Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 208 (2023) 95–135
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Original articles

Orchard Algorithm (OA): A new meta-heuristic algorithm for solving discrete and continuous optimization problems

Mehrdad Kaveh[a,*], Mohammad Saadi Mesgari[a,*], Bahram Saeidian[b]

[a] *Faculty of Geodesy and Geomatics, K. N. Toosi University of Technology, Tehran 19967-15433, Iran*
[b] *The Centre for Spatial Data Infrastructures and Land Administration (CSDILA), Department of Infrastructure Engineering, The University of Melbourne, VIC 3010, Melbourne, Victoria, Australia*

Received 11 July 2022; received in revised form 3 December 2022; accepted 18 December 2022
Available online 18 January 2023

Abstract

Meta-heuristic algorithms have been widely used to solve different optimization problems. There have always been ongoing efforts to develop new and efficient algorithms. In this paper, the Orchard Algorithm (OA) is designed and introduced, inspired by fruit gardening. In this process, various actions such as irrigation, fertilization, trimming, and grafting lead to a fruit orchard where most trees grow and produce fruit adequately. In OA, both explorations of the search space and exploitation of the best solutions are achieved using personal and social behavior. By introducing various operators such as annual growth, screening, and grafting, the algorithm can efficiently search and explore the search space. The performance of the proposed OA algorithm was evaluated on CEC2005, IEEE CEC06 2019,test functions, and five real-world engineering problems compared with 13 widely used and competitive algorithms. Thirty benchmark functions were used to compare the capabilities of the OA algorithm with other research. The OA yields far better results in many aspects than the other algorithms. The results show the OA's superiority and this algorithm's capability in solving optimization problems.

Keywords: Orchard Algorithm; Meta-heuristic; Optimization; Plants; Engineering problems

1. Introduction

Many issues in different areas of engineering can be considered optimization problems. Various optimization algorithms have already been developed to solve such problems efficiently [103]. The optimization methods can be classified into two categories: exact and meta-heuristic. Exact methods such as dynamic programming, linear and integer programming-based methods, and lagrangian relaxation-based methods can be used to find global solutions for optimization problems. However, their runtime increases dramatically with the increase in the problem size. Therefore, these methods are often used to solve small-size and medium-sized problems [68,101,106]. For real-world problems with extensive search space, the efficiency and applicability of such algorithms are reduced [20,107]. Puchinger and Raidl [101] indicated that using meta-heuristic algorithms is the only efficient way to solve such

* Corresponding authors.
E-mail addresses: m.kaveh11@email.kntu.ac.ir (M. Kaveh), mesgari@kntu.ac.ir (M.S. Mesgari), bsaeidian@student.unimelb.edu.au (B. Saeidian).

https://doi.org/10.1016/j.matcom.2022.12.027

problems. The limitation of meta-heuristic algorithms is that they may never find the global optimum. Instead, they find near-to-optimum solutions in a reasonable time [37,38,41,101].

By defining different exploration and exploitation operators, meta-heuristics can search the solution space efficiently without being trapped in local optimums [97]. They can find solutions close enough to the global optimum in a reasonable time [20]. Thus, these algorithms have been successfully used for different optimization problems [63,64,103,111,113,114]. On the other hand, for many types of combinatorial optimization problems, no exact method has been offered [84]. They traditionally could be solved using heuristic and approximate methods only. Therefore, it is reasonable to use meta-heuristics for solving such problems, where even the estimations of the solutions are difficult [107]. Meta-heuristic algorithms have become very popular in engineering problems. With the increasing complexity of problems, the need for new meta-heuristics becomes obvious more than before [29,46,97]. The reasons for this demand, can be summarized into five main motivations: (a) simple structure and concepts; (b) derivation-free mechanisms; (c) local optima avoidance; (d) flexibility; (e) simple and effective implementation [1,41,71,119].

Since a few decades ago, a few nature-inspired meta-heuristics, such as genetic algorithm (GA) [52], ant colony optimization (ACO) [24], particle swarm optimization (PSO) [26], simulated annealing (SA) [73], and differential evolution (DE) [122] have been introduced and used for different optimization problems. Afterward, many studies concentrated on improving or adapting these algorithms for new applications [20]. Other researchers tried to introduce new meta-heuristics by taking inspiration from nature. Some newer algorithms, such as the chimp optimization algorithm (ChOA) [71], capuchin search algorithm (CapSA) [14], and gannet optimization algorithm (GOA) [97], are the results of such efforts. Table 1 presents general information about some of the more popular algorithms.

The algorithms presented in Table 1 are inspired by Darwin's theory, swarm intelligence, social and political issues, and many other natural phenomena. However, the life and behavior of plants are rarely considered. Table 2 presents a few algorithms inspired by the plants, which are all population-based. In many other studies, the efficiency and applicability of the algorithms are evaluated and compared for different applications [2–4,10,13,28,30,66,67,69,72,112,114,129,139]. The general conclusion is that each algorithm has its advantages and limitations [1,41,119]. According to the no free lunch (NFL) theorem, no algorithm is better than others in all aspects and for all applications [68,112,114]. However, if the performance of an algorithm is better about many criteria, then it could probably be suitable for similar optimization problems as well. The research for finding and developing new algorithms will be continued [9,68,109].

As noted, little research has been conducted on developing plant-inspired meta-heuristics. One of the phenomena related to humans and nature, in which the optimization concept can be seen, is the creation and maintenance of a fruit orchard. In this process, some seedlings are planted. The growth requirements of these seedlings, including light, water, and fertilizer, are provided in the following years. In the meantime, the weak seedlings are sometimes removed and replaced with new ones. Eliminating the strong trees' scion can also improve the weaker trees. Over some years, the process leads to an orchard with stronger and more fruitful trees. The general concept of this phenomenon can be used as a basis for developing a new meta-heuristic algorithm for these reasons: There is both individual/personal behavior (self-driven growth of the tree) and social behavior (grafting); There is both random search and the concentrated search around the previous solutions (growth of trees); Finally, it includes both elitism (choosing the strong trees) and using of such good solutions to improve the weak ones (grafting strong trees to the weaker ones). This paper presents the orchard algorithm (OA) inspired by a fruit garden's planting, maintenance, and development. To introduce the basics of the algorithm, we describe the creation and development of an orchard in reality in Section 2. In Section 2, the developed algorithm and its steps are also explained. Section 3 presents the results of implementing the algorithm to some benchmark functions and engineering optimization problems. In Section 4, a summary of the results and conclusions are provided. **Paper contributions**

- This paper proposes a new meta-heuristic algorithm called the orchard algorithm (OA) to solve discrete and continuous optimization problems.
- By introducing various operators such as annual growth, screening, and grafting, the algorithm can efficiently search and explore the search space. In OA, both explorations of the search space and exploitation of the best solutions are achieved using personal and social behavior.

Table 1
General information of some meta-heuristic algorithms.

Developers and references	Algorithm's name and abbreviation	Year
Hooke and Jeeves [53]	Pattern Search (PS)	1961
Rastrigin [105]	Random Search (RS)	1963
Matyas [80]	Random Optimization	1965
Fogel et al. [32]	Evolutionary strategy (ES)	1966
Hastings [47]	Metropolis–Hastings algorithm	1970
Kernighan and Lin [70]	Graph Partitioning Method	1970
Holland [52]	Genetic Algorithm (GA)	1975
Glover [39]	Scatter Search	1977
Kirkpatrick et al. [73]	Simulated Annealing (SA)	1983
Glover [40]	Tabu Search (TS)	1986
Farmer et al. [31]	Artificial Immune System (AIS)	1986
Moscato [90]	Memetic Algorithm	1989
Koza [74]	Genetic Programming (GP)	1992
Srinivas and Deb [121]	NSGA for Multi-Objective Optimization	1994
Eberhart and Kennedy [26]	Particle Swarm Optimization (PSO)	1995
Dorigo et al. [24]	Ant Colony Optimization (ACO)	1996
Storn and Price [122]	Differential Evolution (DE)	1997
Rubinstein [108]	Cross Entropy Method (CEM)	1997
Mladenovic and Hansen [88]	Variable Neighborhood search (VNS)	1997
Hansen and Ostermeier [44]	CMA-ES	2001
Geem et al. [36]	Harmony Search (HS)	2001
Hanseth and Aanestad [45]	Bootstrap Algorithm (BA)	2001
Larranaga and Lozano [77]	Estimation of Distribution Algorithms (EDA)	2001
Nakrani and Tovey [95]	Bees Optimization	2004
Pham et al. [99]	Bees Algorithms (BA)	2005
Karaboga [57]	Artificial Bee Colony Algorithm (ABC)	2005
Krishnanand and Ghose [75]	Glowworm Swarm Optimization (GSO)	2006
Haddad et al. [43]	Honey-bee Mating Optimization (HMO)	2006
Mucherino and Seref [91]	Monkey Search (MS)	2007
Atashpaz-Gargari and Lucas [8]	Imperialist competitive algorithm (ICA)	2007
Wierstra et al. [130]	Natural Evolution Strategies	2008
Simon [120]	Biogeography-based optimization (BBO)	2008
Teodorović [126]	Bee Colony Optimization (BCO)	2009
He et al. [50]	Group Search Optimizer (GSO)	2009
Yang and Deb [134]	Cuckoo Search (CS)	2009
Rashedi et al. [104]	Gravitational Search Algorithm (GSA)	2009
Kashan [59]	League Championship Algorithm (LCA)	2009
Kadioglu and Sellmann [55]	Dialectic Search	2009
Shah-Hosseini [117]	Intelligent Water Drops (IWD)	2009
Yang [131]	Firefly Algorithm (FA)	2009
Battiti and Brunato [11]	Reactive Search Optimization (RSO)	2010
Yang [132]	Bat Algorithm (BA)	2010
Shah-Hosseini [118]	Galaxy-based Search Algorithm (GbSA)	2011
Tamura and Yasuda [125]	Spiral Optimization (SO)	2011
Alsheddy [7]	Guided Local Search (GLS)	2011
Rajabioun [103]	Cuckoo Optimization Algorithm (COA)	2011
Gandomi and Alavi [34]	Krill Herd (KH) Algorithm	2012
Civicioglu [22]	Differential Search Algorithm (DS)	2012
Sadollah et al. [110]	Mine Blast Algorithm (MBA)	2013
Hatamlou [48]	Black Hole (BH)	2013
Gandomi [33]	Interior Search Algorithm (ISA)	2014
Cheng and Prayogo [19]	Symbiotic Organisms Search (SOS)	2014
Kashan [60]	Optics Inspired Optimization (OIO)	2015

(*continued on next page*)

Table 1 (*continued*).

Developers and references	Algorithm's name and abbreviation	Year
Kaveh and Mahdavi [65]	Colliding Bodies Optimization (CBO)	2015
Salimi [116]	Stochastic Fractal Search (SFS)	2015
Zheng [138]	Water Wave Optimization (WWO)	2015
Dogan and olmez [25]	Vortex Search Algorithm (VSA)	2015
Wang et al. [128]	Elephant Herding Optimization (EHO)	2015
Kashan et al. [61]	Grouping Evolution Strategies (GES)	2015
Mirjalili [85]	Dragonfly algorithm	2016
Liang et al. [79]	Virus Optimization Algorithm (VOA)	2016
Mirjalili [86]	Sine Cosine Algorithm (SCA)	2016
Ebrahimi and Khamehchi [27]	Sperm Whale Algorithm (SWA)	2016
Mirjalili et al. [87]	Salp Swarm Algorithm (SSA)	2017
Baykasoğlu and Akpinar [12]	Weighted Superposition Attraction (WSA)	2017
Mortazavi et al. [89]	Interactive Search Algorithm (ISA)	2018
Heidari et al. [51]	Harris Hawks Optimization (HHO)	2019
Yapici and Cetinkaya [135]	Pathfinder Algorithm (PFA)	2019
Kaur et al. [62]	Tunicate Swarm Algorithm (TSA)	2020
Hayyolalam and Kazem [49]	Black Widow Optimization (BWO)	2020
Khishe and Mosavi [71]	Chimp Optimization Algorithm (ChOA)	2020
Braik et al. [14]	Capuchin Search Algorithm (CapSA)	2021
Talatahari et al. [124]	Crystal Structure Algorithm (CryStAl)	2021
Pan et al. [97]	Gannet Optimization Algorithm (GOA)	2022
Eslami et al. [29]	Aphid–Ant Mutualism (AAM)	2022
Hashim et al. [46]	Honey Badger Algorithm (HBA)	2022

Table 2
General information of some meta-heuristic algorithms inspired by the life and behavior of plants.

Developers and references	Algorithm's name and abbreviation	Year
Murase [93]	Photosynthetic Algorithm (PA)	2000
Mehrabian and Lucas [81]	Invasive Weed Optimization (IWO)	2006
Karci [58]	Sapling Growing up Algorithm (SGuA)	2007
Cai et al. [15]	Plant Growth Optimization (PGO)	2008
Wang et al. [127]	Plant Growth Simulation Algorithm (PGSA)	2008
Premaratne et al. [100]	Paddy Field Algorithm (PFA)	2009
Zhao et al. [137]	Artificial Plant Optimization Algorithm (APOA)	2011
Salhi and Fraga [115]	Plant Propagation Algorithm (PPA)	2011
Yang [133]	Flower Pollination Algorithm (FPA)	2012
Qi et al. [102]	Root Mass Optimization (RMO)	2013
Zhang et al. [136]	Root Growth Algorithm (RGA)	2014
Merrikh-Bayat [82]	Runner Root Algorithm (RRA)	2015
Labbi et al. [76]	Rooted Tree Optimization (RTO)	2016
Zhou et al. [140]	Path Planning Algorithm based on plant growth (PGPP)	2017
Li et al. [78]	Artificial tree (AT)	2017
Cheraghalipour et al. [20]	Tree Growth Algorithm (TGA)	2018
Alimoradi et al. [6]	Trees Social Relations Optimization Algorithm (TSR)	2022

- The performance of the proposed OA is evaluated on several optimization problems in comparison with 13 widely-used and competitive algorithms: (a) CEC2005 test functions, (b) IEEE CEC06-2019 test functions, (c) Thirty unimodal, multimodal, and composite functions, (d) Real engineering optimization problems.
- For validation, a parametric and non-parametric statistical analysis such as Wilcoxon's test, standard deviation, the mean value of the fitness function, convergence curve, and run time of algorithms are used.

2. The orchard algorithm (OA)

This section explains the theoretical basis and main steps of the OA. The exploration and exploitation abilities of the algorithm will also be discussed.

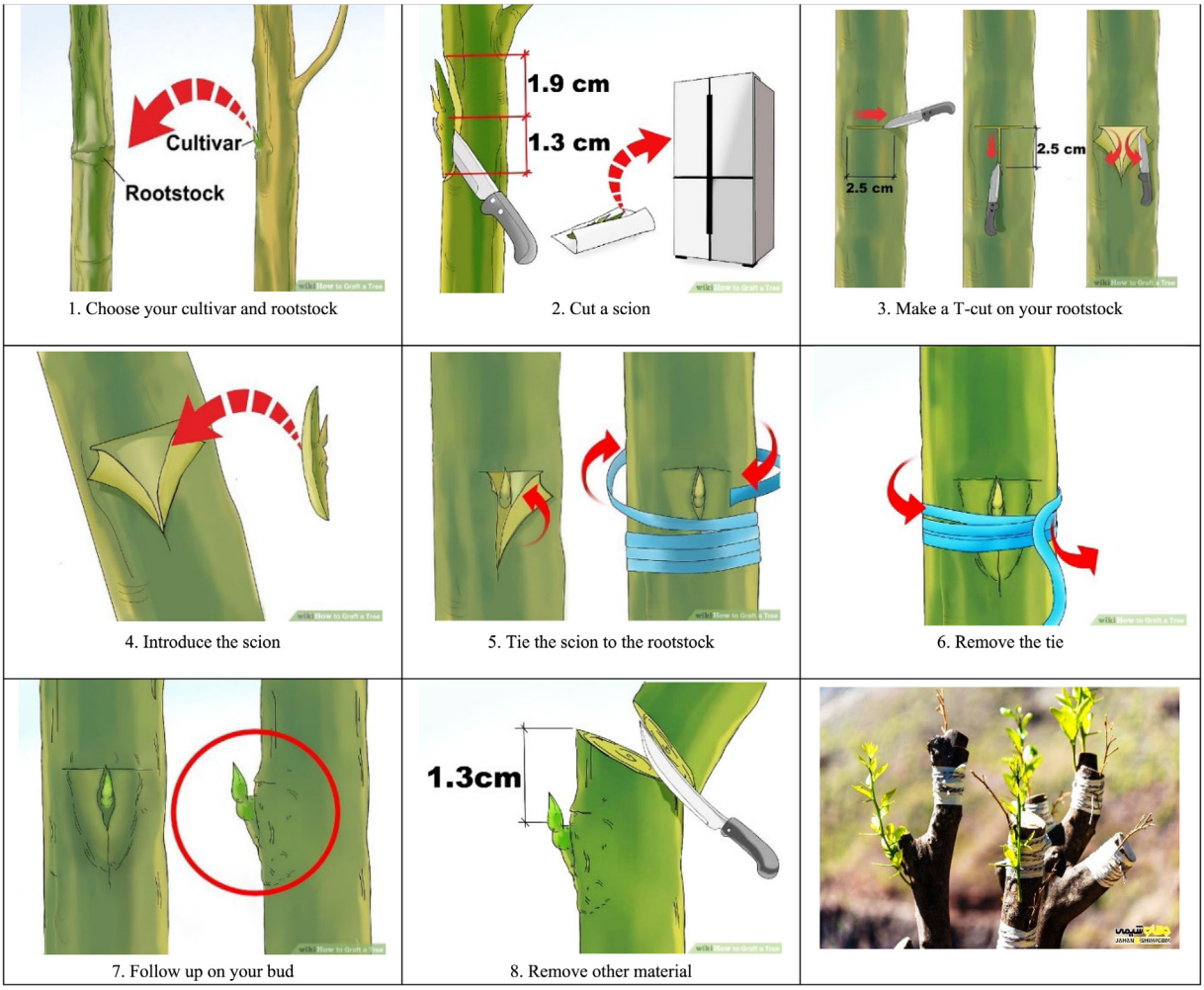


Fig. 1. A simple illustration of tree grafting [16].

2.1. Creation, maintenance, and improvement of an orchard

The main stages related to the creation, maintenance, and improvement of an orchard are [17,18,96,98]:

- Examination of environmental factors such as temperature, light, soil, water, and location
- Study of economic factors
- Selection of cultivars and preparation of seedlings
- Selecting the land, preparing the land, and planting the seedlings
- Maintenance of the orchard until fruiting (including irrigation, fertilization, annual plowing between rows, and control of pests, diseases, weeds, etc.)
- Orchard improvement and renovation (including trimming, density adjustment and spacing between trees, removal of unsuitable trees, and grafting)

One of the most useful operations mentioned in the above list is grafting trees. During grafting, a part of a plant is cut and attached to a section of another plant in such a way that the two can fuse after a while. The plant resulting from their union can grow independently [42,92]. A simple illustration of a common tree grafting method is provided in Fig. 1.

Fig. 2. An example of tree grafted in an orchard.

The important benefits of grafting, related to this research, are using the benefits of both rootstock and scion, repairing the damaged parts, and rejuvenating old trees [35,54,92]. Fig. 2 also shows an example of a tree grafted in an orchard. Creating, maintaining, and improving an orchard is a set of activities to optimize all trees. A new meta-heuristic algorithm is developed by defining some algorithmic operators inspired by those activities. This new algorithm is described in the following section.

2.2. The OA algorithm steps

As described in the previous section, the fruit gardening process inspires the developed OA algorithm. The main steps of the OA algorithm are as follows:

1. Creation of an orchard by planting some seedlings
2. Growth of the seedlings
3. Screening of the seedlings
4. Grafting
5. Replacing of weak seedlings by new ones
6. Elitism
7. Checking the stop condition

The procedure of the OA algorithm is sown in Fig. 3. In the following, the modeling of each step is described.

2.2.1. The creation of an orchard

Each tree is defined as a solution to the optimization problem. Some new seedlings need to be planted to create an orchard; i.e., some random solutions should be constructed to create an orchard. The number of initially planted seedlings is considered one of the problem's parameters.

