplier 3	1416	1385	1509
Supplier 4	1675	1415	1817

(P-value ≤ 0.05) show the abnormal distribution of the data and the need for using nonparametric tests since confirmation of the test is defined as nonparametric data. Figure 10 diagrams show the result of the assessment of all defined criteria in the test.

As observed, K–S test results are indicative of the abnormal distribution of all criteria. Therefore, nonparametric tests should be used to compare the two communities. In this research, we used the Kruskal–Wallis test, the results of which are presented in Tables 11 and 12.

As observed, the two algorithms are different in terms of all criteria, and SIFO optimizer has higher efficiency, compared to GWO, at all times.

However, the difference is not significant, and it can be expressed that the two algorithms had similar functions. As

Table21 Fixed cost of construction of production centres with different capacity levels (million Rials)

	Capacity level 1	Capacity level 2	Capacity level 3
Potential centre 1	316	429	323
Potential centre 2	445	366	343
Potential centre 3	474	391	474
Potential centre 4	448	403	362
Potential centre 5	416	469	485
Potential centre 6	445	418	315

Table 22 Fixed cost of construction of collection centres with different capacity levels (million Rials)

	Capacity level 1	Capacity level 2	Capacity level 3
Potential centre 1	391	438	340
Potential centre 2	366	359	428
Potential centre 3	422	329	381
Potential centre 4	392	463	450

Table 23 Fixed cost of construction of destruction centres with different capacity levels (million Rials)

	Capacity level 1	Capacity level 2	Capacity level 3
Potential centre 1	254	157	312
Potential centre 2	214	187	319
Potential centre 3	267	164	327
Potential centre 4	219	155	355

Table 24 Total costs of transferring a product unit from supplier i to producer j (hundred Rials)

	Producer 1	Producer 2	Producer 3	Producer 4	Producer 4	Producer 5
Supplier 1	17	12	30	12	29	16
Supplier 2	30	17	16	15	11	17
Supplier 3	24	10	22	14	17	18
Supplier 4	12	25	17	16	15	23

such, SIFO implementation can lead to highly efficient solutions for problems related to the management chain field.

6.6 Advanced supply chain design problem solving

In this research, we focus on the design of a reverse supply chain to minimize the costs of facility location and material transfer between different categories of the chain in question. Evaluation of this issue is important since determining the proper location of the facility and the proper flow of products transferred between the levels of the chain can improve the performance of the chain and ultimately achieve a suitable profit margin for industry managers. The proposed model is explained in Tables 13, 14, 15, 16.

The first sentence of the objective function calculates the cost of activation of a supply centre. The second sentence of the objective function shows the cost of activation of producers. In addition, the third sentence estimates the activation cost of collection centres, whereas the fourth sentence calculates the cost of activation of a destruction centre. Moreover, the fifth sentence calculates savings resulted from matching the centres. Notably, the sentence is presented in the target function with a negative symbol since it must be maximized. Furthermore, the sixth sentence shows the transfer of producers from suppliers to producers, while the seventh sentence estimates the costs of transfer from producer to retailer. The eighth sentence calculates the cost of transfer of returned products from retailer to collection centres, and the ninth centre estimates the cost of transferring products from collection centres to destruction centres. Finally, the 10th sentence shows the entire costs of moving a returned product unit from the collection centre to the supplier. Phrases (16) and (17) guarantee that all demands of customers are responded to the direct flow, and all returned products are collected from retailers in the reverse flow. In addition, phases (18–20) are related to flow balance constraints at nodes. In fact, it is guaranteed in these constraints that the products that enter each chain level must exist from the same level. Moreover, constraints (19–26) guarantee that the flow exists only between points where facilitation has been activated and the entire flow does not exceed the capacity of each facilitation. Furthermore, constraints (29–33) guarantee that only one level of capacity is allocated to each facilitation, whereas constraint (31) guarantees that the necessary module exists in the products sent to the desired customer if the modules are mandatory. Additionally, constraint (34) guarantees that customer demand can only be met by a manufacturer when that manufacturer has the ability to add a mandatory customer module. Finally, constraints (35–36) are logical and obvious limitations related to decision variables of a problem.