2.2.2. Seedling growth

After planting, by providing the right conditions, each seedling grows. This growth continues for several years. Every few years, the screening and other garden maintenance activities (the next steps of the algorithm) are performed. Each year growth of the seedling is represented by a local search in the neighborhood of the solution. If the result of a local search is better than the solution, it will replace the solution. The number of subsequent local

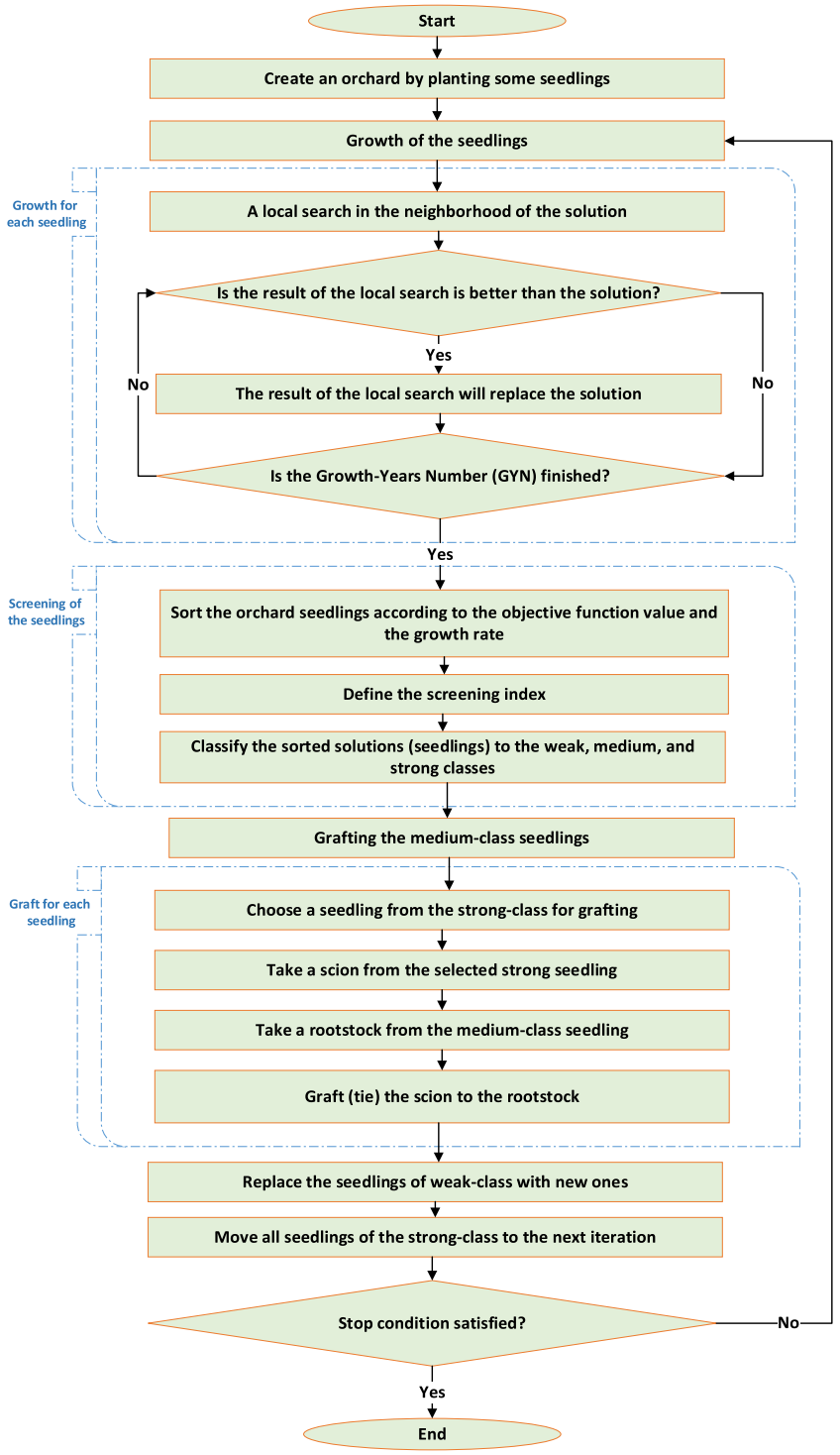


Fig. 3. The procedure of the OA algorithm.

searches is called the growth-years number (GYN). It is the number of years we want the seedlings to grow before the screening stage. The GYN (m) is also defined as an algorithm parameter.

2.2.3. Screening

After each seedling has been grown (improved by local search) for several years, screening is performed, in which the weak, medium, and strong seedlings (solutions) are identified. For this purpose, it is necessary to sort and rank the garden seedlings according to the optimality index of the solutions. The ranking determines whether a solution will be transferred without change as an elite, replaced by a new one, or selected as a scion or stem for grafting.

In OA, the quality of each solution is evaluated by two criteria. The first, which is consistent with other meta-heuristics, is the objective function value. The second is based on the growth rate of the solution, which is inspired by the seedling selection procedure in the orchard. This aspect is one of the major differences between OA and other optimization algorithms. The solution that has been steadily improving for the past few years (iterations of the growth operator) should be appreciated more. In other words, solutions with good objective function values and growth rates can drive the algorithm faster toward the global optimum. The solutions are evaluated and ranked according to the optimality index defined by Eq. (1).

$$F_j = \alpha \dot{f}_j + \beta \dot{g}_j \tag{1}$$

In Eq. (1), $dotf_j$ and $dotg_j$ are the normalized objective function value and the normalized growth rate of solution j, and α and β are two constant coefficients that determine the relative importance of the objective function and growth rate. The sum of these two must be equal to one. They are also regarded as parameters of the OA. The objective function and the growth rate are of the same value and data type, but the amount of the objective function is much higher than the growth rate. When normalized, their importance and role in solution ranking are defined merely by α and β coefficients. The objective function values are normalized using Eq. (2) where n is the total number of seedlings (solutions).

$$\dot{f}_j = \frac{f_j}{max(f_j) \quad 1 \le j \le n} \tag{2}$$

The growth rate of each solution is calculated using Eq. (2).

$$g_j = \Sigma_{i=m-l}^{m} \lambda_i (f_j^i - f_j^{i-1}) \tag{3}$$

Here, i is the growth year, m is the total number of growth years before the screening, l is the number of growth years before screening for which the growth rate is considered, and λ is the weight given to those years. As described in the previous section, each seedling grows for m year. Here, the growth rate only in the last l years is considered. The growth rates in the latter iterations before screening are more important than the previous ones. Because they represent the more recent states of the tree, in other words, a solution with high growth rates in initial years but lower growth rates in final years is not much promising, and vice versa. In the implementation of this study, as discussed in the following sections, only the last three years before screening are considered with weights of 0.6, 0.3, and 0.1, respectively. Therefore, Eq. (3) will be simplified into Eq. (4) These weights are defined by the trial and error method.

$$g_j = 0.6(f_j^i - f_j^{i-1}) + 0.3(f_j^{i-1} - f_j^{i-2}) + 0.1(f_j^{i-2} - f_j^{i-3}) \tag{4}$$

Finally, the calculated growth rate also needs to be normalized. Eq. (5) shows the normalized growth rate.

$$\dot{g}_j = \frac{g_j}{max(g_j) \quad 1 \le j \le n} \tag{5}$$

After sorting the solutions, the screening index should be defined, and the sorted solutions should be classified accordingly. The screening index can be defined as the thresholds between the classes (e.g., solutions with optimality indexes lower than a specified value fall into the weak class). The screening index can also be defined as the population percentages (i.e., a certain percentage of the solutions fall into the weak, medium, and strong classes). Here, for the sake of simplicity, the latter method is used.

2.2.4. Grafting

Only the medium-class solutions (seedlings), expected to improve, are grafted with a portion (scions) of the strong solutions. The chance of choosing any strong solution for grafting depends on its optimality index (Eq. (1)). The Roulette wheel method can be used for such a selection. Then, a part of the medium solution is replaced by the corresponding part from the strong solution, provided that the obtained solution is correct. Otherwise, the grafted solution needs to be modified.

2.2.5. Replacement of the weak seedlings by the new ones

The weak solutions are replaced with new random ones. Because after a few years of growing and searching for different neighborhoods in these years of growth, they have not recovered and remain in the weak category. In reality of the orchard, the weak seedlings are usually considered to have a genetic or other serious problem and are replaced with new seedlings. Because after a few years of growth, they have not improved. Therefore, grafting is not regarded as a resolution for them. As a result, inspired by the gardener's attempt to plant new solutions instead of these, the algorithm eliminates weak batch responses and replaces new random ones.

2.2.6. Elitism

In nature, trees that grow well do not need grafting. Similarly, in the OA, a solution of the strong class is moved to the next iteration without any change. Such a solution is appreciated even more when a part of it, as a stem, is used to improve a solution of the medium class.

2.2.7. Checking the stop condition

An algorithm iteration is performed by implementing the operators defined in the previous sections. In any such iteration, all the strong solutions are kept, stems from the strong ones graft the medium solutions, and the weak solutions are replaced by randomly created new ones. Next, the stopping condition is checked. If not met, the algorithm is repeated. The stopping condition can be defined in various ways, such as achieving a specific number of iterations, achieving a specific accuracy, reaching a specific time limit, etc. Most seedlings have grown into adequate trees when the stopping condition is met. Finally, the most fertile tree (the best solution) is introduced as the algorithmic result.

2.3. How the OA searches the solution space

Although the meta-heuristic algorithms fall into the class of random algorithms, their operators are defined in such a way as to guide the solutions to the final optimum. In other words, although the operators of the algorithms behave randomly, they ensure a gradual movement toward the optimum solutions. The search (solution) space in optimization problems is usually very large, requiring appropriate operators for a comprehensive yet effective search. A meta-heuristic algorithm should be able to explore new areas of the solution space continuously and randomly; this activity is called exploration. At the same time, the close neighborhood of the already-found promising solutions should be examined more precisely; this activity is called exploitation. The effectiveness and efficiency of a meta-heuristic algorithm largely depend on defining proper operators for exploitation and exploration and on keeping a balance between these two. This section will examine how exploitation and exploration activities are carried out in the OA algorithm.

In OA, the seedling growth is a straight exploitation operator carried out on all solutions. In each iteration of this operator, the neighborhood of each solution is examined using a simple local search. If a better solution is found, it replaces the base solution. The most straightforward form of exploration happens when the weak seedlings are replaced with new randomly-generated solutions. In addition, grafting is an operator that can be considered for exploration and exploitation. In the early iterations of the algorithm, when the solutions are scattered, this operator mostly results in the exploration of new areas. However, in the later iterations, when some level of convergence is achieved, and solutions are similar in many of their elements, the grafting results in the exploitation of the best solutions. Furthermore, in the screening process, the solutions with higher optimality functions and higher growth rates are selected for exploitation in the subsequent operators. The pseudo-codes of the orchard algorithm (OA) are shown in Algorithm 1. Fig. 4 also shows an example of modeling the OA operator in a binary problem. The RMSE value equals the fitness function, and the lower the RMSE, the better the solution.

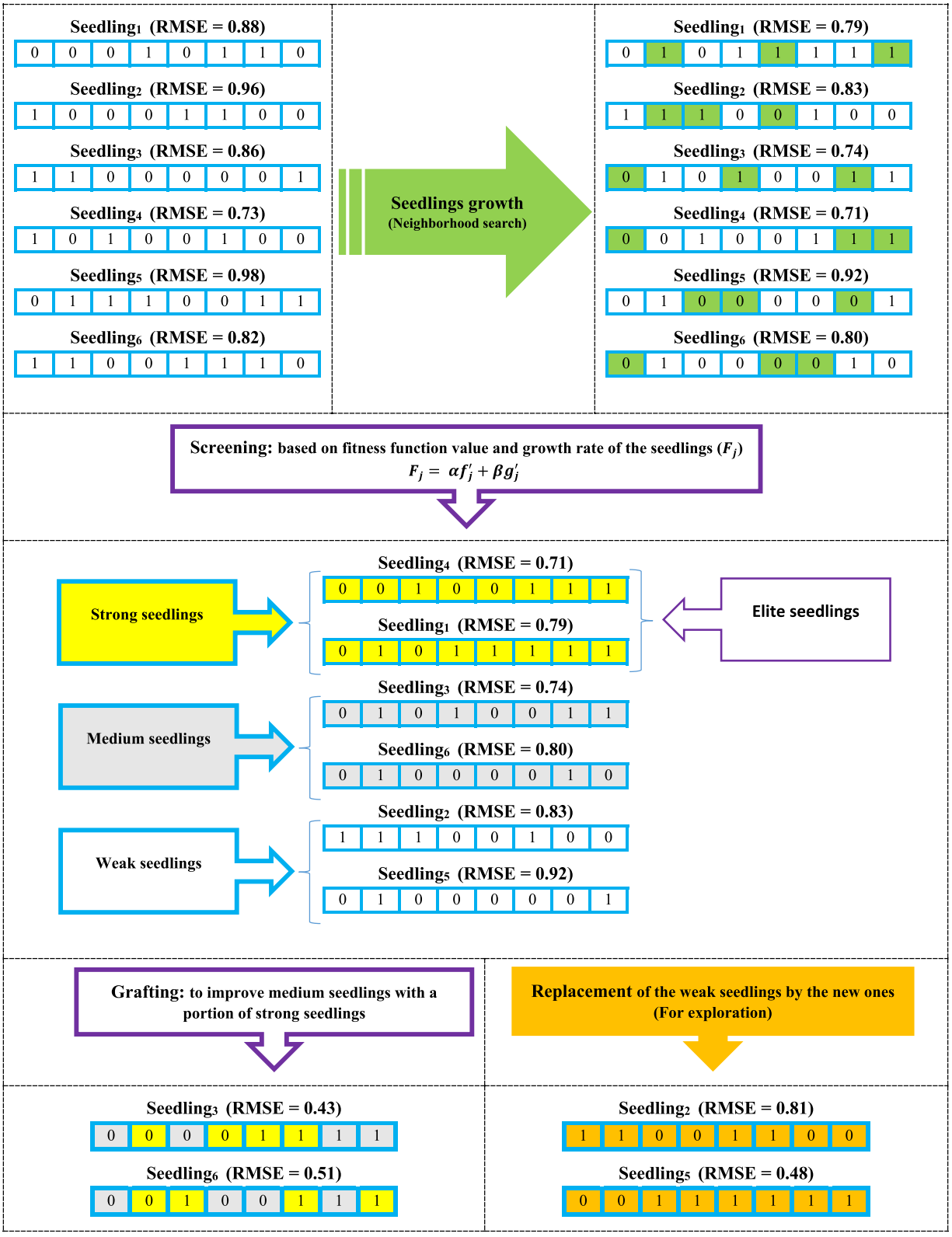


Fig. 4. An example of modeling the OA operator in a binary problem.

Table 3
The main parameters of OA for calibration.

No.	Variable	Description
1	Max_iter	Iteration number
2	N_pop	Initial population
3	N_high	No. of strong trees in screening
4	N_low	No. of weak trees in screening
5	N_trans	No. of trees that need grafting
6	α	The weight of fitness function
7	β, $(\beta = 1 - \alpha)$	The weight of growth rate

Algorithm 1 Pseudo-codes of orchard algorithm (OA)

```
%% Seedlings initialization;
for i = 1 to N do
        Create orchard by planting some seedlings;
        Set number of strong trees in screening;
        Set number of weak trees in screening;
        Set number of trees that need grafting;
        Set the growth-years number (GYN);
        Set α (The weight of fitness function);
        Set β (The weight of growth rate);
        Evaluate the fitness of seedlings;
 end
 %% Seedlings growth through the search space;
While (termination criteria is not met) do
        for i = 1 to N do
                Growth of the seedlings;
                Screening based on fitness function and growth rate (Eq. (1));
                Grafting;
                Replacement of the weak seedlings by the new ones;
                Elitism;
        end
end while
```

3. Experimental results

This section has been divided into five subsections. In the first subsection, the parameters of the OA and other algorithms are calibrated. In the second subsection, the performance of the proposed OA on CEC2005 test functions [123] is evaluated in comparison with some widely-used and competitive algorithms, including particle swarm optimization (PSO), genetic algorithm (GA), capuchin search algorithm (CapSA) [14], chimp optimization algorithm (ChOA) [71], and black widow optimization (BWO) [49]. In the third subsection, the performance of the proposed algorithms on IEEE CEC06-2019 test functions [56] is evaluated. The fourth subsection evaluates the algorithm's ability to solve 30 benchmark functions, and its results are compared with other algorithms. These 30 functions are provided by [5,12,20,21,23], and the results of OA are compared with Cheraghalipour et al. [20]. In the last subsection, four real-world engineering problems are used to evaluate the performance of the proposed algorithm. The results are obtained by implementing it on a personal computer with specifications Core (TM) i7-5500 U2 2.40 GHz.

3.1. Part 1: Calibration of OA parameters

All meta-heuristics have several parameters that should be calibrated. Therefore, the best combination these parameters must be determined before evaluating the algorithm. The parameters of OA, are listed in Table 3.

In this paper, calibration of OA parameters is set by trial and error method. Since the optimal solution for the test functions is recognized, one can obtain the best combination of parameters. For finding the best value for each

Table 4
Calibration of the OA parameters using trial and error method.

No. Run	Parameter					Average of fitness function values in 20 runs of algorithm			
	N_high	N_low	N_trans	α	β	BL	BF1	EP	GP
1	30	100	70	0.7	0.3	1.12E−10	0	0.99998	3.000124
2	50	80	70	0.7	0.3	2.22E−18	4.13E−12	−1	3
3	**60**	**80**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
4	80	60	60	0.7	0.3	1.05E−17	0	−1	3
5	100	40	60	0.7	0.3	5.35E−07	1.18E−09	0.99998	3.000012
6	**80**	**60**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
7	**60**	**80**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
8	60	100	40	0.7	0.3	0	0	0.99999	3
9	80	90	30	0.7	0.3	5.32E−19	0	−1	3.000750
10	**70**	**80**	**50**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
11	**60**	**80**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
12	60	60	80	0.7	0.3	0	0	0.99999	3
13	**60**	**80**	**60**	**0.4**	**0.6**	**0**	**0**	**−1**	**3**
14	**60**	**80**	**60**	**0.6**	**0.4**	**0**	**0**	**−1**	**3**
15	**60**	**80**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
16	60	80	60	0.9	0.1	3.41E−15	3.41E−11	0.99999	3.000027
17	**60**	**80**	**60**	**0.8**	**0.2**	**0**	**0**	**−1**	**3**
18	**60**	**80**	**60**	**0.7**	**0.3**	**0**	**0**	**−1**	**3**
19	**60**	**80**	**60**	**0.5**	**0.5**	**0**	**0**	**−1**	**3**
20	60	80	60	0.3	0.7	2.01E−06	0	0.99998	3

of the parameters, other parameters were kept fixed and the algorithm was implemented with different values of parameters. For test functions, the parameters of Max_iter and N_pop are set to 200 and 300, respectively. We have: $N\_high + N\_low + N\_trans = N\_pop$. Here, four test functions of BL, BF1, EP, and GP [5] were used to determine the optimal values of the parameters.

Table 4 evaluates different combinations of calibration parameters for each test function. For each setting of any test function, the table shows the average values of the objective function in 20 different runs of the algorithm. All the test functions are minimization problems. Therefore, the lower values of the fitness function mean better settings of the parameters. The best-found values for the OA (and other algorithms) parameters are presented in Table 5. The following sections use these optimal values for the other test functions and engineering optimization problems.

3.2. Part 2: Evaluation of OA by the CEC2005 test functions

In this part, 20 benchmark test functions introduced by the CEC2005 session were used to evaluate OA performance [123]. These test functions are rotated, shifted, expanded, and combined with conventional test functions that offer the highest complexity among conventional test functions. This benchmark problem is divided into unimodal, multimodal, and composite functions. Table 6 shows the details of the CEC2005 test functions. The 3D plot of these benchmark functions is shown in Fig. 5.

Tables 7–9 show the results of the proposed algorithms in the CEC2005 benchmark problems. The algorithms are run 25 times for each benchmark problem. Tables 7–9 report the mean value of the fitness function (Ave), the standard deviation of the fitness function (Std), and the non-parametric Wilcoxon's test (p-value). It should be noted that the best outcomes are reported in bold kind. Wilcoxon test was performed at a significance level of 5.00%. In this test, N/A stands for "Not Applicable", meaning that the OA algorithm cannot be statistically compared to itself. As can be seen, the results of the OA are better than the other algorithms.

Table 7 shows the results of unimodal CEC2005 test functions. In this table, the OA algorithm has reached the highest "Ave" value in 5 functions (100.00%). Table 8 shows the results of CEC2005 multimodal test functions. In this table, OA has reached the highest "Ave" value in 7 functions (77.77%). Table 9 also shows the results of CEC2005 composite test functions. In this table, OA has reached the highest "Ave" value in 4 functions (66.66%). Given the p-values of algorithms, OA can achieve significant results compared to other algorithms. Fig. 6 indicates

Table 5
Calibration of algorithm parameters by trial and error method.

Algorithm	Parameter	Value
OA	N_high	60
	N_low	60
	N_trans	80
	α	0.7
	β	0.3
	Population size	200
	Iteration	300
CapSA	Velocity control constants	1.00
	Inertia parameter	0.68
	Balance and elasticity factors	0.70, 9
	Population size	200
	Iteration	300
ChOA	a	[−1, 1]
	f	Linearly decreased from 2 to 0
	Population size	200
	Iteration	300
BWO	Procreate rate (PP)	0.62
	Mutation rate (PM)	0.23
	Cannibalism rate (CR)	0.46
	Population size	200
	Iteration	300
GA	Elitism percent	10%
	Mutation rate	0.08
	Crossover rate	0.92
	Population size	200
	Iteration	300
PSO	The inertial movement rate α	0.11
	The movement toward the best personal experience rate ($\Phi 1$)	0.65
	The movement toward the best global experience rate ($\Phi 2$)%	0.93
	Population size	200
	Iteration	300

the performance of the algorithms at the "Ave" metric. The ranking of the algorithms is OA, CapSA, ChOA, BWO, PSO, and GA, respectively.

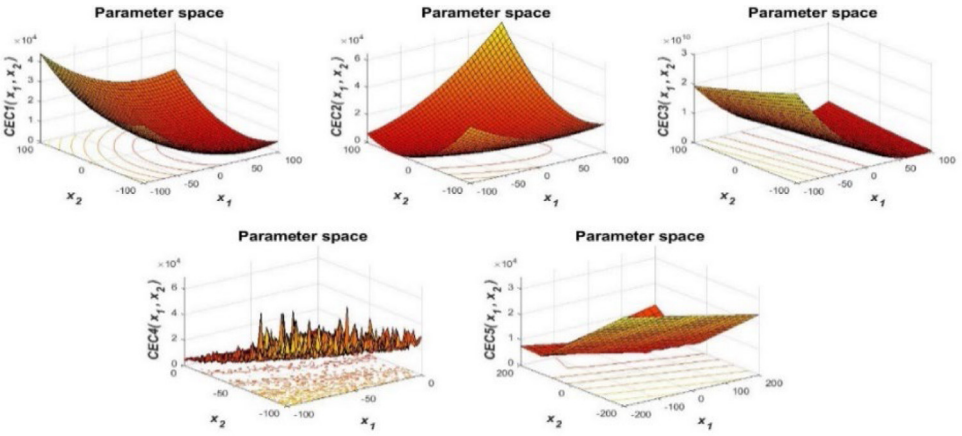
Figs. 7–9 show the convergence curve of algorithms on CEC2005 benchmark functions. As can be seen, the convergence curve of the OA is faster than other algorithms. OA achieved the best convergence curve in 16 test functions.

3.3. Part 3: Evaluation of OA by the CEC06-2019 test functions

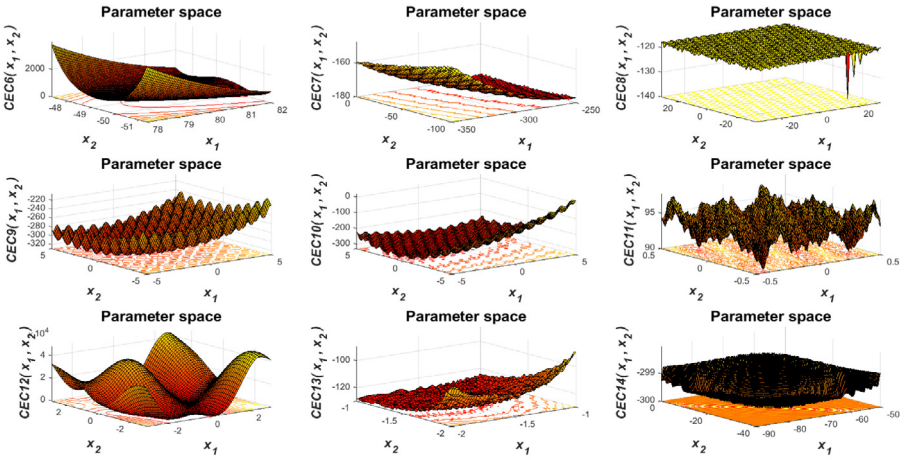
In this section, IEEE CEC06-2019 test functions are used to compare algorithms. Table 10 shows the details of IEEE CEC06-2019 (see [56] for more details). Fig. 10 shows a 3D plot of these test functions. Table 11 shows the results of algorithms on IEEE CEC06-2019 test functions. OA shows the best result in 9 test functions. CapSA and ChOA show the best result in 7 and 6 problems, respectively. GA got the worst results.

3.4. Part 4: Evaluation of OA with previous studies

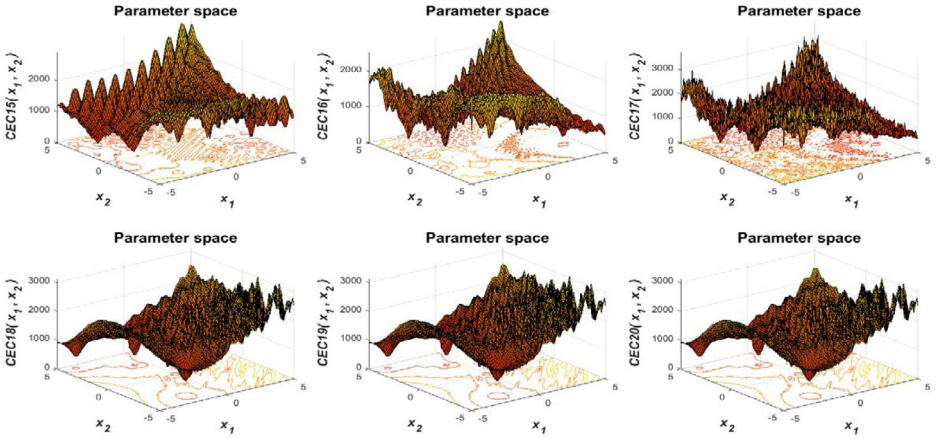
In this section, the performance of the OA is compared with eight other meta-algorithmic algorithms called best performance algorithm (BPA), iterative best performance algorithm (IBPA), largest absolute difference algorithm (LADA), tabu search (TS), simulated annealing (SA), weighted superposition attraction (WSA), covariance matrix adaptation evolution strategy (CMAES), and tree growth algorithm (TGA). The results of implementing these



a) Unimodal CEC2005 test functions



b) Multimodal CEC2005 test functions



c) Composite CEC2005 test functions

Fig. 5. The 3-D plots of CEC2005 benchmark functions.

Table 6
The details of the CEC2005 test functions.

No.	Test functions	Range	Dimension	F (x*)
	Unimodal functions			
F1	Shifted sphere function	[−100,100]	5	−450
F2	Shifted Schwefel's	[−100,100]	5	−450
F3	Shifted rotated high conditioned elliptic function (F4)	[−100,100]	5	−450
F4	Shifted Schwefel's with noise in fitness	[−100,100]	5	−450
F5	Schwefel's with global optimum on bounds	[−100,100]	5	−310
	Multimodal functions			
F6	Shifted Rosenbrock's function	[−100,100]	5	390
F7	Shifted rotated Griewank's function without bounds	[0,600]	5	−180
F8	Shifted rotated ackley's function with global optimum on bounds	[−32,32]	5	−140
F9	Shifted Rastrigin's function	[−5,5]	5	−330
F10	Shifted rotated Rastrigin's function	[−5,5]	5	−330
F11	Shifted rotated Weierstrass function	[−0.5,0.5]	5	90
F12	Schwefel's	[−100,100]	5	−460
F13	Expanded extended Griewank's plus Rosenbrock's function (F8F2)	[−3,1]	5	−130
F14	Shifted rotated expanded scaffer's F6	[−100,100]	5	−300
	Composite functions			
F15	Hybrid composition function	[−5,5]	10	120
F16	Rotated hybrid composition function	[−5,5]	10	120
F17	Rotated hybrid composition function with noise in fitness	[−5,5]	10	120
F18	Rotated hybrid composition function	[−5,5]	10	10
F19	Rotated hybrid composition function with a narrow basin for the global optimum	[−5,5]	10	10
F20	Rotated hybrid composition function with the global optimum on the bounds	[−5,5]	10	10

Table 7
The results of unimodal CEC2005 test functions.

Algorithm	Metric	CEC_{F1}	CEC_{F2}	CEC_{F3}	CEC_{F4}	CEC_{F5}
OA	Ave	**−439.256**	**−437.694**	**−440.985**	**−438.563**	**−309.895**
	Std	**4.3659**	**5.2685**	**2.2984**	**3.98541**	**0.1256**
	p-value	**N/A**	**N/A**	**N/A**	**N/A**	**N/A**
CapSA	Ave	−437.214	−436.836	−439.256	−429.196	−308.169
	Std	11.3658	15.3265	8.7463	36.9852	4.6532
	p-value	0.0039	0.0048	0.0009	0.1896	0.0005
ChOA	Ave	−435.198	−434.198	−415.647	−435.823	−308.865
	Std	10.2985	18.2147	32.3654	22.8962	2.2698
	p-value	0.0852	0.0085	0.8265	0.0652	0.0002
BWO	Ave	−434.95	−432.286	−422.698	−431.816	−302.328
	Std	15.2365	20.9614	11.3981	14.6341	50.7562
	p-value	0.1269	0.0453	0.0965	0.1526	0.0651
PSO	Ave	−430.265	−426.258	−409.453	−415.826	−298.84
	Std	30.2479	45.2985	41.8361	43.17963	60.2562
	p-value	0.6521	0.8632	0.9265	0.6532	0.9624
GA	Ave	−431.813	−424.952	−411.274	−417.836	−296.192
	Std	36.9074	48.7365	50.8412	49.2198	62.5132
	p-value	0.6929	0.9658	.09987	0.5179	0.9841

Table 8
The results of CEC2005 multimodal test functions.

Algorithm	Metric	CEC_{F6}	CEC_{F7}	CEC_{F8}	CEC_{F9}	CEC_{F10}	CEC_{F11}	CEC_{F12}	CEC_{F13}	CEC_{F14}
OA	Ave	398.238	−176.532	−129.856	−315.985	−316.256	91.256	−459.625	−129.256	−278.951
	Std	4.2496	0.5231	1.2125	16.2546	0.9654	0.1456	0.0025	0.0874	3.6321
	p-value	N/A	N/A	N/A	0.0019	N/A	N/A	N/A	N/A	0.0027
CapSA	Ave	399.564	−173.254	−127.256	−316.354	−312.856	93.785	−459.126	−127.256	−275.693
	Std	12.6412	18.3263	18.9856	10.1963	20.1463	5.6354	0.0754	3.5412	8.0186
	p-value	0.0028	0.0089	0.0042	N/A	0.0049	0.0029	0.0009	0.0049	0.0269
ChOA	Ave	401.208	−174.456	−126.741	−313.745	−313.745	92.126	−459.468	−128.746	−279.563
	Std	26.8745	15.1963	26.4632	25.7456	18.7456	4.7456	0.0254	1.1856	0.2563
	p-value	0.0047	0.0089	0.0083	0.0185	0.0024	0.0016	0.0009	0.0019	N/A
BWO	Ave	400.843	−170.126	−123.963	−310.189	−308.745	93.965	−458.974	−125.189	−276.475
	Std	18.6351	35.7456	32.7416	30.7456	35.7452	8.4123	1.8426	10.1896	5.1706
	p-value	0.0047	0.2365	0.0175	0.1852	0.1426	0.0146	0.0089	0.0269	0.0147
PSO	Ave	409.761	−169.756	−121.186	−307.175	−301.106	95.789	−456.863	−126.863	−271.756
	Std	44.9854	42.1286	41.1875	51.8561	65.8421	20.7561	15.7456	5.4123	35.7963
	p-value	0.1631	0.3456	0.1826	0.3258	0.6523	0.2384	0.2156	0.0096	0.4286
GA	Ave	407.516	−167.256	−122.416	−305.862	−302.147	96.475	−453.745	−122.863	−270.475
	Std	40.5367	48.1036	39.7521	54.1286	61.0756	25.7463	21.7456	19.0863	41.1963
	p-value	0.1631	0.3806	0.1619	0.3619	0.5213	0.3654	0.3571	0.4189	0.4429

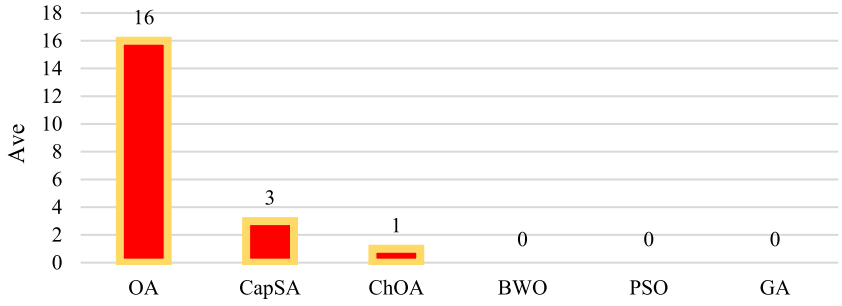


Fig. 6. The ranking of algorithms at (Ave) metric on CEC2005 problems.

algorithms on all 30 test functions are reported in Cheraghalipour et al. [20]. Here, the OA algorithm is implemented 30 times for each test function. The fitness function values smaller than 1E-15 are regarded as zero. The statistical results related to 30 separate executions of the OA and other algorithms on the 30 test functions are presented in Table 12.

In this table, the best value of the fitness function (BestF(x)), the mean value of the fitness function (AvgF(x)), the standard deviation of the fitness function (SDF(x)), and the average runtime of the meta-heuristic algorithms (AvgRuntime) in 30 executions are reported. As presented in Table 12, the results of the OA are often better than the other meta-heuristics. The algorithms in order of better results were the OA, TGA, CMAES, and WSA. According to Table 12, the OA had the best value of "BestF(x)" in 22 out of 33 cases and the best value of "AvgF(x)" in 17 cases, which means the success rates of 66.66 and 51.51%, respectively. The OA also achieved the lowest "AvgRuntime" value in 24 out of 33 cases, representing a success rate of 72.72%. Although the OA has not arrived at the lowest value of "BestF(x)" in 11 cases, still in 8 of these cases, its values were reasonable and near to optimal. The OA did not produce good results in only three test functions (RG, RB, and SWF), which covers 9.1% of the criteria set.

The SDF(x) values presented in Table 12 show that OA has the highest level of repeatability compared to the other 8 algorithms. In fact, in most of the test functions, the OA produced results with the least amount of standard deviation. Therefore, it can be concluded that the OA is a very stable search method.

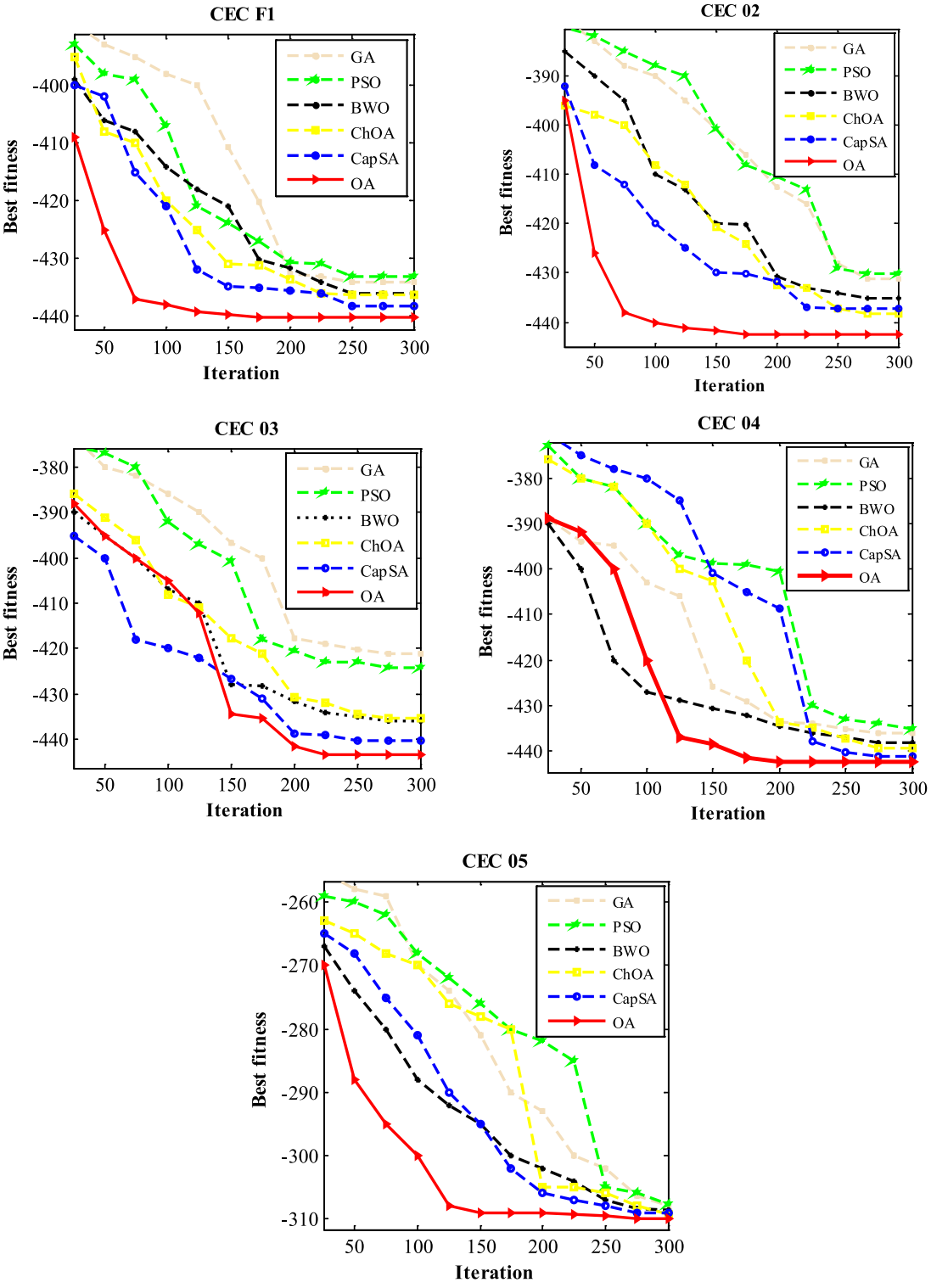


Fig. 7. The convergence curve of algorithms on unimodal CEC2005 test functions.

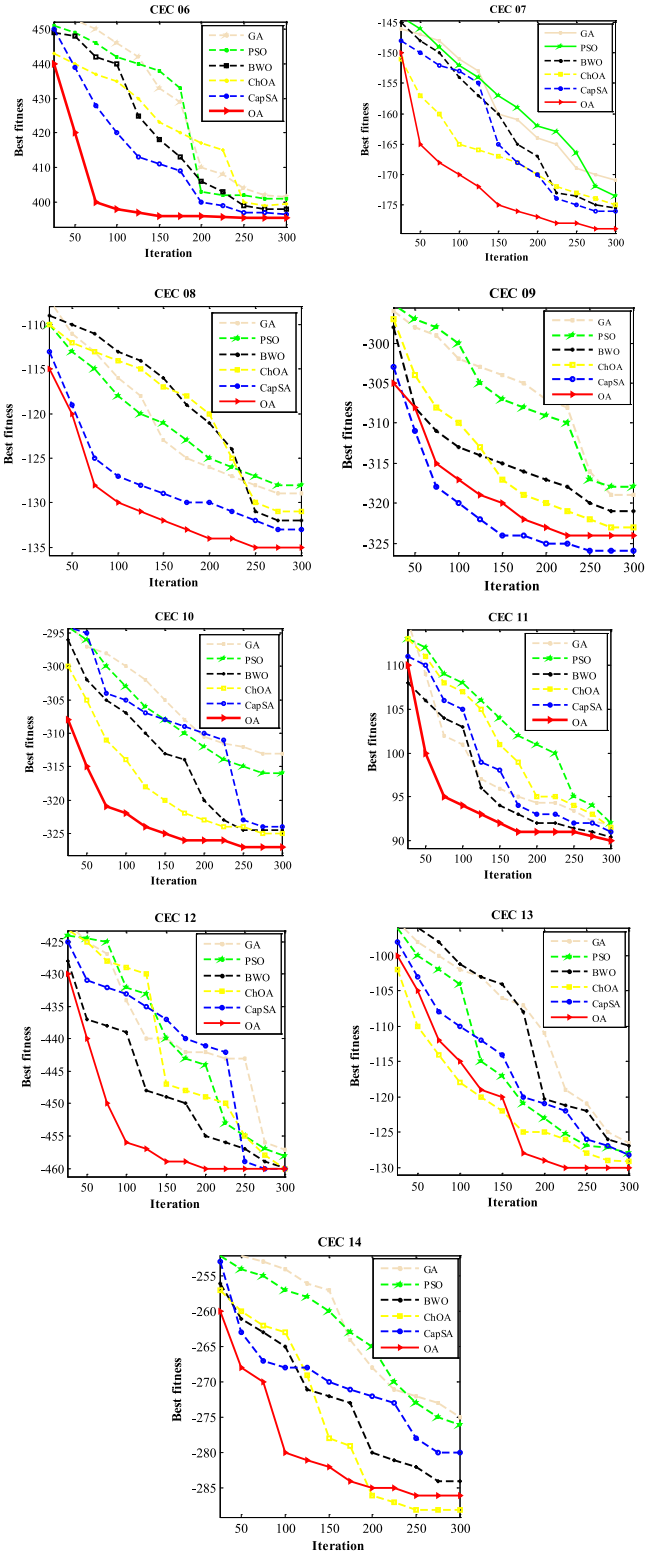


Fig. 8. The convergence curve of algorithms on multimodal CEC2005 test functions.