Table 25 Total costs of transferring a product unit from the production centre j to customer centre k (hundred Rials)

Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer
5	17	6	13	9	9	9	5	6	1
7	18	5	14	5	10	9	6	5	2
7	19	8	15	9	11	5	7	6	3
5	20	8	16	6	12	7	8	9	4
Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer
9	17	9	13	7	9	9	5	6	1
5	18	7	14	9	10	5	6	6	2
8	19	9	15	7	11	6	7	8	3
5	20	7	16	6	12	7	8	7	4
Production centre 3	Customer	Production centre 3	Customer	Production centre 3	Customer	Production centre 3	Customer	Production centre 3	Customer
9	17	6	13	6	9	7	5	7	1
6	18	5	14	5	10	5	6	5	2
7	19	5	15	5	11	5	7	7	3
6	20	9	16	9	12	7	8	6	4
Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer
8	17	88	13	114	9	95	5	120	1
9	18	75	14	121	10	56	6	85	2
5	19	109	15	78	11	74	7	65	3
9	20	118	16	68	12	91	8	110	4
Production centre 5	Customer	Production centre 5	Customer	Production centre 5	Customer	Production centre 5	Customer	Production centre 5	Customer
7	17	6	13	5	9	7	5	8	1
6	18	6	14	5	10	8	6	7	2
9	19	6	15	8	11	6	7	7	3
9	20	8	16	8	12	9	8	5	4
Production centre 6	Customer	Production centre 6	Customer	Production centre 6	Customer	Production centre 6	Customer	Production centre 6	Customer
5	17	5	13	9	9	7	5	9	1
6	18	7	14	6	10	5	6	8	2
7	19	8	15	7	11	9	7	9	3
7	20	6	16	8	12	6	8	9	4

6.6.1 Comparison of results obtained from supply chain design problem solving values by SIFO optimizer and latest algorithm

The supply chain model is solved using the random values by SIFO optimizer and latest algorithm, Equilibrium Optimizer (EO) (Farmarzi et al. 2020), Marine Predators Algorithm (MPA) (Farmarzi et al. 2020), Slime mould algorithm (SMA) (Li et al. 2020), Horse herd optimization algorithm (HOA) (MiarNaeimi et al. 2021), Wild horse optimizer (WHO) (Naruei and Keynia 2021), and the results are compared in Table 17. The results of solved supply chain model by SIFO optimizer indicated better optimal values than the MPA, EO, SMA, HOA, and WHO.

Table 26 Total costs of transferring a product unit from customer k to collection centres (hundred Rials)

Collection centre 1	Customer	Collection centre 1	Customer	Collection centre 1	Customer	Collection centre 1	Customer	Collection centre 1	Customer
2	17	3	13	3	9	5	5	3	1
2	18	2	14	2	10	5	6	3	2
3	19	3	15	4	11	4	7	2	3
4	20	4	16	1	12	7	8	4	4
Collection centre 2	Customer	Collection centre 2	Customer	Collection centre 2	Customer	Collection centre 2	Customer	Collection centre 2	Customer
3	17	6	13	5	9	4	5	4	1
2	18	5	14	2	10	2	6	2	2
2	19	3	15	3	11	2	7	2	3
3	20	4	16	5	12	3	8	2	4
Collection centre 3	Customer	Collection centre 3	Customer	Collection centre 3	Customer	Collection centre 3	Customer	Collection centre 3	Customer
3	17	5	13	4	9	3	5	5	1
2	18	5	14	2	10	3	6	5	2
4	19	2	15	3	11	2	7	2	3
2	20	2	16	4	12	4	8	4	4
Collection centre 4	Customer	Collection centre 4	Customer	Collection centre 4	Customer	Collection centre 4	Customer	Collection centre 4	Customer
4	17	8	13	12	9	10	5	10	1
7	18	12	14	14	10	11	6	15	2
5	19	12	15	10	11	12	7	12	3
7	20	10	16	10	12	10	8	14	4