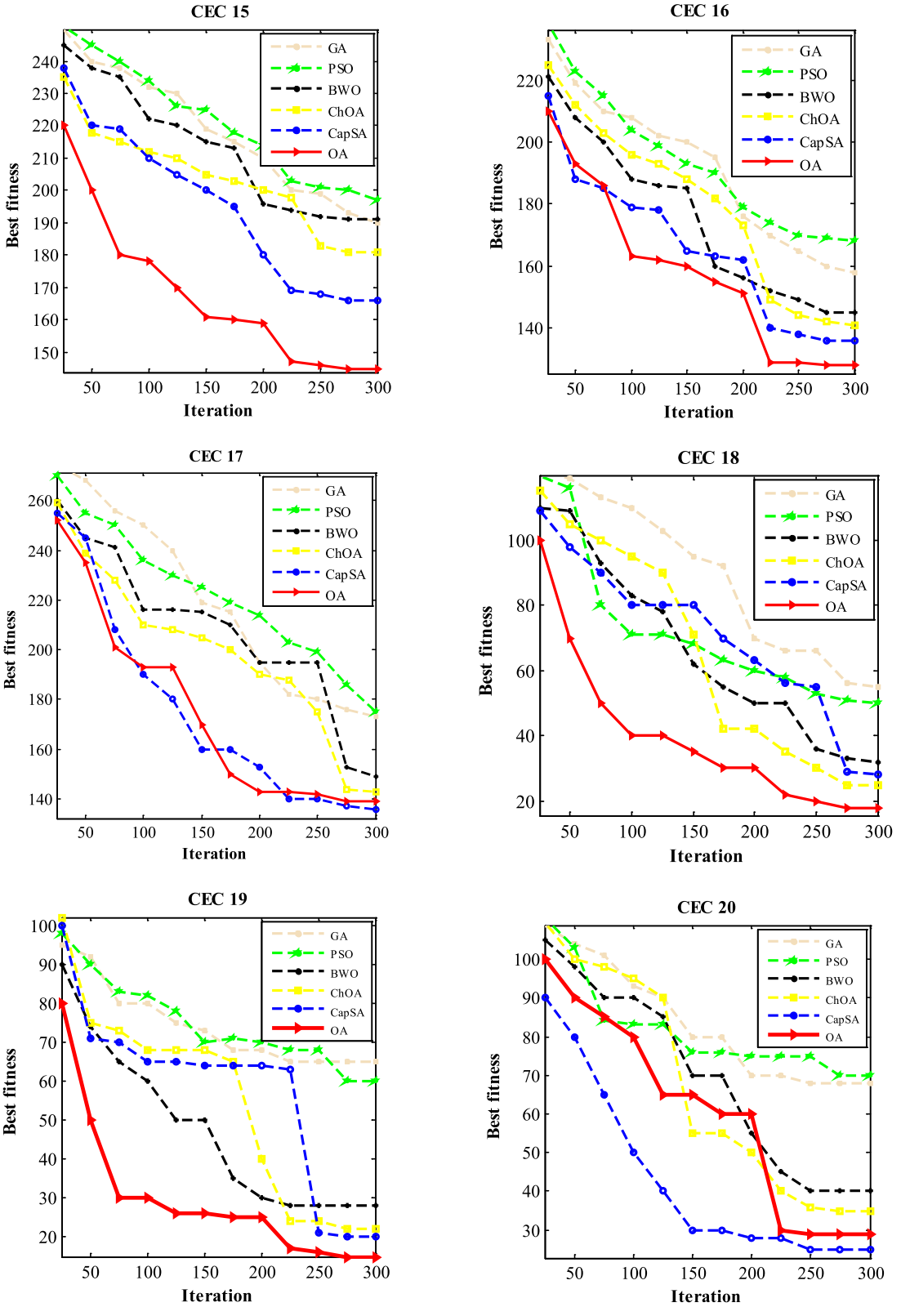


Fig. 9. The convergence curve of algorithms on composite CEC2005 test functions.

Table 9
The results of CEC2005 multimodal test functions.

Algorithm	Metric	CEC_{F15}	CEC_{F16}	CEC_{F17}	CEC_{F18}	CEC_{F19}	CEC_{F20}
OA	Ave	174.325	168.5621	211.574	50.245	45.325	65.796
	Std	30.2561	35.1475	50.2563	20.2863	15.7456	26.2145
	p-value	N/A	N/A	0.0086	N/A	N/A	0.0019
CapSA	Ave	181.562	175.256	210.256	55.635	48.896	62.5261
	Std	50.1452	41.2597	45.2865	30.5231	23.7456	18.2563
	p-value	0.0098	0.0076	N/A	0.0086	0.0075	N/A
ChOA	Ave	202.459	173.826	215.9641	58.412	47.159	68.146
	Std	61.8563	36.1863	55.3652	33.745	21.7632	36.8516
	p-value	0.0425	0.0053	0.0196	0.0145	0.0043	0.0079
BWO	Ave	210.763	180.756	225.736	62.107	52.763	79.153
	Std	65.7856	56.8632	60.7463	40.1496	29.7453	49.1365
	p-value	0.1574	0.0752	0.2486	0.1258	0.0186	0.0179
PSO	Ave	220.753	184.743	227.841	80.384	85.745	91.706
	Std	70.7651	60.4521	61.9641	63.4123	73.8324	61.8361
	p-value	0.4628	0.1834	0.2963	0.3654	0.4176	0.3586
GA	Ave	218.126	188.173	232.176	85.175	91.756	88.412
	Std	68.7523	70.1963	68.1863	71.1263	80.1967	53.1453
	p-value	0.2919	0.3451	0.4263	0.4672	0.5283	0.2948

Table 10
The details of IEEE CEC06-2019 test functions.

No.	Functions	$F_i^* = F_i(x*)$	D	Search range
F1	Storn's Chebyshev Polynomial Fitting Problem	1	9	[−8192,8192]
F2	Inverse Hilbert Matrix Problem	1	16	[−16 384,16 384]
F3	Lennard-Jones Minimum Energy Cluster	1	18	[−4,4]
F4	Rastrigin's Function	1	10	[−100,100]
F5	Griewank's Function	1	10	[−100,100]
F6	Weierstrass Function	1	10	[−100,100]
F7	Modified Schwefel's Function	1	10	[−100,100]
F8	Expanded Schaffer's F6 Function	1	10	[−100,100]
F9	Happy Cat Function	1	10	[−100,100]
F10	Ackley's Function	1	10	[−100,100]

Table 11
The results of algorithms on IEEE CEC06-2019 test functions.

Rank	Algorithm	Test functions										Total
		F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	
1	OA	10	10	10	10	10	10	10	10	9.26	10	99.26
2	CapSA	10	10	10	10	10	10	9.45	8.19	8.39	10	96.03
3	ChOA	10	10	10	10	10	9.50	9.24	7.65	8.06	10	94.45
4	BWO	10	10	10	10	9.70	9.85	8.19	5.89	5.43	10	89.06
5	PSO	10	10	10	10	9.75	9.09	7.96	4.52	4.86	10	86.18
6	GA	10	10	10	9.51	9.20	8.19	4.06	3.69	3.86	10	78.49

3.4.1. Statistical analysis of the algorithms implementation results

Table 13 summarizes the superiority of the meta-heuristics concerning the "BestF(x)" and "AvgF(x)" parameters. In some cases, several algorithms have jointly achieved the best value of the "BestF(x)" parameter. As can be seen, the OA in 22 test functions out of 33 functions (66.66%) has achieved the best value of the "BestF(x)". TGA in 19 functions (57.57%), CMAES in 17 functions (51.51%), WSA in 15 functions (45.45%), BPA and IBPA in 5 functions (15.15%), and LADA in 4 functions (12.12%), TS in 3 functions (09.9%) and SA in 2 functions (06.6%) have achieved the best value of "BestF(x)". Also, OA in 17 test functions (51.51%), TGA and CMAES in 14

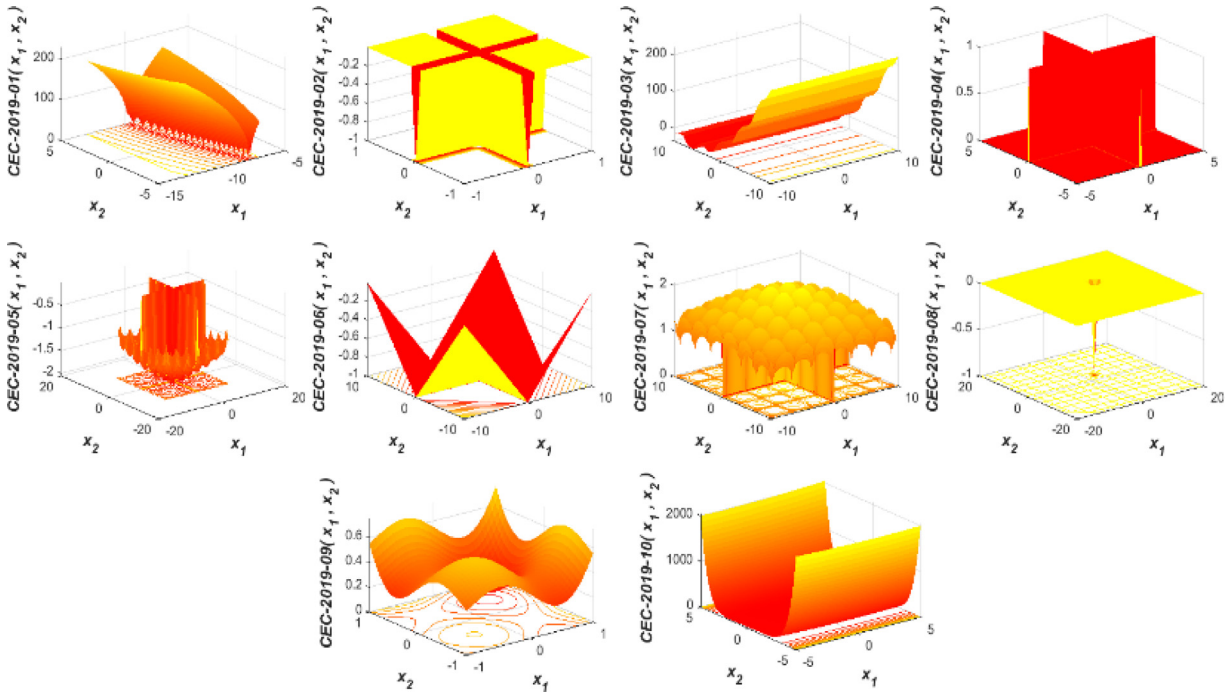


Fig. 10. The 3D plot of IEEE CEC06-2019 test functions.

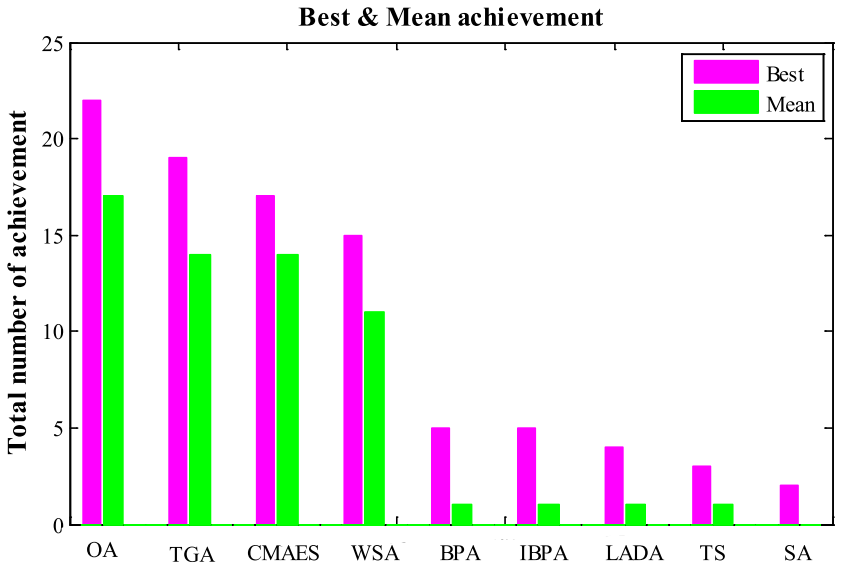


Fig. 11. The superiority of the meta-heuristic algorithms in "BestF(x)" and "MeanF(x)" achievement.

functions (42.42%), WSA in 11 functions (33.33%), BPA, IBPA, LADA and TS in one function (03.3%), and SA in no function (0%) have achieved the best value of "AvgF(x)". The data presented in Table 13 are compared visually in Fig. 11. The figure shows the total number of its superiority in the 33 benchmark functions concerning the "BestF(x)" and "MeanF(x)" columns. This figure shows the obvious superiority of the OA.

In Fig. 12, for each algorithm, the values of "AvgRunime" (the average execution time) of the algorithm on all benchmark functions are added up and presented as the "Total execution time" values. As shown in the figure, the

Table 12
The results of algorithms on IEEE CEC06-2019 test functions.

Function	Criteria	BP_{AMH}	IBP_{AMH}	LAD_{AMH}	TS_{MH}	SA_{MH}	WS_{AMH}	$CMAES_{MH}$	TGA_{MH}	OA_{MH}
F1(ACK)	BestF(x)	4.11E−3	8.15E−3	8.822E−3	1.4185E−1	1.261E−2	0.000	1.2E−9	0.000	1.154E−14
	AvgF(x)	2.843E−2	2.260E−2	4.73E−3	3.852E−1	5.488E−2	0.000	1.8E−9	0.000	1.154E−14
	SDF(x)	1.338E−2	1.021E−2	1.57E−3	7.488E−3	5.456E−2	0.000	4E−10	0.000	0.000
	AvgTime	112.00	102.00	68.00	108.00	115.00	33.00	16.00	12.00	57.14
F2(AP)	BestF(x)	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5239E−1	−0.35239
	AvgF(x)	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5238E−1	−3.5213E−1	−3.5236E−1	−3.523E−1	−3.5239E−1	−0.35239
	SDF(x)	1.4490E−6	1.0670E−6	5.5760E−7	2.1830E−5	6.6540E−5	8.7610E−6	0.000	6.090E−7	0.000
	AvgTime	71.00	65.00	36.00	66.00	75.00	20.00	12.00	9.00	3.01
F3(BL)	BestF(x)	1.0170E−8	3.2170E−9	1.2590E−9	3.9550E−7	1.3740E−6	5.5890E−8	0.000	1.07E−08	0.000
	AvgF(x)	2.6970E−7	2.8260E−7	2.4860E−7	7.6370E−6	2.7220E−5	1.2670E−7	0.000	3.70E−07	0.000
	SDF(x)	2.3230E−7	2.8380E−7	2.7040E−7	6.3020E−6	2.9700E−5	3.8770E−8	0.000	5.84E−07	0.000
	AvgTime	55.00	47.00	22.00	48.00	57.00	17.00	12.00	9.00	3.18
F4(BF1)	BestF(x)	3.6560E−5	2.3620E−7	6.6280E−6	1.1230E−4	2.5920E−5	0.000	0.000	0.000	0.000
	AvgF(x)	1.79E−3	6.4190E−4	5.2500E−4	2.104E−2	2.76E−3	0.000	0.000	0.000	0.000
	SDF(x)	1.59E−3	7.6110E−4	1.22E−3	1.801E−2	3.06E−3	0.000	0.000	0.000	0.000
	AvgTime	62.00	55.00	29.00	57.00	70.00	24.00	13.00	10.00	2.82
F5(BF2)	BestF(x)	2.9270E−6	3.3970E−6	4.9110E−5	6.2230E−4	1.5890E−4	0.000	0.000	0.000	0.000
	AvgF(x)	1.03E−3	7.86E−3	3.24E−3	2.354E−2	3.96E−3	0.000	0.000	0.000	0.000
	SDF(x)	1.44E−3	3.976E−2	6.43E−3	2.656E−2	6.67E−3	0.000	0.000	0.000	0.000
	AvgTime	64.00	57.00	29.00	58.00	71.00	24.00	12.00	9.00	2.44
F6(BP)	BestF(x)	3.9788E−1	3.9788E−1	3.9788E−1	3.9788E−1	3.9788E−1	3.9788E−1	3.9788E−1	3.9789E−1	0.39789
	AvgF(x)	3.9788E−1	3.9788E−1	3.9791E−1	3.9792E−1	3.9796E−1	0.39790	3.9788E−1	3.9789E−1	0.39789
	SDF(x)	2.3610E−6	2.1560E−6	9.3730E−5	3.5430E−5	8.4430E−5	9.4750E−6	0.000	0.000	0.000
	AvgTime	60.00	53.00	24.00	53.00	63.00	18.00	12.00	9.00	3.22
F7(CB3)	BestF(x)	6.0350E−10	5.4050E−9	7.5220E−9	1.3850E−8	1.1740E−6	0.000	0.000	0.000	0.000
	AvgF(x)	4.2850E−7	3.1990E−7	8.6260E−7	1.5030E−5	3.4710E−5	0.000	0.000	0.000	0.000
	SDF(x)	6.2550E−7	3.1230E−7	1.7520E−6	1.4370E−5	3.4580E−5	0.000	0.000	0.000	0.000
	AvgTime	94.00	86.00	58.00	86.00	96.00	25.00	14.00	10.00	3.11
F8(CB6)	BestF(x)	−0.103162E1	−0.103162E1	−0.103162E1	−0.103162E1	−0.103162E1	−0.103161E1	−0.103163E1	−0.103163E1	−1.03163
	AvgF(x)	−0.103162E1	−0.103162E1	−0.103162E1	−0.103160E1	−0.103155E1	−0.103161E1	−0.103163E1	−0.103113E1	−1.03163
	SDF(x)	2.2550E−6	2.2210E−6	1.7470E−6	2.0600E−5	5.6150E−5	4.5160E−16	0.000	1.450E−03	0.000
	AvgTime	118.00	111.00	82.00	112.00	122.00	19.00	12.00	9.00	3.16
F9(CM)	BestF(x)	1.9999E−1	1.9999E−1	1.9999E−1	1.9999E−1	1.9999E−1	0.200	0.200	0.200	0.200
	AvgF(x)	1.9999E−1	1.9999E−1	1.9999E−1	1.9998E−1	1.9984E−1	0.200	0.200	0.200	0.200
	SDF(x)	2.6470E−7	1.6460E−7	6.4310E−8	4.2710E−6	3.3090E−5	8.4690E−17	0.000	0.000	0.00.0
	AvgTime	71.00	65.00	33.00	59.00	78.00	25.00	13.00	11.00	3.10
F9(CM)	BestF(x)	3.9999E−1	3.9999E−1	3.9987E−1	3.9994E−1	3.9991E−1	0.400	0.400	0.400	0.400
	AvgF(x)	3.9999E−1	3.9999E−1	3.9873E−1	3.9937E−1	3.9855E−1	0.400	0.400	0.400	0.400
	SDF(x)	4.3540E−6	4.9510E−6	6.9090E−4	3.6040E−4	7.0500E−4	1.6930E−16	0.000	0.000	0.000
	AvgTime	89.00	82.00	59.00	78.00	97.00	26.00	13.00	11.00	8.03
F10(DA)	BestF(x)	−24 776.510	−24 776.510	−24 776.510	−24 776.510	−24 776.510	−24 776.520	−24 776.520	−24 775.720	−24 776.520
	AvgF(x)	−24 776.470	−24 776.470	−24 776.390	−24 776.240	−24 776.460	−24 776.490	−24 776.520	−24 775.660	−24 776.520
	SDF(x)	5.370E−2	5.579E−2	2.5194E−1	2.7919E−1	9.251E−2	1.570E−2	0.000	9.4458E−2	0.000
	AvgTime	74.00	67.00	39.00	67.00	83.00	26.00	12.00	9.00	3.04
F11(EP)	BestF(x)	−9.9999E−1	−9.9999E−1	−9.9999E−1	−9.9999E−1	−9.9999E−1	−9.9999E−1	−1.000	−9.9999E−1	−1.000
	AvgF(x)	−9.9999E−1	−8.3334E−1	−9.9999E−1	−4.6667E−1	−9.9991E−1	−9.9957E−1	−1.000	−9.9999E−1	−0.98496
	SDF(x)	2.0030E−6	3.79001E−1	2.8850E−6	5.0733E−1	1.0110E−4	2.0250E−4	0.000	1.590E−6	5.945E−4
	AvgTime	73.00	66.00	37.00	66.00	73.00	27.00	12.00	9.00	3.36
F12(EM)	BestF(x)	−0.955986E1	−0.952608E1	−0.927356E1	−0.897041E1	−0.952701E1	−0.720847E1	−0.861049E1	−0.783315E1	−8.1698
	AvgF(x)	−0.919363E1	−0.928089E1	−0.859255E1	−0.872569E1	−0.914595E1	−0.674067E1	−0.598581E1	−0.734387E1	−7.6089
	SDF(x)	2.1804E−1	1.6721E−1	2.6812E−1	1.3066E−1	2.2567E−1	7.655710	1.061680	3.3437E−1	0.16043
	AvgTime	394.00	379.00	336.00	390.00	399.00	54.00	22.00	17.00	45.75
F13(EXP)	BestF(x)	9.9998E−1	9.9998E−1	9.9986E−1	9.9994E−1	9.9207E−1	1.000	1.000	1.000	1.000
	AvgF(x)	9.9984E−1	9.9995E−1	9.9968E−1	9.9752E−1	9.8487E−1	1.000	1.000	1.000	1.000
	SDF(x)	2.4690E−5	1.6340E−5	1.1230E−4	7.9750E−4	4.46E−3	0.000	0.000	0.000	0.000
	AvgTime	97.00	89.00	39.00	87.00	103.00	30.00	14.00	10.00	46.21
F14(GP)	BestF(x)	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000	3.002070	3.000000
	AvgF(x)	3.000020	5.700010	10.007100	3.000530	3.000490	3.000320	3.000200	3.112530	3.000000
	SDF(x)	2.5860E−5	8.238470	16.466700	5.7510E−4	1.140E−3	1.6220E−4	2.960E−16	1.340E−01	0.000
	AvgTime	51.00	45.00	16.00	45.00	58.00	24.00	12.00	9.00	3.41
F15(HSK)	BestF(x)	1.2080E−6	5.3990E−6	8.1240E−5	3.1200E−5	2.3520E−5	32.830	3.728920	1.520E−05	1.054E−6
	AvgF(x)	0.001290	0.001570	5.3620E−4	2.0470E−4	2.8680E−4	32.830	3.728920	7.050E−02	1.830 E-4
	SDF(x)	0.001300	0.001620	3.4560E−4	1.3820E−4	2.2220E−4	1.4450E−15	0.000	2.600E−01	6.172E−6
	AvgTime	5234.00	5211.00	5205.00	5207.00	5269.00	121.00	78.00	51.00	8.85
F16(GRP)	BestF(x)	−2.345810	−2.345810	−2.345810	−2.345810	−2.345810	−2.345810	−2.345810	−2.345810	−2.34581
	AvgF(x)	−2.345810	−2.345810	−2.345810	−2.345790	−2.345730	−2.345580	−2.345760	−2.345810	−2.34581
	SDF(x)	8.2030E−8	1.3560E−7	5.3270E−8	3.1690E−6	1.9940E−5	1.6310E−4	3.2730E−6	1.360E−07	1.0900E−10
	AvgTime	57.00	50.00	22.00	50.00	58.00	27.00	15.00	9.00	3.05
F17(KL)	BestF(x)	3.0750E−4	3.0750E−4	3.1050E−4	3.0750E−4	3.0870E−4	3.0790E−4	3.0860E−4	3.0840E−4	3.075E−4
	AvgF(x)	3.1050E−4	3.1180E−4	3.6840E−4	3.0830E−4	3.2040E−4	3.1110E−4	3.2040E−4	3.1440E−4	3.123E−4
	SDF(x)	3.4880E−6	4.0960E−6	6.4000E−5	6.6380E−7	6.8420E−6	1.7060E−6	6.6480E−7	7.4020E−6	2.167E−12
	AvgTime	89.00	84.00	54.00	83.00	91.00	57.00	22.00	10.00	8.43
F18(LM1)	BestF(x)	1.0050E−7	1.0950E−7	8.2400E−8	3.1240E−6	3.5150E−5	3.3150E−7	0.000	3.470E−04	4.131E−15
	AvgF(x)	7.4570E−6	4.9790E−6	2.3220E−6	1.4360E−4	3.3920E−4	6.7050E−6	0.000	2.870E−02	1.020E−06
	SDF(x)	6.0410E−6	6.7160E−6	2.8670E−6	1.1860E−4	2.2570E−4	3.3920E−6	0.000	2.830E−02	1.000E−12
	AvgTime	84.00	76.00	48.00	76.00	86.00	37.00	23.00	19.00	5.83
F19(LM2)	BestF(x)	1.38E−3	1.882E−2	1.8022E−1	1.7907E−1	7.06E−3	001.03E−3	0.000	0.000	1.914E−07
	AvgF(x)	3.0253E−1	5.5175E−1	2.629750	5.7022E−1	2.7072E−1	2.390E−2	0.000	0.000	2.285E−06
	SDF(x)	2.7903E−1	4.8275E−1	6.7992E−1	3.2601E−1	2.5005E−1	1.237E−2	0.000	0.000	4.206E−12

(*continued on next page*)

Table 12 (*continued*).

Function	Criteria	BP_{AMH}	IBP_{AMH}	LAD_{AMH}	TS_{MH}	SA_{MH}	WS_{AMH}	$CMAES_{MH}$	TGA_{MH}	OA_{MH}
	AvgTime	64.00	58.00	36.00	58.00	72.00	29.00	27.00	18.00	11.39
F19(LM2)	BestF(x)	3.87E−3	9.62E−3	6.966E−2	5.080E−2	1.122E−2	2.570E−3	2.836 E-2	1.730E−3	3.940E−4
	AvgF(x)	3.5532E−1	4.7384E−1	5.9860E−1	4.5002E−1	3.5846E−1	8.629E−2	3.6960E−1	3.4527E−1	2.0656E−2
	SDF(x)	4.0486E−1	4.1149E−1	4.9381E−1	2.4229E−1	2.8039E−1	5.220E−2	3.1849E−1	4.0532E−1	1.811E−4
	AvgTime	65.00	58.00	36.00	58.00	72.00	33.00	31.00	20.00	44.74
F20(MR)	BestF(x)	4.3560E−5	4.3580E−5	4.4260E−5	4.3570E−5	4.5740E−5	4.4090E−5	4.7730E−05	4.0580E−05	1.900E−03
	AvgF(x)	6.0490E−5	6.0840E−5	6.4960E−5	4.8320E−5	5.9300E−5	4.7730E−5	9.9900E−05	4.9920E−05	1.931E−03
	SDF(x)	3.0540E−5	2.2700E−5	2.2640E−5	7.2990E−6	1.0270E−5	2.2710E−6	5.1820E−05	4.9720E−05	1.5830E−09
	AvgTime	68.00	61.00	34.00	61.00	70.00	25.00	32.00	23.00	4.53
F21(MCP)	BestF(x)	6.6840E−12	5.6060E−12	5.2600E−8	3.3760E−11	1.0980E−9	8.9340E−13	0.000	0.000	0.000
	AvgF(x)	9.1850E−9	1.1200E−8	2.6270E−6	1.3530E−9	4.0940E−7	2.4140E−10	0.000	0.000	1.481E−08
	SDF(x)	8.5760E−9	1.2350E−8	3.1290E−6	2.1670E−9	5.1130E−7	1.7530E−10	0.000	0.000	6.475E−16
	AvgTime	147.00	140.00	127.00	140.00	152.00	79.00	12.00	10.00	7.82
F22(MRP)	BestF(x)	1.9600E−6	5.8650E−7	3.2100E−8	1.6230E−5	1.7320E−5	4.2910E−9	0.000	0.000	0.000
	AvgF(x)	8.6320E−4	2.110E−3	1.470E−3	7.9070E−4	1.05E−3	6.8120E−7	2.970E−3	2.230E−3	4.070E−4
	SDF(x)	2.37E−3	4.030E−3	2.180E−3	7.948E−4	1.0370E−4	3.8710E−7	3.830E−3	3.580E−3	6.124E−7
	AvgTime	59.00	52.00	23.00	52.00	64.00	17.00	12.00	9.00	4.32
F23(MGP)	BestF(x)	1.296950	1.296950	1.296950	1.296940	1.296940	1.296950	1.295560	1.296950	1.29695
	AvgF(x)	1.296950	1.275570	1.291600	1.296810	1.296650	1.296660	1.291880	1.295400	1.29490
	SDF(x)	4.8880E−6	3.605E−2	2.033E−2	1.2860E−4	3.4700E−4	1.7780E−4	3.632E−2	4.89E−3	1.317E−6
	AvgTime	121.00	114.00	86.00	114.00	121.00	35.00	13.00	10.00	3.07
F24(PWQ)	BestF(x)	1.6510E−5	2.2940E−7	3.5300E−4	1.320E−3	1.1130E−4	0.000	1.000E−7	0.000	0.000
	AvgF(x)	1.360E−3	1.020E−3	3.571E−2	1.224E−2	3.560E−3	0.000	5.000E−7	0.000	0.
	SDF(x)	1.330E−3	1.360E−3	3.693E−2	8.970E−3	3.300E−3	0.000	3.000E−7	0.000	0.000
	AvgTime	97.00	91.00	69.00	92.00	106.00	29.00	12.00	10.00	11.50
F25(RG)	BestF(x)	1.6961E−1	8.790E−2	6.060E−3	4.587530	3.932E−2	0.000	4.974800	0.000	3.4665
	AvgF(x)	4.3289E−1	2.9275E−1	1.584E−2	6.355410	1.5916E−1	0.000	14.613620	0.000	6.0491
	SDF(x)	1.6469E−1	1.2481E−1	5.540E−3	8.9405E−1	9.522E−2	0.000	9.552800	0.000	1
	AvgTime	133.00	125.00	61.00	134.00	141.00	32.00	22.00	14.00	43.90
F26(RB)	BestF(x)	3.300500	1.657800	13.11610	24.73950	9.1870E−1	8.91670	2.010E−2	5.653E−1	3.9037
	AvgF(x)	13.56810	12.14200	26.47400	66.10240	6.41090	8.94490	6.030E−2	8.231E−1	7.8884
	SDF(x)	17.97070	14.92020	14.95210	19.17630	1.81800	1.600E−2	4.390E−2	3.419E−1	2.0973
	AvgTime	89.00	83.00	34.00	83.00	98.00	25.00	14.00	9.00	46.29
F27(SAL)	BestF(x)	9.987E−2	9.987E−2	9.987E−2	9.996E−2	9.987E−2	0.000	9.987E−2	9.987E−2	9.987E−2
	AvgF(x)	2.0329E−1	1.8655E−1	1.4497E−1	3.0807E−1	2.0668E−1	0.000	9.988E−2	1.3987E−1	1.6654E−1
	SDF(x)	9.609E−2	7.301E−2	4.966E−2	8.101E−2	9.040E−2	0.000	1.000E−5	5.164E−2	6.660E−3
F27(SAL)	AvgTime	67.00	60.00	26.00	60.00	69.00	20.00	17.00	13.00	12.17
	BestF(x)	2.9988E−1	2.9987E−1	3.0989E−1	3.152E−2	3.0006E−1	0.000	1.0408E−1	9.987E−2	1.9987E−1
	AvgF(x)	4.5003E−1	4.7596E−1	5.7499E−1	1.39242	5.9059E−1	0.00	1.7422E−1	1.1987E−1	4.1987E−1
	SDF(x)	8.947E−2	1.2510E−1	8.872E−2	1.6112E−1	1.5625E−1	0.000	4.158E−2	4.216E−2	1.066E−2
F28(SF1)	AvgTime	91.00	84.00	40.00	86.00	97.00	24.00	14.00	11.00	43.88
	BestF(x)	1.20510E−4	1.9790E−5	1.5350E−6	1.4090E−4	6.6710E−4	0.000	7.77E−3	0.000	0.000
	AvgF(x)	7.490E−3	5.360E−3	5.570E−3	6.210E−3	8.400E−3	0.000	9.840E−3	0.000	0.000
	SDF(x)	3.620E−3	4.740E−3	4.440E−3	3.430E−3	2.770E−3	0.000	1.230E−3	0.000	0.000
	AvgTime	65.00	62.00	28.00	56.00	63.00	25.00	16.00	13.00	3.01
F29(SF2)	BestF(x)	6.332E−2	8.329E−2	2.1116E−1	5.8920E−1	6.096E−2	8.93710E−7	1.27875E−1	0.000	1.417E−07
	AvgF(x)	1.6367E−1	3.217740	3.248470	3.633560	3.0990E−1	1.20110E−6	2.33297E−1	0.000	1.417E−07
	SDF(x)	5.4980E−2	1.405960	1.336470	7.7425E−1	3.5340E−1	1.34310E−7	7.68640E−2	0.000	0.000
	AvgTime	102.00	103.00	67.00	95.00	104.00	23.00	15.00	14.00	4.02
F30(SWF)	BestF(x)	−4188.830	−4188.800	−4189.790	−4146.860	−4189.640	−4080.130	−8340.03860	−4189.7020	−3751.5666
	AvgF(x)	−4186.570	−4187.480	−4189.740	−4099.800	−4189.390	−3505.520	−6835.18370	−4189.2920	−3593.0374
	SDF(x)	1.417130	9.0048E−1	3.6700E−2	23.245700	2.3796E−1	328.200	750.730	2.3796E−1	8455.3307
	AvgTime	137.00	129.00	71.00	134.00	145.00	27.00	25.00	18.00	48.12

Table 13
The superiority of the meta-heuristic algorithms based on Table 12.

No	OA_{MH}		TGA_{MH}		$CMAES_{MH}$		WSA_{MH}		BPA_{MH}		$IBPA_{MH}$		$LADA_{MH}$		TS_{MH}		SA_{MH}	
	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
1	F.AP	F.AP	F.ACK	F.ACK	F.BL	F.BL	F.BF1	F.BF1	F.EM	F.MGP	F.GP	F.EM	F.GP	F.SWF	F.GP	F.KL	F.GP	
2	F.BL	F.BL	F.AP	F.BF1	F.BF1	F.BF1	F.BF2	F.BF2	F.GP		F.HSK		F.HSK		F.HSK		F.HSK	
3	F.BF1	F.BF1	F.BF1	F.BF2	F.BF2	F.BF2	F.CB3	F.CB3	F.HSK		F.KL		F.MGP		F.KL			
4	F.BF2	F.BF2	F.BF2	F.BP	F.CB3	F.CB3	F.CM(2)	F.EXP	F.KL		F.MR		F.SWF					
5	F.BP	F.BP	F.BP	F.CB3	F.CB6	F.CB6	F.CM(4)	F.MR	F.MGP		F.MGP							
6	F.CB3	F.CB3	F.CB3	F.CM(2)	F.CM(2)	F.CM(2)	F.DA	F.MRP										
7	F.CB6	F.CB6	F.CB6	F.CM(4)	F.CM(4)	F.CM(4)	F.EXP	F.PWQ										
8	F.CM(2)	F.CM(2)	F.CM(2)	F.EXP	F.DA	F.DA	F.GP	F.RG										
9	F.CM(4)	F.CM(4)	F.CM(4)	F.LM2(5)	F.EP	F.EP	F.HSK	F.SAL(5)										
10	F.DA	F.DA	F.EXP	F.MCP	F.EXP	F.EXP	F.MGP	F.SAL(10)										
11	F.EP	F.EXP	F.HSK	F.PWQ	F.GP	F.LM1	F.PWQ	F.SF1										
12	F.EXP	F.GP	F.LM2(5)	F.RG	F.HSK	F.LM2(5)	F.RG											
13	F.GP	F.GRP	F.MCP	F.SF1	F.LM1	F.MCP	F.SAL(5)											
14	F.GRP	F.HSK	F.MRP	F.SF2	F.LM2(5)	F.RB	F.SAL(10)											
15	F.HSK	F.LM2	F.MGP		F.MCP		F.SF1											
16	F.KL	F.PWQ	F.PWQ		F.MRP													
17	F.LM2	F.SF1	F.RG		F.RB													
18	F.MCP		F.SF1															
19	F.MRP		F.SF2															
20	F.MGP																	
21	F.PWQ																	
22	F.SF1																	

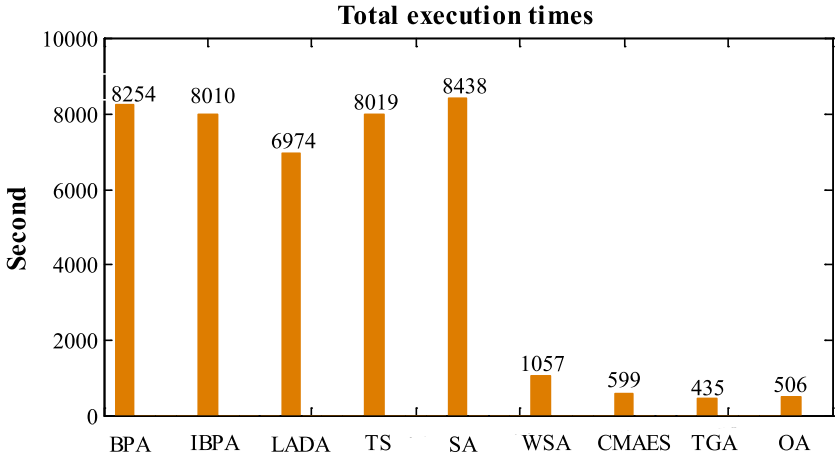


Fig. 12. Comparison of total runtimes of the meta-heuristic algorithms.

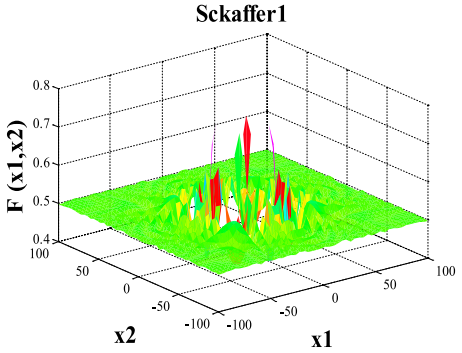


Fig. 13. The 3-dimensional visualization of the SF1.

"Total runtime" value of the OA is more than that of the TGA and less than those of the other seven algorithms. It should also be mentioned that the OA takes more time than the TGA only in 10-dimensional functions. In all other 24 functions, the OA takes fewer times. In fact, in 24 out of 33 functions, the OA takes the least value of the "AvgRunime", which shows a 72.72 percent success. As shown in Fig. 12, the runtime of the OA, TGA, CMAES, and WSA are much less than those of the BPA, IBPA, LADA, TS, and SA.

3.4.2. The convergence trend of the OA

In this section, the convergence curve of the OA on the SF1 test function is studied. The OA is run with an initial population of 100 seedlings and 20 repetitions. Then, the population distribution in steps of four generations is presented to show the algorithm's convergence visually. The 3D visualization of the SF1 is illustrated in Fig. 13.

To show the convergence trend of the SF1, first, the contour visualization of the SF1 is presented, and then the population distribution is presented. The population distribution is presented on a different contour visualization every four iterations. These are shown in Fig. 14. The figure shows the adequate diversification of the initial population. With the algorithm's progress, the convergence trend shows an adequate level of exploitation while keeping enough exploration in the population. The remaining exploration is the result of creating a few seedlings randomly in each generation (in the screening operation). Similar to the GA and ABC algorithms, in the orchard algorithm, all the generation populations are not necessarily converged to the global optimum. In each generation, a few new solutions are created to search the problem space randomly. Therefore, even in the final generations, some population members are still in diverse locations of the problem space. This reduces the risk of local optimums.

The convergence trend diagrams of the OA related to 30 different test functions and a 3D plot of functions are also shown in Fig. 15. In 20 functions, the algorithm convergence is achieved in less than 10 iterations. In 5 other

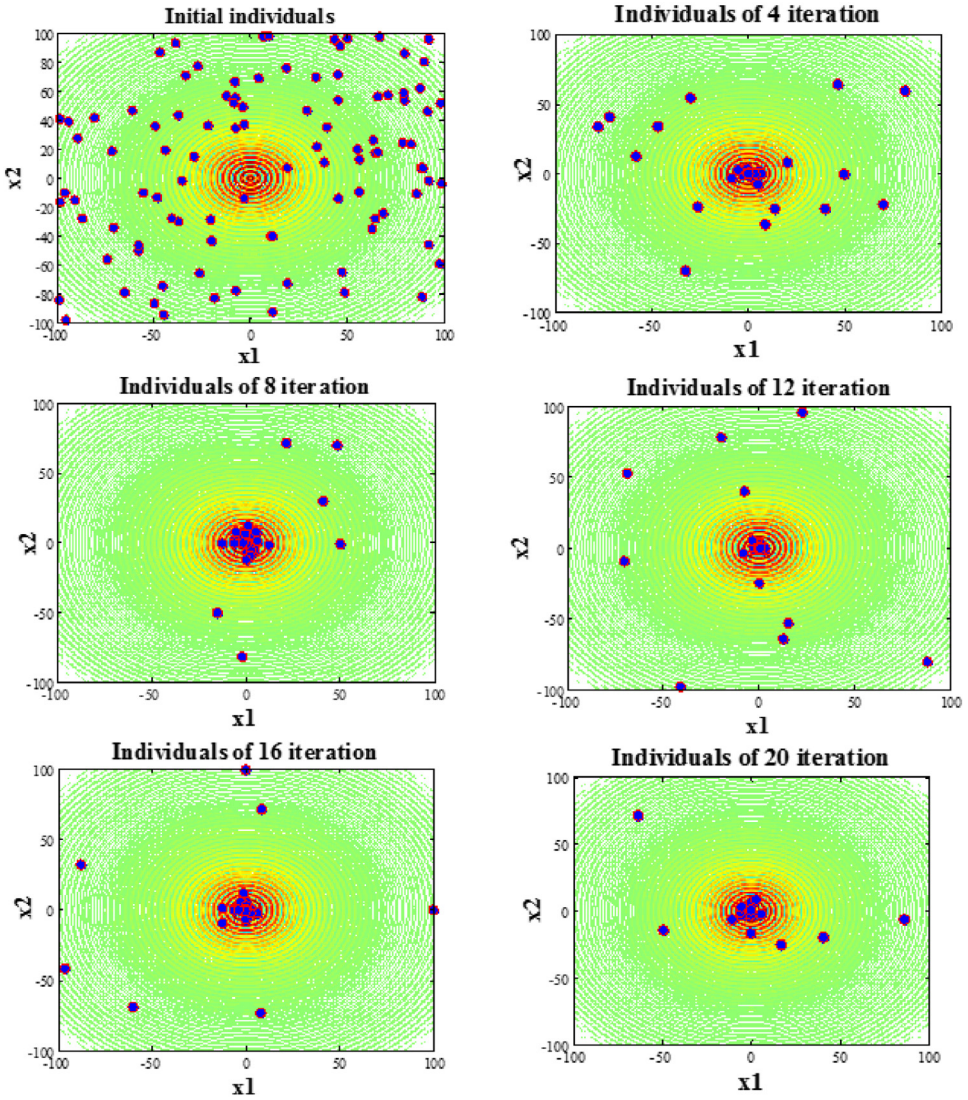


Fig. 14. Convergence trend of the OA for SF1.

functions, it is achieved in less than 50 iterations. These results show the high convergence trend of the OA in solving such problems.