Table 27 Total costs of moving a returned product unit from collection centre l to supplier I (hundred Rials)

	Producer 1	Producer 2	Producer 3	Producer 4
Collection centre 1	5	4	5	9
Collection centre 2	7	7	6	6
Collection centre 3	8	5	5	4
Collection centre 4	6	9	5	5

Table 28 Total costs of moving a returned product unit from collection centre l to destruction centres (hundred Rials)

	Destruction centre 1	Destruction centre 2	Destruction centre 3	Destruction centre 4
Collection centre 1	3	2	4	3
Collection centre 2	2	3	2	3
Collection centre 3	4	2	3	2
Collection centre 4	3	3	3	4

Table29 Cost of loading different modules

	Module 1	Module 2	Module 3	Module 4	Module 5
Cost	50	45	60	52	48
Value	58	43	62	50	45

So, the SIFO is presented to solve industrial engineering and supply chain problems as an efficient metaheuristic algorithm. The parameters of each of the mentioned algorithms are adjusted based on Table 17.

6.6.2 Evaluation of results obtained from supply chain design problem solving

In the numerical example of the present model, we assume that a supply chain network is being designed, where four potential points are available for the activation of suppliers, whereas 6, 20, 4, and 4 potential points exist for activation of production centres, customers, the establishment of collection centres, and construction of destruction centres, respectively. Overall, two suppliers, three producers, two collection centres, and two destruction centres must be constructed among the potential points. Also, all centres have three capacity levels. The amount of customer demand is presented in Tables 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30.

It is possible that all producers are able to upload any module for their customers in order to create more complications in problem-solving process. After solving the problem using the proposed algorithm, the convergence diagram is presented in Fig. 11.

As observed, the SIFO optimizer was suitably converged towards the optimal solution, and the mean of solutions matched the best solution in the final iterations. This shows that the proposed algorithm has efficiently guided all members in search teams towards the final solution. The structure of the supply chain can also be described in Tables 31, 32, 33, 34.

Table 30 Determining mandatory (1) or optional (0) modules

Module 1	Customer	Module 1	Customer	Module 1	Customer	Module 1	Customer	Module 1	Customer
1	17	0	13	0	9	1	5	0	1
0	18	0	14	0	10	0	6	0	2
0	19	1	15	0	11	0	7	1	3
0	20	0	16	0	12	1	8	0	4
Module 2	Customer	Module 2	Customer	Module 2	Customer	Module 2	Customer	Module 2	Customer
0	17	1	13	1	9	1	5	0	1
0	18	0	14	0	10	0	6	0	2
1	19	0	15	0	11	0	7	0	3
0	20	1	16	1	12	1	8	0	4
Module 3	Customer	Module 3	Customer	Module 3	Customer	Module 3	Customer	Module 3	Customer
1	17	0	13	0	9	0	5	0	1
0	18	0	14	0	10	0	6	0	2
0	19	0	15	1	11	0	7	1	3
1	20	0	16	0	12	0	8	0	4
Module 4	Customer	Module 4	Customer	Module 4	Customer	Module 4	Customer	Module 4	Customer
0	17	0	13	0	9	0	5	0	1
0	18	0	14	0	10	0	6	0	2
0	19	1	15	0	11	1	7	1	3
0	20	0	16	0	12	0	8	0	4
Module 5	Customer	Module 5	Customer	Module 5	Customer	Module 5	Customer	Module 5	Customer
0	17	0	13	0	9	1	5	0	1
0	18	0	14	0	10	0	6	0	2
0	19	1	15	0	11	0	7	1	3
0	20	0	16	0	12	1	8	0	4