3.5. Part 5: Real optimization problems

This section uses four real problems to evaluate the OA more accurately. The problems considered in this section assess the proposed method from different perspectives. Due to the complexity of these problems, solving them can be a challenge for meta-heuristic algorithms.

3.5.1. Design of a pressure vessel

Optimizing the cost of producing a pressure vessel with two parabolic heads is chosen as the first real problem for evaluating the OA's performance. The schematic view of the problem is shown in Fig. 16 and can be formulated as Eqs. (6)–(11) [89].

$$f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{6}$$

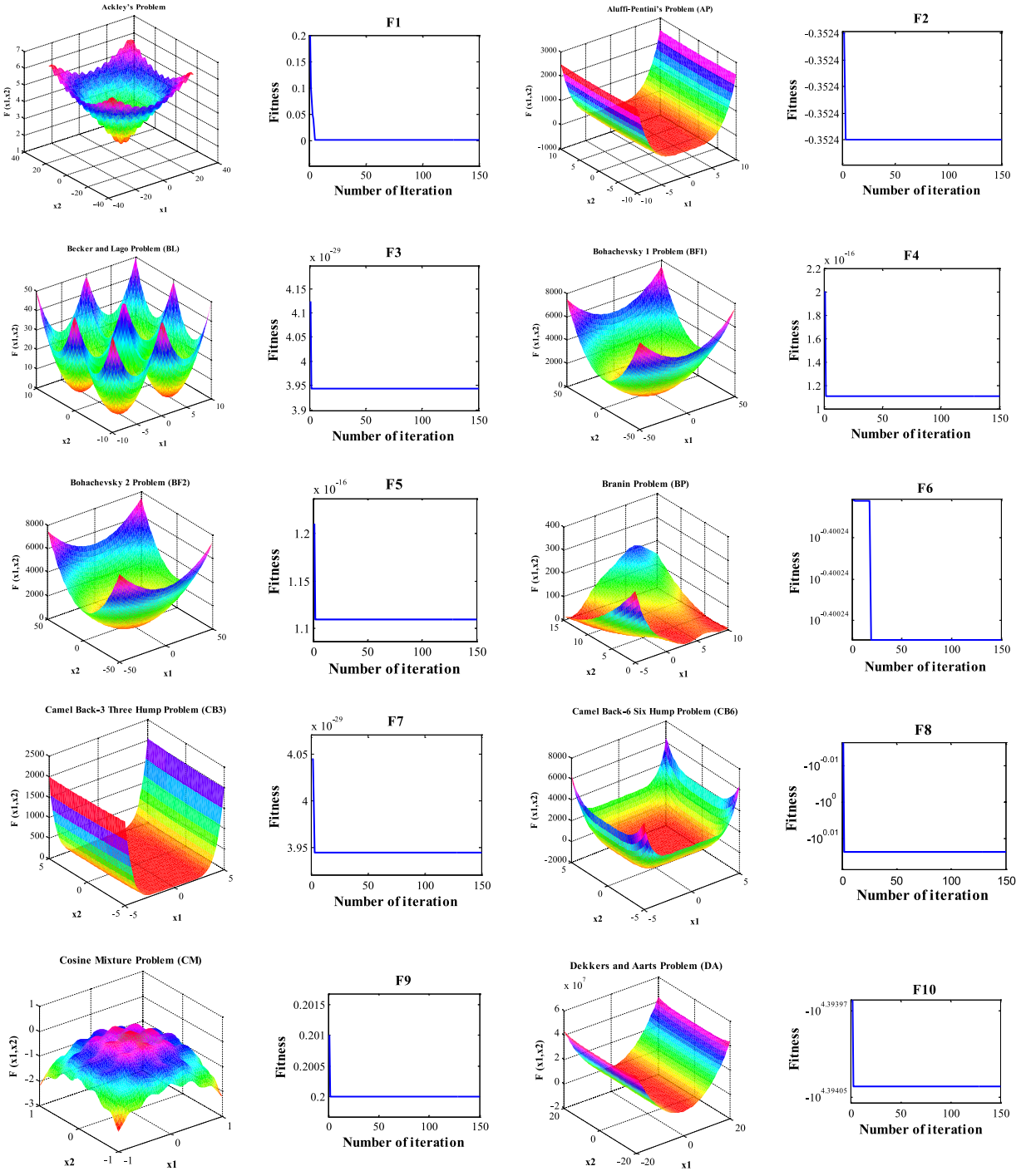


Fig. 15. Convergence trend of OA on the thirty test functions and 3D plot of test functions.

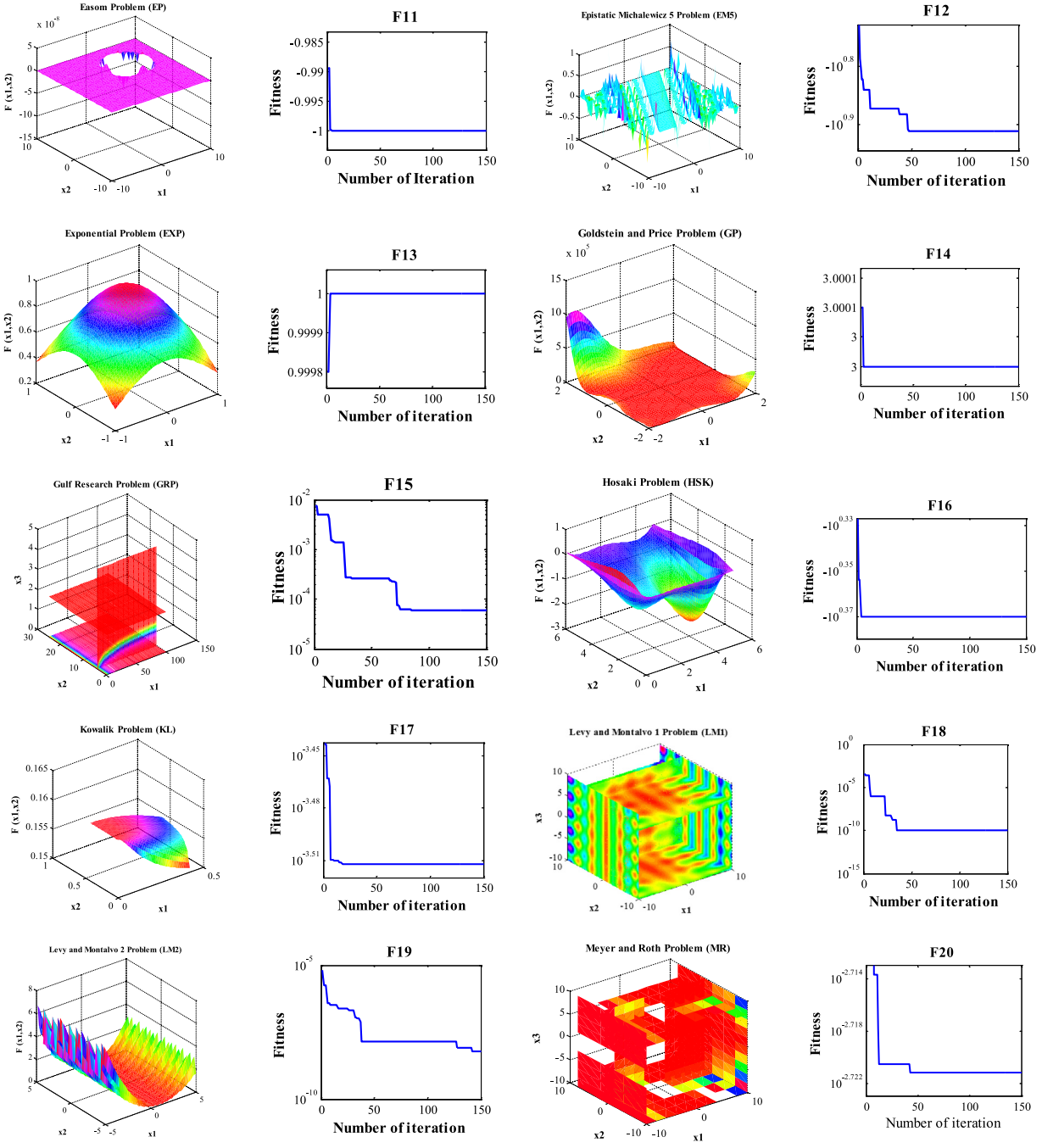


Fig. 15. (*continued*).

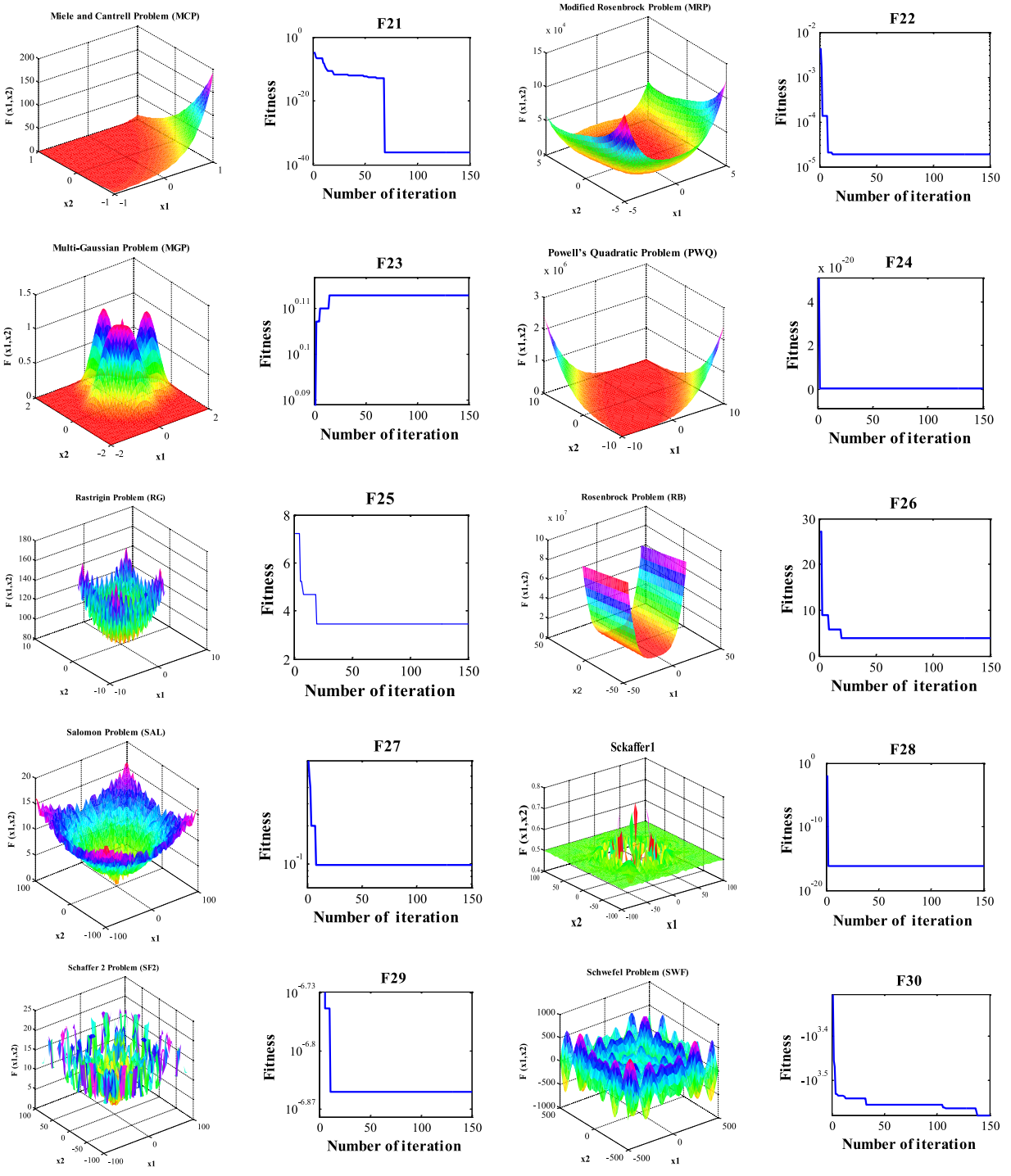


Fig. 15. (*continued*).

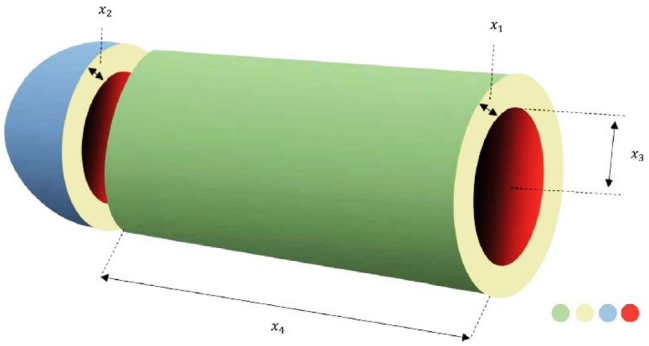


Fig. 16. The pressure vessel system [33].

Table 14
The results of the algorithms for pressure vessel system.

Algorithm	Best fitness ($)	Mean fitness ($)	Standard deviation	Iteration
OA	5905.7031	5909.3240	0.0007635	300
CapSA	6085.2365	6195.7892	0.0475632	300
ChOA	6098.1795	6309.4789	0.8563271	300
BWO	6165.1785	6821.1478	5.2563981	300
PSO	6196.2869	7009.1453	9.2145896	300
GA	6345.2456	7179.9625	25.148563	300

Subjected to

$$g_1(X) = -x_1 + 0.0193x_3 \leq 0 \tag{7}$$

$$g_2(X) = -x_2 + 0.00954x_3 \leq 0 \tag{8}$$

$$g_3(X) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 129600 \leq 0 \tag{9}$$

$$g_4(X) = x_4 - 240 \leq 0 \tag{10}$$

$$0 \leq x_1, \quad x_2 \leq 100, \quad 10 \leq x_3, \quad x_4 \leq 200 \tag{11}$$

where x_1 is the shell thickness (T_s), x_2 is the head thickness (T_h), x_3 is the radius of the cylindrical shell (R), and x_4 is the length of the shell (L).

Table 14 shows the results of the meta-heuristic algorithms' implementations for the pressure vessel production problem. The best value of the fitness function (minimum cost) obtained by the OA algorithm is 5905/7031 $, which is better than the results of other algorithms. Such a good result is achieved in only 300 iterations, which confirms the great convergence of the OA algorithm (Fig. 17). The optimal values obtained from the OA are (X = 0.7843, 0.39063, 40.6316, 195.7031).

3.5.2. Welded beam system

In this section, the well-known problem of welded-beam is used as the second engineering problem to evaluate OA. Here, the goal is to minimize the total cost of construction. A schematic view of the problem is shown in Fig. 18 and can be formulated as Eqs. (12)–(20) [89].

$$f(X) = 1.10471x_1^2 x_2 + 0.04811x_3x_4(14 + x_2) \tag{12}$$

Subjected to

$$g_1(X) = \tau(x) - \tau_{max} \leq 0 \tag{13}$$

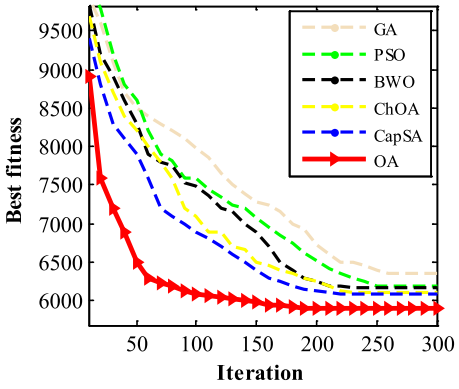


Fig. 17. The convergence trend of algorithms for pressure vessel system.

$$g_2(X) = \sigma(x) - \sigma_{max} \leq 0 \tag{14}$$

$$g_3(X) = x_1 - x_4 \leq 0 (15) \tag{15}$$

$$g_4(X) = 0.10471 x_1^2 + 0.04811 x_3 x_4 (14 + x_2) - 5 \leq 0 \tag{16}$$

$$g_5(X) = 0.125 - x_1 \leq 0 \tag{17}$$

$$g_6(X) = \delta(x) - \delta_{max} \leq 0 \tag{18}$$

$$g_7(X) = P - P_c(x) \leq 0 \tag{19}$$

$$0.1 \leq x_1, \quad x_4 \leq 2, \quad 0.1 \leq x_2, \quad x_3 \leq 0 \tag{20}$$

where

$$\tau(x) = \sqrt{\dot{\tau}^2 + 2\dot{\tau}\ddot{\tau} + \frac{x_2}{2R}\ddot{\tau}^2}, \quad \sigma(x) = \frac{6PL}{x_3^2 x_4}$$

$$\dot{\tau} = \frac{P}{\sqrt{2} x_1 x_2}, \quad \delta(x) = \frac{4PL^3}{E x_3^3 x_4}$$

$$\ddot{\tau} = \frac{PMR}{J}, \; P_c(x) = \frac{(4.013\sqrt{E((x_3^2 x_4^6)/36)})}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}})$$

$$M = P(L + \frac{x_2}{2})R = \sqrt{\frac{(x_2^2)}{4} + (\frac{(x_1 + x_3)}{2})^2}$$

$$J = 2\sqrt{2} x_1 x_2 [\frac{(x_2^2)}{12} + (\frac{(x_1 + x_3)}{2})^2]$$

where $P = 6000$ lb, $L = 14$ in, $E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{max} = 13600$ psi, $\sigma_{max} = 30000$ psi, and $\delta_{max} = 0.25$ in. $x_1 =$ The thickness of the weld. $x_2 =$ The length of the welded joint. $x_3 =$ The width of the beam. $x_4 =$ The thickness of the beam.

Table 15 shows the comparison of OA results with other algorithms. As can be seen, OA has obtained the best value of the fitness function compared to other algorithms. The optimal fitness function of OA is 1.8014 when the maximum number of iterations is considered to be 300. Fig. 19 indicates the convergence curve of OA and other algorithms. According to Fig. 19, the convergence curve of the OA is faster than other algorithms. The values $(X = 0.225, 3.225, 8.6938, 0.225)$ are also the optimal decision values obtained from the OA.

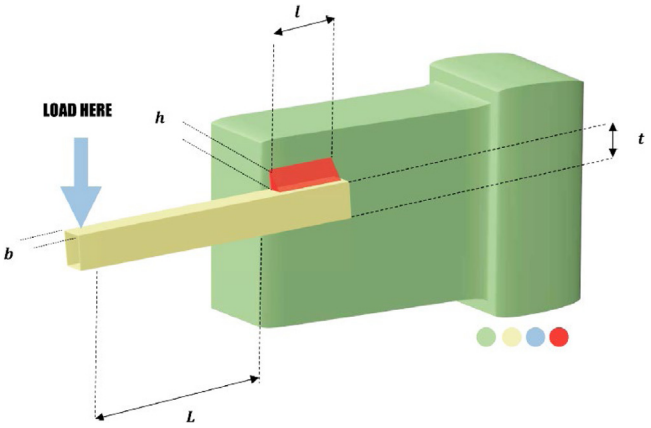


Fig. 18. Welded beam system [33].

Table 15
The results of the algorithms for pressure vessel system.

Algorithm	Best fitness	Mean fitness	Standard deviation	Iteration
OA	0.180140E1	0.180250E1	0.0000032	300
CapSA	0.205159E1	0.242652E1	0.0008563	300
ChOA	0.227529E1	0.258219E1	0.0045362	300
BWO	0.236753E1	0.372169E1	0.1722351	300
PSO	0.237396E1	0.395365E1	0.2874522	300
GA	0.238496E1	0.411563E1	2.2586314	300

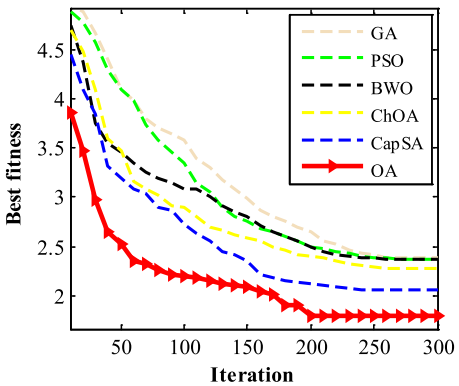


Fig. 19. The convergence curve of algorithms for welded beam problem.