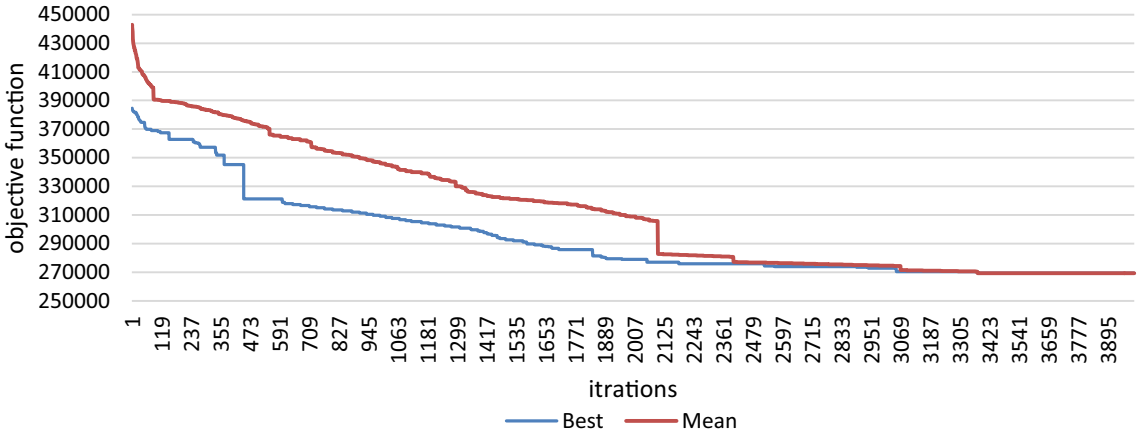


Fig. 11 Diagram of convergence obtained from supply chain design problem-solving

Table 31 Mandatory and optional allocation of modules to products of customers

	Module 1	Module 2	Module 3	Module 4	Module 5
Customer 3	*		*	*	*
Customer 5	*	*			*
Customer 7				*	
Customer 8	*	*			*
Customer 9		*			
Customer 11			*		
Customer 12		*			
Customer 13		*			
Customer 15	*			*	*
Customer 16		*			
Customer 17	*		*		
Customer 19		*			
Customer 20			*		

The supplier 1 with capacity level 1 and supplier 4 with capacity level 2 are selected to supply the products required by producers. In addition, producers 1, 2, and 4 with capacity level 1 and producer 6 with capacity level 3 are activated. Moreover, the collection centre 1 with capacity level 1, collection centre 3 with capacity level 2, and collection centres 2 and 4 with capacity level 3 are constructed to collect products used by customers. Furthermore, destruction centres 2, 3, and 4 with capacity level 2 are constructed to destroy the nonrecyclable part of products. In the management of material flow, 151, 1211, and 84 units are delivered to producer 1, 2, and 3, respectively, by supplier 1. In addition, the supplier 4 sends 410 product units to producer 1. Information related to the flow of products from manufacturers to customers is also in accordance with Table 33.

As observed, all demands of each customer are met based on the flow of presented products, which shows the proper performance of the proposed model. The rate of return of products from customers to collection centres is also presented in Table 34.

As observed, the necessary allocations are made based on the parameter of the percentage of returned products and cost of transferring products between customers and each centre in order to conduct the product collection operations with the lowest cost possible. Table 35 shows the product flow between the collection centres and the suppliers, aiming at reusing the recycled materials obtained in the collection centres.

As observed, the total products delivered to suppliers are equal to the deduction of recyclable producers delivered by customers to collection centres, which makes the return of products into the production cycle possible. However, a part of the products is unrecyclable and is directly sent to destruction centres to carry out the destruction stages. The flow of products is presented in Table 35.

Table 32 Optimal flow of products between producers and customers

Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer	Production centre 1	Customer
	17		13	114	9	95	5		1
	18	75	14	121	10		6		2
	19	109	15		11		7	65	3
	20	118	16		12	91	8		4
Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer	Production centre 2	Customer
	17	88	13		9		5	120	1
95	18		14		10	56	6	85	2
100	19		15	78	11	74	7		3
110	20		16	68	12		8	110	4
Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer	Production centre 4	Customer
84	17		13		9		5		1
	18		14		10		6		2
	19		15		11		7		3
	20		16		12		8		4

Table 33 Optimal flow of returned products between customers and collection centres

	Collection centre 1	Collection centre 2	Collection centre 3	Collection centre 4
Customer 1			60	
Customer 2			60	
Customer 3	7			
Customer 4		55		
Customer 5			76	
Customer 6			23	
Customer 7		9		
Customer 8		37		
Customer 9			69	
Customer 10			97	
Customer 11		1		38
Customer 12	28			
Customer 13	27			
Customer 14			60	
Customer 15		11		
Customer 16	13			
Customer 17				42
Customer 18		2	46	
Customer 19		40		
Customer 20			99	

Table 34 The flow of products between collection centres and suppliers

	Supplier 1	Supplier 2
Collection centre 1	60	0
Collection centre 2	0	124
Collection centre 3	0	472
Collection centre 4	0	64

Table 35 The flow of products between collection and destruction centres

	Destruction centre 2	Destruction centre 3
Collection centre 1	15	0
Collection centre 2	31	0
Collection centre 3	118	0
Collection centre 4	0	16

7 Conclusions

This article evaluates a new SIFO-based structure to present a new optimal problem solution approach as a metaheuristic algorithm. The mathematical structure of the algorithm involves determining a specific number of groups in the form of teams, each encompassing a number of search members. Two different types of search are carried out in each iteration of the algorithm; the first type is an intragroup search, which is adapted from the organizing structure of search teams to accurately inspect the region, and the second type is local search of all teams in the solution space in order not to cover the entire solution space. This guarantees the effective and parallel assessment of the whole solution space. One of the advantages of the present research is the exploitation of parallel search operators in the algorithm structure, which leads to a proper balance in the diversification and contemplation phases of the algorithm. In addition, standard functions presented in two unimodal and multimodal modes are used to evaluate the efficiency of the proposed algorithm (Digalakis and Margaritis 2001; Molga and Smutnicki 2005). According to the numerical results, the SIFO optimizer yielded better results in numerical representations, compared to GA and ALO. SIFO optimizer can also be used in optimization problems as an efficient algorithm. Moreover, SIFO optimizer yielded far better solutions in solving CEC2014 standard numerical representations, compared to the other algorithms. Furthermore, according to the result obtained from supply chain design problem solving, the algorithm developed in the present research seemed to effectively find optimal solutions in the solution space and converge towards the global optimization solution when solving some practical issues in the field of industrial engineering. As such, the algorithm is proposed to solve optimization problems in various engineering fields. In order to expand the dimensions of research, it is recommended that multi-purpose SIFO optimizer is developed, and numerical results obtained from solving various optimization problems are compared to the existing algorithms.

Author contributions Amin Ahwazian helped in study concept and design, acquisition of data, analysis and interpretation of data, drafting of the manuscript, statistical analysis, critical revision of the manuscript for important intellectual content. Atefeh Amindoust and Reza Tavakkoli-Moghaddam supervised the research. Mehrdad Nikbakht was the research consultant.

Data availability The authors confirm that the data supporting the findings of this research are available within its supplementary materials.

Declarations

Conflict of interest The authors of the manuscript declare that they have completely avoided publishing ethics, including plagiarism, misconduct, data forgery, or double publishing. There are no commercial interests in this article, and the authors have not received any payment for their manuscript. The authors also claim that this manuscript has not been published elsewhere and has not been published in another journal. All rights to use the content, tables, images, etc., have been assigned to the publisher.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1007/s00500-021-06522-6.