3.5.3. Design of tension spring

Another complex problem used in this paper is the tension spring design. A schematic view of the problem is shown in Fig. 20 and can be formulated as Eqs. (21)–(26) [89].

$$f(X) = (x_3 + 2)x_2x_1^2 \tag{21}$$

Subjected to

$$g_1(X) = 1 - \frac{(x_3x_2^3)}{(71785x_1^4)} \leq 0 \tag{22}$$

$$g_2(X) = \frac{(4x_2^2 - x_1x_2)}{(12566(x_2x_1^3 - x_1^4))} + \frac{1}{(5108x_1^2)} - 1 \leq 0 \tag{23}$$

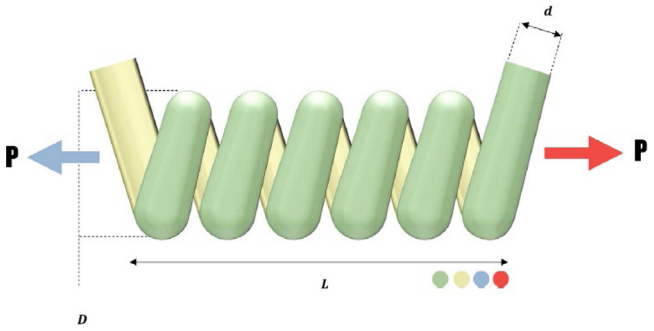


Fig. 20. Design of tension spring [33].

Table 16
Comparison of the results of algorithms for spring system design.

Algorithm	Best fitness	Mean fitness	Standard deviation	Iteration
OA	1.26650E−2	1.28653E−2	0.0000241	300
CapSA	1.26762E−2	1.39541E−2	0.0019852	300
ChOA	1.26908E−2	1.42365E−2	0.0169523	300
BWO	1.26819E−2	1.45215E−2	0.9852312	300
PSO	1.27156E−2	1.98565E−2	2.8563218	300
GA	1.27354E−2	2.10253E−2	3.9632518	300

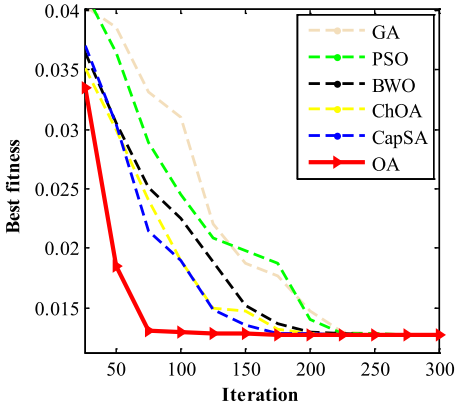


Fig. 21. The convergence curve of algorithms for spring system design problem.

$$g_3(X) = 1 - \frac{(140.45x_1)}{(x_2^2 x_3)} \leq 0 \tag{24}$$

$$g_4(X) = \frac{(x_1 + x_2)}{1.5} - 1 \leq 0 \tag{25}$$

$$0.05 \leq x_1 \leq 2.00, 0.25 \leq x_2 \leq 1.30, 2 \leq x_3 \leq 15 \tag{26}$$

where x_1 = Wire diameter (d). x_2 = Mean coil diameter (D). x_3 = The number of active coils (N).

Table 16 shows the implementation results of the OA and other algorithms on this problem. As can be seen, OA found the best value of the fitness function. The best-found solution to the problem is 0.012665. The OA convergence trend for this problem is shown in Fig. 21. According to Fig. 21, the convergence curve of the OA is faster than other algorithms

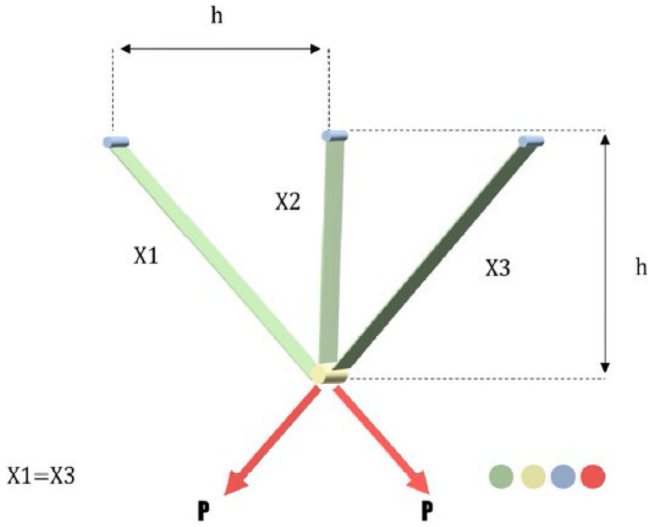


Fig. 22. Three-bar truss design problem.

Table 17
Comparison of the results of algorithms for spring system design.

Algorithm	Best fitness	Mean fitness	Standard deviation	Iteration
OA	263.895843	265.231456	0.0000184	300
CapSA	263.985239	271.852361	0.0016352	300
ChOA	264.253178	275.254785	0.0278562	300
BWO	264.351242	274.856381	0.3865291	300
PSO	265.743652	277.156325	2.5896275	300
GA	266.058963	278.796521	4.8965213	300

3.5.4. Three-bar truss design problem

The three-bar truss design problem is a widely used real-world engineering optimization problem. Three-bar truss problem intends to find the minimum weight of the Three-bar truss. A schematic view of the problem is shown in Fig. 22 and can be formulated as Eqs. (27)–(31) [89].

$$f(X) = (2\sqrt{2x_1} + x_2) \times l \tag{27}$$

Subjected to

$$g_1(X) = \frac{(\sqrt{2}x_1 + x_2)}{(\sqrt{2}x_1^2 + 2x_1x_2)} P - \sigma \leq 0 \tag{28}$$

$$g_2(X) = \frac{x_2}{(\sqrt{2}x_1^2 + 2x_1x_2)} P - \sigma \leq 0 \tag{29}$$

$$g_3(X) = \frac{1}{(\sqrt{2}x_2 + x_1)} P - \sigma \leq 0 \tag{30}$$

$$0 \leq x_1, x_2 \leq 1 \tag{31}$$

where $x_1 = A_1$. $x_2 = A_2$. $x_3 = x_1 = A_3$. $l = 100$ cm $P = 2$ kN/cm^2 $\sigma = 2$ kN/cm^2

Table 17 shows the comparison of OA results with other algorithms. As can be seen, OA found the best value of the fitness function. The best-found solution to the problem is 263.895843. The values ($x_1 = 0.788676944$, $x_2 = 0.408243170$) are also the optimal decision values obtained from the OA. According to the table, OA has the best standard deviation. Fig. 23 indicates the convergence curve of OA and other algorithms. The convergence curve of the OA is faster than other algorithms.

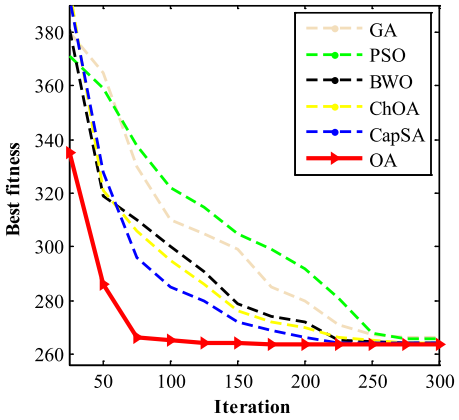


Fig. 23. The convergence curve of algorithms for three-bar truss design problem.

3.5.5. Feature selection problem

Classification of polarimetric synthetic aperture radar (POLSAR) images plays an important role in land cover management. Removing redundant features and improving classifier performance necessitates to use meta-heuristic and deep learning (DL) algorithms in feature selection and classification problems [83,94]. In this section, a hybrid approach of OA and deep convolutional neural networks (OA-DCNN) is proposed for optimal feature selection. Also, the data and images of Rostami and Kaveh's research [107] are used. The first step involves performing the necessary pre-processing, which includes radiometric calibration, Speckle-noise reduction and feature extraction from POLSAR images, as well as generating test and training samples. Then the proposed OA-DCNN approach is implemented to select the optimal features in the second phase. In the third step, the DCNN architecture is applied for land-cover classification. Fig. 24 shows the high resolution image and Pauli RGB image of study area.

Table 18 shows the results of the proposed algorithms for land-cover classification. For validation of classified images, sensitivity, overall accuracy, and specificity metrics are used to compare the efficiency of the hybrid architectures. These criteria can be calculated as Eqs. (32)–(34).

$$Sensitivity = \frac{TP}{(TP + FN)} \tag{32}$$

$$Specificity = \frac{TN}{(TN + FP)} \tag{33}$$

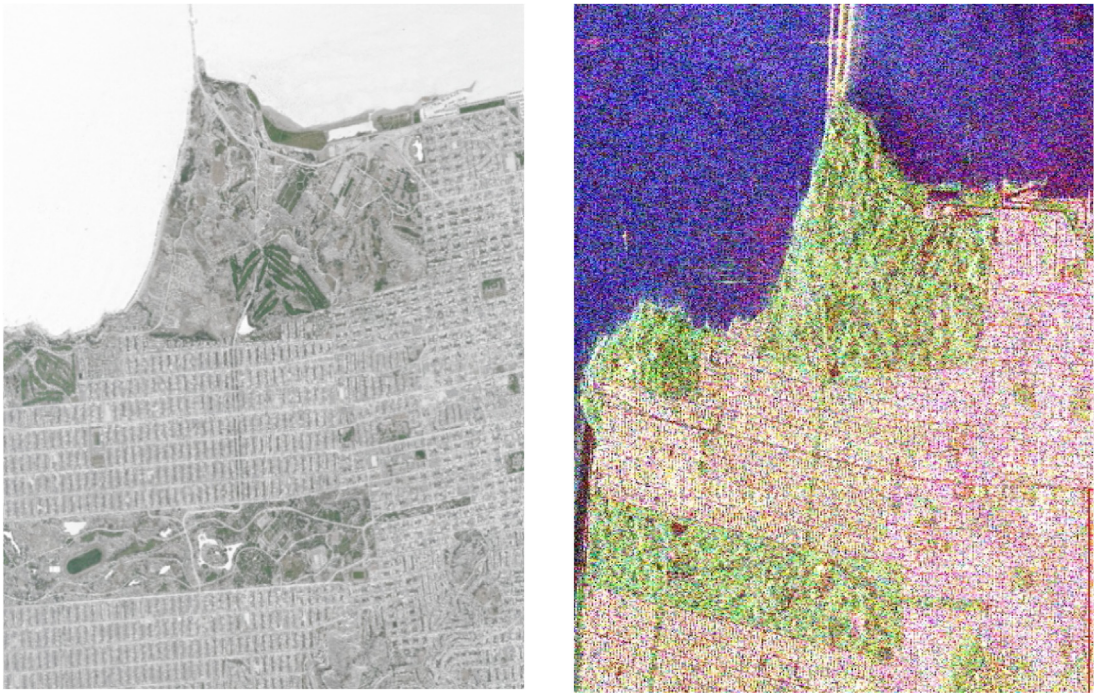
$$Accuracy = \frac{(TP + TN)}{(TP + FN + FP + TN)} \tag{34}$$

where TP is true positive, FN is false negative, TN is true negative, and FP is false positive. As can be seen, OA-DCNN shows the highest efficiency on training and validation datasets. OA-DCNN achieved 96.92% and 96.16% of accuracy in the test and train datasets, respectively. The sensitivity of OA is only lower than CapSA on the training dataset. As shown in Table 4, the "Average RunTime" of the OA-DCNN and CapSA-DCNN is less than other algorithms.

Figs. 25 and 26 indicate the comparison of algorithms in these metrics. According to these figures, the rank of the algorithms are: OA-DCNN, CapSA-DCNN, ChOA-DCNN, BWO-DCNN, PSO-DCNN, and GA-DCNN, respectively. The results of OA algorithm in the test and train dataset show that the proposed feature selection method performs well. Because specificity, accuracy, and sensitivity of OA algorithm are highly stable.

4. Conclusion and future works

In this paper, the Orchard Algorithm (OA) was introduced, inspired by the creation and maintenance of an orchard. In this phenomenon, planting seedlings and performing maintenance operations such as irrigation, trimming, grafting, and planting new seedlings instead of weak ones lead to creating an orchard with grown, strong, fruitful trees. The salient features of this phenomenon used in the OA are the annual growth of trees, the screening of trees



a) high resolution image b) Pauli RGB image

Fig. 24. POLSAR images dataset.

Table 18
Comparison of the results of algorithms for feature selection problem.

Proposed algorithms	Training dataset			Validation dataset			Run time (s)
	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy	
OA-DCNN	97.41%	94.63%	96.92%	96.77%	94.31%	96.16%	896
CapSA-DCNN	97.52%	93.76%	96.48%	95.63%	93.18%	95.51%	874
ChOA-DCNN	96.61	92.51%	95.32%	94.84	92.08	94.29%	956
BWO-DCNN	95.21%	91.88%	94.83%	93.36%	90.61%	92.88%	1021
PSO-DCNN	94.32%	90.33%	93.26%	92.91%	89.39%	92.26%	1126
GA-DCNN	93.76%	89.49%	92.83%	92.46%	88.81%	91.68%	1109

by the gardener periodically, the grafting of some trees with strong ones, and finally, the cutting of weak trees and replacing them with new seedlings.

The developed algorithm was tested on CEC2005 test functions, IEEE CEC06-2019 test functions, and 30 benchmark functions compared to 13 widely used and competitive algorithms. For each function, the algorithms were run 30 times. The best value of the fitness function (Best), the mean of the fitness function (Mean), the standard deviation of the fitness function (StdDev), Wilcoxon's test, and the average execution time of the algorithms (AvgTime) for all algorithms, were compared. According to the results, the OA had achieved good results in most executions and had performed better than other algorithms.

In CEC2005 benchmark functions, OA achieved the best results (Avg) in 17 test functions. Given the p-values of algorithms, OA can achieve significant results compared to other algorithms. In IEEE CEC06-2019 test functions, OA shows the best result in 9 test functions. CapSA and ChOA show the best result in 7 and 6 problems, respectively. GA got the worst results. In 30 benchmark functions, the OA had the best value of the "Best" criterion in 66.66% of the benchmark functions among all algorithms. Likewise, OA had the best "Mean" value in 51.51% of the benchmark functions compared to another algorithm. In the mentioned evaluation, in 72.72% of the benchmark functions, the OA had the lowest value of "AvgTime" compared to the other 8 algorithms. Also, according to the

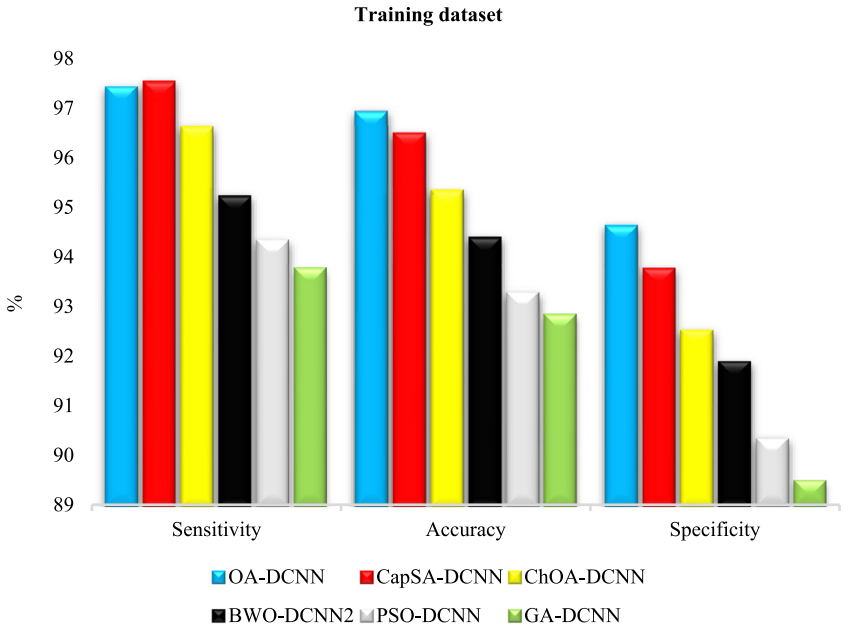


Fig. 25. Comparison of algorithms in training dataset.

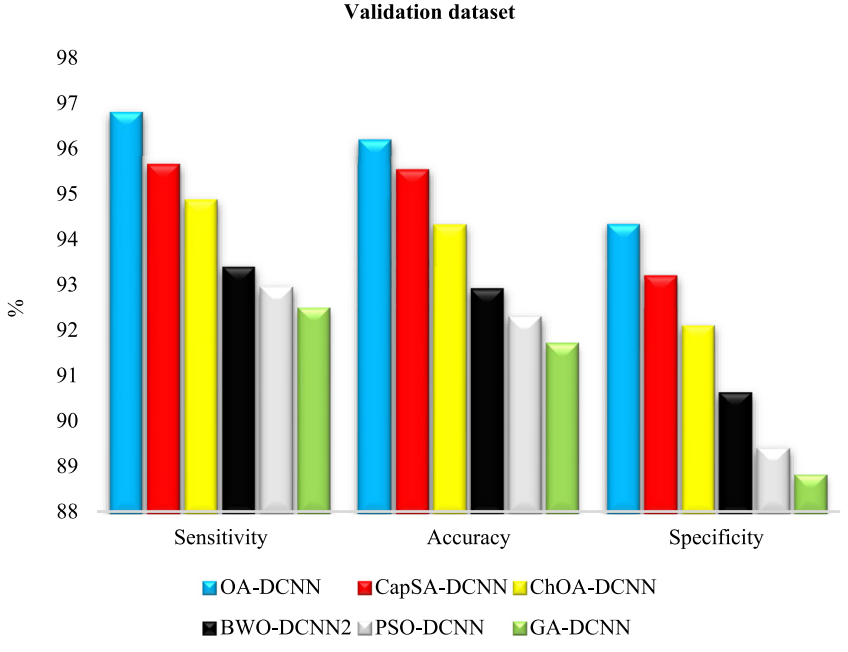


Fig. 26. Comparison of algorithms in validation dataset.

standard deviation values reported in Table 6, OA had the best levels of stability and repeatability compared to the other 8 algorithms.

After showing the OA's capability to solve benchmark problems, the algorithm was evaluated and tested on four engineering problems. The OA results were compared with those of some other meta-heuristic algorithms. The performance of the OA was better than the other algorithms. According to the OA's convergence diagrams for benchmark and engineering problems, the convergence speed of the OA was very high. This was also confirmed by

the OA execution time. The main reason for this is the definition of effective operators, such as growth, screening, and grafting. The OA finally achieves a set of optimal solutions. In fact, after a few iterations, the seedlings had grown into an orchard with fruitful trees. In addition, according to the concept and operators of the OA, this algorithm can solve both continuous and discrete problems. Like most meta-heuristic algorithms, there are many operators in OA. Therefore, modeling these operators for real applications can be a challenge for a designer. In OA, some operators have a conceptual state and there is no specific mathematical model (like growth operator). Therefore, modeling this operator for real applications can be a challenge. Accurate setting of initial parameters of OA is a limitation and special methods should be used. Another challenge in real applications is the computational complexity and computing time of OA algorithm. For example, in one iteration of the OA, several neighborhood searches are performed for each seedling in the growth operator, and a large number of seedlings are generated. Calculating the fitness function of all solutions and choosing the best solution causes computational complexity. Therefore, high computing power is usually required to solve a real-world application.

Several research directions can be recommended for future works. Multi-objective, modify, and binary variants of the OA may be extended to tackle various, multi-objective, discrete, and real-world optimization problems. The multiple engineering applications of OA in the real world will be a curious area of research. Also, a few thresholds and selective parameters in the OA's equations have not been optimally fine-tuned, making it a new path for further work in the future and hybrid algorithms will improve the operators. As well, tackling problems in different fields (i.e., deep learning, neural networks, image processing, scheduling, data mining, big data, smart home, industry, etc.) could be a valuable contribution and beneficial.

CRediT authorship contribution statement

Mehrdad Kaveh: Conceptualization, Investigation, Methodology, Software, Writing – original draft,Writing – review & editing. **Mohammad Saadi Mesgari:** Conceptualization, Validation, Supervision, Writing – review & editing. **Bahram Saeidian:** Conceptualization, Data management, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Funding information

There is no funding information.

Ethical approval

This paper does not contain any studies with human participants or animals.

Informed consent

As this paper does not contain any studies with human participants or animals, the informed consent is not applicable.