References

Abbassi R, Abbassi A, Heidari AA, Mirjalili S (2019) An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models. Energy Convers Manage 179:362–372

Back T (1996) Evolutionary algorithms in theory and practice: Evolution strategies, evolutionary programming, and genetic algorithms: Oxford University Press. Oxford University Press, Oxford

Barisal A (2013) Dynamic search space squeezing strategy based intelligent algorithm solutions to economic dispatch with multiple fuels. Int J Electr Power Energy Syst 45:50–59

Beamon BM (1998) Supply chain design and analysis: Models and methods. Int J Prod Econ 55:281–294

Blackwell TM (2005) Particle swarms and population diversity. Soft Comput 9:793–802

Brownlee J (2011) clever algorithms: Nature-inspired programming recipes. Published by Jason Brownlee, Melbourne, Australia

Chen S, Chen R, Wang G-G, Gao J, Sangaiah AK (2018) An adaptive large neighborhood search heuristic for dynamic vehicle routing problems. Comput Electr Eng 67:596–607

Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6:58–73

Cui Z, Sun B, Wang G, Xue Y, Chen J (2017) A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber–physical systems. J Parallel Distrib Comput 52:42–17

Cui Z, Xue F, Cai X, Cao Y, Wang G-g, Chen J (2018) Detection of malicious code variants based on deep learning. IEEE Trans Industr Inf 14:3187–3196

Davis L (1991) Handbook of genetic algorithms

Digalakis JG, Margaritis KG (2001) On benchmarking functions for genetic algorithms. Int J Comput Math 77:481–506

Dorigo M, Birattari M (2010) Ant colony optimization. Springer, Berlin

Dréo J, Pétrowski A, Siarry P, Taillard E (2006) Metaheuristics for hard optimization: methods and case studies. Springer Science and Business Media, Berlin

Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: MHS'95 Proceedings of the Sixth International Symposium on Micro Machine and Human Science: IEEE. pp 39–43

Eddalya M, Jarbouia M, Siarryba P (2016) Combinatorial particle swarm optimization for solving blocking flow shop scheduling problem. J Comput Des Eng 3:295–311

Emara HM, Fattah HA (2004) Continuous swarm optimization technique with stability analysis. In: Proceedings of the American control conference: IEEE. pp 2811–2817

Faramarzi A, heidarnejad M, Mirjalili M, Gandomi A.H (2020) Marine Predators Algorithm: A Nature-inspired Metaheuristic. Exp Syst Appl, p 152

Faramarzi A, heidarnejad M, Stephens B. Mirjalili M (2020) Equilibrium optimizer: A novel optimization algorithm. Knowledge-Based Syst, p 191

Faris H, Ala'M A-Z, Heidari AA, Aljarah I, Mafarja M, Hassonah MA et al (2019) An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. Inf Fusion 83:67–42

Feng Y, Wang G-G, Li W, Li N (2018a) Multi-strategy monarch butterfly optimization algorithm for discounted 0–1 knapsack problem. Neural Comput Appl 30:3019–3036

Feng Y, Yang J, Wu C, Lu M, Zhao X-J (2018b) Solving 0–1 knapsack problems by chaotic monarch butterfly optimization algorithm with Gaussian mutation. Memetic Comput 10:135–150

García-Gonzalo E, Fernández-Martínez JL (2014) Convergence and stochastic stability analysis of particle swarm optimization variants with generic parameter distributions. Appl Math Comput 249:286–302

Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. SIMULATION 76:60–68

Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3:95–99

Kadirkamanathan V, Selvarajah K, Fleming PJ (2006) Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Trans Evol Comput 10:245–255

Kennedy J (2003) Bare bones particle swarms. Proceedings of the IEEE Swarm Intelligence Symposium SIS'03 (Cat No 03EX706): IEEE. pp 80–87

Kennedy J (2000) Stereotyping: Improving particle swarm performance with cluster analysis. Proceedings of the Congress on Evolutionary Computation CEC00 (Cat No 00TH8512): IEEE. pp 1507–1512

Kennedy J, Eberhart R (1995) Particle swarm optimization (PSO) Paper presented at the Proc. IEEE International Conference on Neural Networks, Perth, Australia

Koyuncu H, Ceylan R (2018) A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems. J Comput Des Eng 6:129–142

Li C, Yang S, Nguyen TT (2011) A self-learning particle swarm optimizer for global optimization problems. IEEE Trans Syst Man Cybern Part B (cybern) 7:62–42

Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: A new method for stochastic optimization. Futur Gener Comput Syst 111:300–323

Li X, Qian J, Wang G-g (2013) Fault prognostic based on hybrid method of state judgment and regression. Adv Mech Eng 5:1495–1562

MiarNaeimi F, Azizyan G, Rashki M (2021) Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems. Knowledge-Based Syst 213:106711

Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–169

Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61

Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98

Molga M, Smutnicki C (2005) Test functions for optimization needs. Test Funct Optim Needs, p 101

Naruei I, Keynia F (2021) Wild horse optimizer: a new meta-heuristic algorithm for solving engineering optimization problems. Eng Comput, pp 1–32

Nan X, Bao L, Zhao X, Zhao X, Sangaiah A, Wang G-G et al (2017) EPuL: an enhanced positive-unlabeled learning algorithm for the prediction of population sites. Molecules 22:1463

Pandey AS, Ehtesham PV, Hasan M, Parhi R (2020) DV-REP-based navigation of automated wheeled robot between obstacles using PSO-tuned feed forward neural network. J Comput Des Eng 7:427–434

Park J-B, Jeong Y-W, Shin J-R, Lee KY (2009) An improved particle swarm optimization for nonconvex economic dispatch problems. IEEE Trans Power Syst 25:156–166

Peng C-C, Chen C-H (2015) Compensatory neural fuzzy network with symbiotic particle swarm optimization for temperature control. Appl Math Model 39:383–395

Poli R, Kennedy J, Blackwell T (2007) Particle Swarm Optimization. Swarm Intell 1:33–57

Poli R (2008) Dynamics and stability of the sampling distribution of particle swarm optimizers via moment analysis. J Artif Evol Appl, p 15

Poli R (2009) Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. IEEE Trans Evol Comput 13:712–721

Rashidi P, Cook DJ (2009) Keeping the resident in the loop: adapting the smart home to the user. IEEE Trans Syst Man Cybern Part A 39:949–959

Rizk-Allah RM, El-Sehiemy RA, Deb S, Wang G-G (2017) A novel fruit fly framework for multi-objective shape design of tubular linear synchronous motor. J Supercomput 73:1235–1256

Selma B, Chouraqui S, Abouaïssa H (2020) Fuzzy swarm trajectory tracking control of unmanned aerial vehicle. J Comput Des Eng 7:435–447

Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: IEEE international conference on evolutionary computation proceedings IEEE world congress on computational intelligence (Cat No 98TH8360): IEEE. pp 69–73

Srikanth K, Panwar LK, Panigrahi BK, Herrera-Viedma E, Sangaiah AK, Wang G-G (2018) Meta-heuristic framework: quantum inspired binary grey wolf optimizer for unit commitment problem. Comput Electr Eng 70:243–260

Storn R, Price K (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11:341–359

Suganthan PN (1999) Particle swarm optimizer with neighborhood operator. In: Proceedings of the Congress on Evolutionary Computation-CEC99 (Cat No 99TH8406): IEEE. pp 1958–62

Talatahari S, Azar BF, Sheikholeslami R, Gandomi A (2012) Imperialist competitive algorithm combined with chaos for global optimization. Commun Nonlinear Sci Numer Simul 17:1312–1319

Talbi E-G (2009) Metaheuristics: from design to implementation. John Wiley and Sons, New Jersey

Tanweer MR, Suresh S, Sundararajan N (2016) Dynamic mentoring and self-regulation-based particle swarm optimization algorithm for solving complex real-world optimization problems. Inf Sci 326:1–24

Van Den Bergh F (2001) An analysis of particle swarm optimizers. University of Pretoria South Africa