References

[1] B. Abdollahzadeh, F.S. Gharehchopogh, S. Mirjalili, African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems, Comput. Ind. Eng. 158 (2021) 107408.
[2] A.M. Adrian, A. Utamima, K.J. Wang, A comparative study of GA, PSO and ACO for solving construction site layout optimization, KSCE J. Civ. Eng. 19 (3) (2015) 520–527.
[3] M.P. Aghababa, M.H. Amrollahi, M. Borjkhani, Application of GA, PSO, and ACO algorithms to path planning of autonomous underwater vehicles, J. Mar. Sci. Appl. 11 (3) (2012) 378–386.
[4] R. Alberdi, K. Khandelwal, Comparison of robustness of metaheuristic algorithms for steel frame optimization, Eng. Struct. 102 (2015) 40–60.
[5] M.M. Ali, C. Khompatraporn, Z.B. Zabinsky, A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems, J. Global Optim. 31 (4) (2005) 635–672.
[6] M. Alimoradi, H. Azgomi, A. Asghari, Trees social relations optimization algorithm: A new Swarm-Based metaheuristic technique to solve continuous and discrete optimization problems, Math. Comput. Simulation 194 (2022) 629–664.
[7] A. Alsheddy, Empowerment Scheduling: A Multi-Objective Optimization Approach using Guided Local Search (Doctoral dissertation), University of Essex, 2011.
[8] E. Atashpaz-Gargari, C. Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, in: 2007 IEEE Congress on Evolutionary Computation, 2007, pp. 4661–4667.
[9] M. Azizi, S. Talatahari, A.H. Gandomi, Fire Hawk Optimizer: A novel metaheuristic algorithm, Artif. Intell. Rev. (2022) 1–77.
[10] S. Baniasadi, O. Rostami, D. Martín, M. Kaveh, A novel deep supervised learning-based approach for intrusion detection in IoT systems, Sensors 22 (12) (2022) 4459.
[11] R. Battiti, M. Brunato, A. Mariello, Reactive search optimization: learning while optimizing, in: Handbook of Metaheuristics, 2019, pp. 479–511.
[12] A. Baykasoğlu, Ş. Akpinar, Weighted Superposition Attraction (WSA): A swarm intelligence algorithm for optimization problems–Part 1: Unconstrained optimization, Appl. Soft Comput. 56 (2017) 520–540.
[13] S.I. Bejinariu, H. Costin, A comparison of some nature-inspired optimization metaheuristics applied in biomedical image registration, Methods Inf. Med. 57 (05/06) (2018) 280–286.
[14] M. Braik, A. Sheta, H. Al-Hiary, A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm, Neural Comput. Appl. 33 (7) (2021) 2515–2547.
[15] W. Cai, W. Yang, X. Chen, A global optimization algorithm based on plant growth theory: plant growth optimization, in: 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA), Vol. 1, 2008, pp. 1194–1199.
[16] A. Carberry, How to graft a tree, 2019, Available from: https://www.wikihow.com/Graft-a-Tree.
[17] A. Carberry, How to plant fruit trees, 2019, Available from: https://www.wikihow.com/Plant-Fruit-Trees.
[18] J. Carson, G. Shimizu, C. Ingels, P.M. Geisel, C.L. Unruh, Fruit trees: Planting and care of young trees, 2002.
[19] M.Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112.
[20] A. Cheraghalipour, M. Hajiaghaei-Keshteli, M.M. Paydar, Tree Growth Algorithm (TGA): A novel approach for solving optimization problems, Eng. Appl. Artif. Intell. 72 (2018) 393–414.
[21] S. Chetty, A.O. Adewumi, Three new stochastic local search algorithms for continuous optimization problems, Comput. Optim. Appl. 56 (3) (2013) 675–721.
[22] P. Civicioglu, Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm, Comput. Geosci. 46 (2012) 229–247.
[23] P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems, Appl. Math. Comput. 219 (15) (2013) 8121–8144.
[24] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41.
[25] B. Doğan, T. Ölmez, A new metaheuristic for numerical function optimization: Vortex Search algorithm, Inform. Sci. 293 (2015) 125–145.
[26] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43.
[27] A. Ebrahimi, E. Khamehchi, Sperm whale algorithm: an effective metaheuristic algorithm for production optimization problems, J. Nat. Gas Sci. Eng. 29 (2016) 211–222.
[28] E. Elbeltagi, T. Hegazy, D. Grierson, Comparison among five evolutionary-based optimization algorithms, Adv. Eng. Inform. 19 (1) (2005) 43–53.
[29] N. Eslami, S. Yazdani, M. Mirzaei, E. Hadavandi, Aphid-Ant Mutualism: A novel nature-inspired metaheuristic algorithm for solving optimization problems, Math. Comput. Simulation (2022).
[30] A.E. Ezugwu, O.J. Adeleke, A.A. Akinyelu, S. Viriri, A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems, Neural Comput. Appl. 32 (10) (2020) 6207–6251.
[31] J.D. Farmer, N.H. Packard, A.S. Perelson, The immune system, adaptation, and machine learning, Physica D 22 (1–3) (1986) 187–204.
[32] L.J. Fogel, A.J. Owens, M.J. Walsh, Intelligent decision making through a simulation of evolution, Behav. Sci. 11 (4) (1966) 253–272.
[33] A.H. Gandomi, Interior search algorithm (ISA): a novel approach for global optimization, ISA Trans. 53 (4) (2014) 1168–1183.
[34] A.H. Gandomi, A.H. Alavi, Krill herd: a new bio-inspired optimization algorithm, Commun. Nonlinear Sci. Numer. Simul. 17 (12) (2012) 4831–4845.

[35] G.R.J. Garner, The Grafter's Handbook, Chelsea Green Publishing, 2013.
[36] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, Simulation 76 (2) (2001) 60–68.
[37] F.S. Gharehchopogh, B. Abdollahzadeh, An efficient harris hawk optimization algorithm for solving the travelling salesman problem, Cluster Comput. 25 (3) (2022) 1981–2005.
[38] F.S. Gharehchopogh, B. Farnad, A. Alizadeh, A modified farmland fertility algorithm for solving constrained engineering problems, Concurr. Comput.: Pract. Exper. 33 (17) (2021) e6310.
[39] F. Glover, Heuristics for integer programming using surrogate constraints, Decis. Sci. 8 (1) (1977) 156–166.
[40] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (5) (1986) 533–549.
[41] M.J. Goldanloo, F.S. Gharehchopogh, A hybrid OBL-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems, J. Supercomput. 78 (3) (2022) 3998–4031.
[42] E.E. Goldschmidt, Plant grafting: new mechanisms, evolutionary implications, Front. Plant Sci. 5 (2014) 727.
[43] O.B. Haddad, A. Afshar, M.A. Mariño, Honey-bees mating optimization (HBMO) algorithm: a new heuristic approach for water resources optimization, Water Resour. Manag. 20 (5) (2006) 661–680.
[44] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001) 159–195.
[45] O. Hanseth, M. Aanestad, Bootstrapping networks, communities and infrastructures. On the evolution of ICT solutions in heath care, in: Proceedings of the 1st International Conference on Information Technology in Health Care (ITHC'01), 2001.
[46] F.A. Hashim, E.H. Houssein, K. Hussain, M.S. Mabrouk, W. Al-Atabany, Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems, Math. Comput. Simulation 192 (2022) 84–110.
[47] W.K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, 1970.
[48] A. Hatamlou, Black hole: A new heuristic optimization approach for data clustering, Inform. Sci. 222 (2013) 175–184.
[49] V. Hayyolalam, A.A.P. Kazem, Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems, Eng. Appl. Artif. Intell. 87 (2020) 103249.
[50] S. He, Q.H. Wu, J.R. Saunders, Group search optimizer: an optimization algorithm inspired by animal searching behavior, IEEE Trans. Evol. Comput. 13 (5) (2009) 973–990.
[51] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872.
[52] H. Holland John, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
[53] R. Hooke, T.A. Jeeves, Direct SearchSolution of numerical and statistical problems, J. ACM 8 (2) (1961) 212–229.
[54] B.E. Humphrey, The Bench Grafter's Handbook: Principles & Practice, CRC Press, 2019.
[55] S. Kadioglu, M. Sellmann, Dialectic search, in: International Conference on Principles and Practice of Constraint Programming, 2009, pp. 486–500.
[56] W. Kaidi, M. Khishe, M. Mohammadi, Dynamic levy flight chimp optimization, Knowl.-Based Syst. 235 (2022) 107625.
[57] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Vol. 200, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005, pp. 1–10.
[58] A. Karci, Human being properties of saplings growing up algorithm, in: 2007 IEEE International Conference on Computational Cybernetics, 2007, pp. 227–232.
[59] A.H. Kashan, League championship algorithm: a new algorithm for numerical function optimization, in: 2009 International Conference of Soft Computing and Pattern Recognition, 2009, pp. 43–48.
[60] A.H. Kashan, A new metaheuristic for optimization: optics inspired optimization (OIO), Comput. Oper. Res. 55 (2015) 99–125.
[61] A.H. Kashan, A.A. Akbari, B. Ostadi, Grouping evolution strategies: An effective approach for grouping problems, Appl. Math. Model. 39 (9) (2015) 2703–2720.
[62] S. Kaur, L.K. Awasthi, A.L. Sangal, G. Dhiman, Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization, Eng. Appl. Artif. Intell. 90 (2020) 103541.
[63] M. Kaveh, M. Kaveh, M.S. Mesgari, R.S. Paland, Multiple criteria decision-making for hospital location–allocation based on improved genetic algorithm, Appl. Geomat. 12 (3) (2020) 291–306.
[64] M. Kaveh, M. Khishe, M.R. Mosavi, Design and implementation of a neighborhood search biogeography-based optimization trainer for classifying sonar dataset using multi-layer perceptron neural network, Analog Integr. Circuits Signal Process. 100 (2) (2019) 405–428.
[65] A. Kaveh, V.R. Mahdavi, Colliding Bodies Optimization: Extensions and Applications, 2015.
[66] M. Kaveh, M.S. Mesgari, Hospital site selection using hybrid PSO algorithm-Case study: District 2 of Tehran, Sci.-Res. Q. Geogr. Data (SEPEHR) 28 (111) (2019) 7–22.
[67] M. Kaveh, M.S. Mesgari, Improved biogeography-based optimization using migration process adjustment: An approach for location–allocation of ambulances, Comput. Ind. Eng. 135 (2019) 800–813.
[68] M. Kaveh, M.S. Mesgari, Application of meta-heuristic algorithms for training neural networks and deep learning architectures: A comprehensive review, Neural Process. Lett. (2022) 1–104.
[69] M. Kaveh, M.S. Mesgari, A. Khosravi, Solving the local positioning problem using a four-layer artificial neural network, 7 (4) (2020) 21-40.
[70] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291–307.
[71] M. Khishe, M.R. Mosavi, Chimp optimization algorithm, Expert Syst. Appl. 149 (2020) 113338.
[72] N. Kianfar, M.S. Mesgari, A. Mollalo, M. Kaveh, Spatio-temporal modeling of COVID-19 prevalence and mortality using artificial neural network algorithms, Spat. Spatio-Temporal Epidemiol. 40 (2022) 100471.
[73] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.
[74] J.R. Koza, Genetic programming as a means for programming computers by natural selection, Stat. Comput. 4 (2) (1994) 87–112.

[75] K.N. Krishnanand, D. Ghose, Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications, Multiagent Grid Syst. 2 (3) (2006) 209–222.
[76] Y. Labbi, D.B. Attous, H.A. Gabbar, B. Mahdad, A. Zidan, A new rooted tree optimization algorithm for economic dispatch with valve-point effect, Int. J. Electr. Power Energy Syst. 79 (2016) 298–311.
[77] P. Larrañaga, J.A. Lozano (Eds.), Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Vol. 2, Springer Science & Business Media, 2001.
[78] Q.Q. Li, K. Song, Z.C. He, E. Li, A.G. Cheng, T. Chen, The artificial tree (AT) algorithm, Eng. Appl. Artif. Intell. 65 (2017) 99–110.
[79] Y.C. Liang, J.R. Cuevas Juarez, A novel metaheuristic for continuous optimization problems: Virus optimization algorithm, Eng. Optim. 48 (1) (2016) 73–93.
[80] J. Matyas, Random optimization, Autom. Remote Control 26 (2) (1965) 246–253.
[81] A.R. Mehrabian, C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, Ecol. Inform. 1 (4) (2006) 355–366.
[82] F. Merrikh-Bayat, The runner-root algorithm: a metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature, Appl. Soft Comput. 33 (2015) 292–303.
[83] H. Mirhajianmoghadam, M.R. Akbarzadeh-T, E. Lotfi, A harmonic emotional neural network for non-linear system identification, in: 2016 24th Iranian Conference on Electrical Engineering, ICEE, 2016, pp. 1260–1265.
[84] H. Mirhajianmoghadam, S.M. Ghasemi, Efficient parameter selection for scaled trust-region Newton algorithm in solving bound-constrained nonlinear systems, 2020, arXiv preprint arXiv:2009.04354.
[85] S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Comput. Appl. 27 (4) (2016) 1053–1073.
[86] S. Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133.
[87] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191.
[88] N. Mladenović, P. Hansen, Variable neighborhood search, Comput. Oper. Res. 24 (11) (1997) 1097–1100.
[89] A. Mortazavi, V. Toğan, A. Nuhoğlu, Interactive search algorithm: a new hybrid metaheuristic optimization algorithm, Eng. Appl. Artif. Intell. 71 (2018) 275–292.
[90] P. Moscato, On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms, Caltech concurrent computation program, 1989, C3P Report, 826.
[91] A. Mucherino, O. Seref, Monkey search: a novel metaheuristic search for global optimization, in: AIP Conference Proceedings, Vol. 953, American Institute of Physics, 2007, pp. 162–173, (1).
[92] K. Mudge, J. Janick, S. Scofield, E.E. Goldschmidt, A history of grafting, 35 (9) (2009).
[93] H. Murase, Finite element inverse analysis using a photosynthetic algorithm, Comput. Electron. Agric. 29 (1–2) (2000) 115–123.
[94] M. Najarian, Z. Sarmast, S.M. Ghasemi, S. Sarmadi, Evolutionary vertical size reduction: A novel approach for big data computing, Int. J. Math. Appl. 6 (3) (2018) 215–225.
[95] S. Nakrani, C. Tovey, On honey bees and dynamic server allocation in internet hosting centers, Adapt. Behav. 12 (3–4) (2004) 223–240.
[96] J. Olsen, W. Valley, A. Nina Azarenko, Growing Tree Fruits, Master Gardener Publications, 2012.
[97] J.S. Pan, L.G. Zhang, R.B. Wang, V. Snášel, S.C. Chu, Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems, Math. Comput. Simulation (2022).
[98] M.L. Parker, Producing Tree Fruit for Home Use, AG (USA), 1993.
[99] D.T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, M. Zaidi, The bees algorithm—a novel tool for complex optimisation problems, in: Intelligent Production Machines and Systems, 2006, pp. 454–459.
[100] U. Premaratne, J. Samarabandu, T. Sidhu, A new biologically inspired optimization algorithm, in: 2009 International Conference on Industrial and Information Systems, ICIIS, 2009, pp. 279–284.
[101] J. Puchinger, G.R. Raidl, Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification, in: International Work-Conference on the Interplay Between Natural and Artificial Computation, 2005, pp. 41–53.
[102] X. Qi, Y. Zhu, H. Chen, D. Zhang, B. Niu, An idea based on plant root growth for numerical optimization, in: International Conference on Intelligent Computing, 2013, pp. 571–578.
[103] R. Rajabioun, Cuckoo optimization algorithm, Appl. Soft Comput. 11 (8) (2011) 5508–5518.
[104] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: a gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248.
[105] L.A. Rastrigin, The convergence of the random search method in the extremal control of a many parameter system, Autom. Remote Control 24 (1963) 1337–1342.
[106] N. Rodríguez, A. Gupta, P.L. Zabala, G. Cabrera-Guerrero, Optimization algorithms combining (meta) heuristics and mathematical programming and its application in engineering, Math. Probl. Eng. (2018).
[107] O. Rostami, M. Kaveh, Optimal feature selection for SAR image classification using biogeography-based optimization (BBO), artificial bee colony (ABC) and support vector machine (SVM): a combined approach of optimization and machine learning, Comput. Geosci. 25 (3) (2021) 911–930.
[108] R.Y. Rubinstein, Optimization of computer simulation models with rare events, European J. Oper. Res. 99 (1) (1997) 89–112.
[109] F. Sadeghi, O. Rostami, M.K. Yi, S.O. Hwang, A deep learning approach for detecting Covid-19 using the chest X-ray images, Cmc-Comput. Mater. Contin. 74 (1) (2023) 751–768.
[110] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl. Soft Comput. 13 (5) (2013) 2592–2612.

[111] B. Saeidian, M.S. Mesgari, M. Ghodousi, Optimum allocation of water to the cultivation farms using Genetic Algorithm, Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 40 (1) (2015) 631.
[112] B. Saeidian, M.S. Mesgari, M. Ghodousi, Evaluation and comparison of Genetic Algorithm and Bees Algorithm for location–allocation of earthquake relief centers, Int. J. Disaster Risk Reduct. 15 (2016) 94–107.
[113] B. Saeidian, M.S. Mesgari, B. Pradhan, A.M. Alamri, Irrigation water allocation at farm level based on temporal cultivation-related data using meta-heuristic optimisation algorithms, Water 11 (12) (2019) 2611.
[114] B. Saeidian, M.S. Mesgari, B. Pradhan, M. Ghodousi, Optimized location–allocation of earthquake relief centers using PSO and ACO, complemented by GIS, clustering, and TOPSIS, ISPRS Int. J. Geo-Inf. 7 (8) (2018) 292.
[115] A. Salhi, E.S. Fraga, Nature-inspired optimisation approaches and the new plant propagation algorithm, 2011.
[116] H. Salimi, Stochastic fractal search: a powerful metaheuristic algorithm, Knowl.-Based Syst. 75 (2015) 1–18.
[117] H. Shah-Hosseini, The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm, Int. J. Bio-Inspired Comput. 1 (1–2) (2009) 71–79.
[118] H. Shah-Hosseini, Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation, Int. J. Comput. Sci. Eng. 6 (1–2) (2011) 132–140.
[119] H. Shayanfar, F.S. Gharehchopogh, Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems, Appl. Soft Comput. 71 (2018) 728–746.
[120] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[121] N. Srinivas, K. Deb, Muiltiobjective optimization using nondominated sorting in genetic algorithms, Evol. Comput. 2 (3) (1994) 221–248.
[122] R.S. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[123] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, KanGAL report, 2005, 2005.
[124] S. Talatahari, M. Azizi, M. Tolouei, B. Talatahari, P. Sareh, Crystal Structure Algorithm (CryStAl): A metaheuristic optimization method, IEEE Access 9 (2021) 71244–71261.
[125] K. Tamura, K. Yasuda, Spiral dynamics inspired optimization, J. Adv. Comput. Intell. Intell. Inform. 15 (8) (2011) 1116–1122.
[126] D. Teodorović, Bee colony optimization (BCO), in: Innovations in Swarm Intelligence, 2009, pp. 39–60.
[127] C. Wang, H.Z. Cheng, Z.C. Hu, Y. Wang, Distribution system optimization planning based on plant growth simulation algorithm, J. Shanghai Jiaotong Univ. (Science) 13 (4) (2008) 462–467.
[128] G.G. Wang, S. Deb, L.D.S. Coelho, Elephant herding optimization, in: 2015 3rd International Symposium on Computational and Business Intelligence, ISCBI, 2015, pp. 1–5.
[129] J. Wang, M. Khishe, M. Kaveh, H. Mohammadi, Binary chimp optimization algorithm (BChOA): A new binary meta-heuristic for solving optimization problems, Cogn. Comput. 13 (5) (2021) 1297–1316.
[130] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, J. Schmidhuber, Natural evolution strategies, J. Mach. Learn. Res. 15 (1) (2014) 949–980.
[131] X.S. Yang, Firefly algorithms for multimodal optimization, in: International Symposium on Stochastic Algorithms, 2009, pp. 169–178.
[132] X.S. Yang, A new metaheuristic bat-inspired algorithm, in: Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), 2010, pp. 65–74.
[133] X.S. Yang, Flower pollination algorithm for global optimization, in: International Conference on Unconventional Computing and Natural Computation, 2012, pp. 240–249.
[134] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), 2009, pp. 210–214.
[135] H. Yapici, N. Cetinkaya, A new meta-heuristic optimizer: pathfinder algorithm, Appl. Soft Comput. 78 (2019) 545–568.
[136] H. Zhang, Y. Zhu, H. Chen, Root growth model: a novel approach to numerical function optimization and simulation of plant root system, Soft Comput. 18 (3) (2014) 521–537.
[137] Z. Zhao, Z. Cui, J. Zeng, X. Yue, Artificial plant optimization algorithm for constrained optimization problems, in: 2011 S International Conference on Innovations in Bio-Inspired Computing and Applications, 2011, pp. 120–123.
[138] Y.J. Zheng, Water wave optimization: a new nature-inspired metaheuristic, Comput. Oper. Res. 55 (2015) 1–11.
[139] Y.J. Zheng, S.Y. Chen, H.F. Ling, Evolutionary optimization for disaster relief operations: A survey, Appl. Soft Comput. 27 (2015) 553–566.
[140] Y.Z. Zhou, Y. Wang, X. Chen, L. Zhang, K. Wu, A novel path planning algorithm based on plant growth mechanism, Soft Comput. 21 (2) (2017) 435–445.