Wang G, Guo L, Duan H, Liu L, Wang H, Shao M (2012a) Path planning for uninhabited combat aerial vehicle using hybrid meta-heuristic DE/BBO algorithm. Adv Sci Eng Med 4:550–564

Wang G-G, Chu HE, Mirjalili S (2016a) Three-dimensional path planning for UCAV using an improved bat algorithm. Aerosp Sci Technol 49:231–238

Wang G-G, Deb S, Gandomi AH, Zhang Z, Alavi AH (2016b) Chaotic cuckoo search. Soft Comput 20:334–296

Wang G-G, Gandomi AH, Alavi AH (2014) An effective krill herd algorithm with migration operator in biogeography-based optimization. Appl Math Model 38:2454–2462

Wang G-G (2018a) Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memetic Comput 10:151–164

Wang H, Yi J-H (2018) An improved optimization method based on krill herd and artificial bee colony with information exchange. Memetic Comput 10:177–198

Wang L, Fu X, Mao Y, Menhas MI, Fei M (2012b) A novel modified binary differential evolution algorithm and its applications. Neurocomputing 98:55–75

Wang L, Zheng X-L (2018b) A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem. Swarm Evol Comput 38:54–63

Wang G-G, Bai D, Gong W, Ren T, Liu X, Yan X (2018) Particle-swarm Krill Herd Algorithm. In: Paper presented at the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)

Wang G-G, Deb S, Coelho LDS (2015) Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Int J Bio-Inspired Comput 7:1–23

Wang G-G, Deb S, Gao X-Z, Coelho LDS (2016) A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. Int J Bio-Inspired Comput 8(6):394–409

Wu G, Pedrycz W, Suganthan PN, Mallipeddi R (2015) A variable reduction strategy for evolutionary algorithms handling equality constraints. Appl Soft Comput 37:774–786

Yang X-S, Gandomi AH, Talatahari S, Alavi AH (2012) Metaheuristics in water, geotechnical and transport engineering. Newnes, Elsevier, Amsterdam

Yang X-S (2008) Introduction to mathematical optimization: From linear programming to metaheuristics. Cambridge International Science Publishing Ltd, Cambridge

Yang, X.-S (2010) Nature-inspired metaheuristic algorithms: L Univer Press

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3:82–102

Yi J-H, Deb S, Dong J, Alavi AH, Wang G-G (2018) An improved NSGA-III Algorithm with adaptive mutation operator for big data optimization problems. Futur Gener Comput Syst 88:571–585

Yi J-H, Wang J, Wang G-G (2016) Improved probabilistic neural networks with self-adaptive strategies for transformer fault diagnosis problem. Adv Mech Eng 8:1687814015624832

Yi J-H, Xing L-N, Wang G-G, Dong J, Vasilakos AV, Alavi AH et al (2020) Behavior of crossover operators in NSGA-III for large-scale optimization problems. Inf Sci 509:87–47

Zhan Z-H, Zhang J, Li Y, Shi Y-H (2010) Orthogonal learning particle swarm optimization. IEEE Trans Evol Comput 15:832–847

Zhou C, Gao HB, Gao L, Zhang WG (2003) Particle swarm optimization (PSO) algorithm [J]. Appl Res Comput 12:7–11

Zou D, Li S, Wang G-G, Li Z, Ouyang H (2016a) An improved differential evolution algorithm for the economic load dispatch problems with or without valve-point effects. Appl Energy 181:375–390

Zou D, Wang G-G, Sangaiah AK, Kong X (2017) A memory-based simulated annealing algorithm and a new auxiliary function for the fixed-outline floor planning with soft blocks. J Ambient Intell Humaniz Comput, pp 1–12

Zou D-X, Deb S, Wang G-G (2018) Solving IIR system identification by a variant of particle swarm optimization. Neural Comput Appl 30:685–698

Zou D-x, Wang G-g, Pan G, Qi H-w (2016b) A modified simulated annealing algorithm and an excessive area model for floor planning using fixed-outline constraints. Front Inf Technol Electr Eng 17:1228–1244