IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 April 2023, accepted 28 May 2023, date of publication 6 June 2023, date of current version 13 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283422

RESEARCH ARTICLE

Red Panda Optimization Algorithm: An Effective Bio-Inspired Metaheuristic Algorithm for Solving Engineering Optimization Problems

HADI GIVI[1], (Member, IEEE), MOHAMMAD DEHGHANI[2], AND ŠTĚPÁN HUBÁLOVSKÝ[3]
[1]Department of Electrical Engineering, Shahreza Campus, University of Isfahan, Iran
[2]Department of Mathematics, Faculty of Science, University of Hradec Králové, 50003 Hradec Králové, Czech Republic
[3]Department of Applied Cybernetics, University of Hradec Králové, 50003 Hradec Králové, Czech Republic

Corresponding author: Hadi Givi (h.givi@shr.ui.ac.ir)

This work was supported by the Excellence project of Faculty of Science, University of Hradec Králové, Czech Republic, 2214/2023-2024.

ABSTRACT This paper presents a new bio-inspired metaheuristic algorithm called Red Panda Optimization (RPO) that imitates the natural behaviors of red pandas in nature. The main design idea of RPO is derived from two characteristic natural behaviors of red pandas: (i) foraging strategy, and (ii) climbing trees to rest. The proposed RPO approach is mathematically modeled in two phases of exploration based on the simulation of red pandas' foraging strategy and exploitation based on the simulation of red pandas' movement in climbing trees. The main advantage of the proposed approach is that there is no control parameter in its mathematical modeling, and for this reason, it does not need a parameter adjustment process. The performance of RPO is evaluated on fifty-two standard benchmark functions including unimodal, high-dimensional multimodal, and fixed-dimensional multimodal types as well as CEC 2017 test suite. The optimization results obtained by the proposed RPO approach are compared with the performance of twelve well-known metaheuristic algorithms. The simulation results show that RPO, by maintaining the balance between exploration and exploitation, is effective in solving optimization problems and its performance is superior over competitor algorithms. Based on the analysis of the optimization results, RPO has provided more successful performance compared to the competitor algorithms in 100% of unimodal functions, 100% of high-dimensional multimodal functions, 100% of fixed-dimensional multimodal functions, and 86.2% of CEC 2017 test suite benchmark functions. Also, the statistical analysis of the Wilcoxon rank sum test shows that the superiority of RPO in the competition with the compared algorithms is significant from a statistical point of view. In addition, the results of implementing RPO on four engineering design problems confirms the ability of the proposed approach to handle real-world optimization applications.

INDEX TERMS Optimization, bio-inspired, red panda, metaheuristic, exploration, exploitation.

I. INTRODUCTION

Optimization problems are a type of problems that have more than one feasible solution. According to this definition, optimization is the process of finding the best feasible solution among the available solutions for a problem [1]. From a mathematical point of view, an optimization problem can be modeled considering three main parts: decision variables, constraints, and objective function. The main goal in optimization is to set values for decision variables such that the objective function is optimized according to the constraints of the problem [2]. There are numerous optimization problems in science that have become more complex with the advancement of technology, and this is the reason for the need to powerful tools for solving optimization problems [3].

Problem solving methods in optimization studies are classified into two groups: deterministic and stochastic approaches [4]. Deterministic approaches are effective tools

The associate editor coordinating the review of this manuscript and approving it for publication was Xiong Luo.

for solving linear, convex, continuous, differentiable, and low-dimensional problems [5]. However, in case of more complex optimization problems, deterministic approaches lose their efficiency due to getting stuck in local optima. This is while many of today's optimization problems and real-world applications are non-linear, non-convex, discontinuous, non-differentiable, and high-dimensional [6]. These disadvantages and the inability of deterministic approaches to solve complex optimization problems have prompted researchers to develop stochastic approaches. Stochastic approaches, without the need for derivative information from the objective function and problem constraints, are able to provide suitable solutions for optimization problems based on the random search process in the problem-solving space [7]. Metaheuristic algorithms are one of the most effective stochastic approaches in solving optimization problems. Advantages such as simplicity of concepts, convenient implementation, no dependence on the type of problem, no need for derivative information, efficient performance in solving nonlinear, non-convex, high-dimensional, and NP-hard problems, as well as desirable efficiency in nonlinear and unknown search spaces are the main reasons for the popularity of metaheuristic algorithms [8].

The nature of random search in metaheuristic algorithms means that there is no guarantee of achieving the global optimal solution with these approaches. However, since the solutions obtained by metaheuristic algorithms are close to the global optima, they are acceptable and known as quasi-optimal solutions [9]. In order to organize an effective search process, metaheuristic algorithms must be able to scan the problem-solving space appropriately at both global and local levels. Global search with the concept of exploration leads to the ability of the algorithm to comprehensively search the problem-solving space with the aim of discovering the main optimal area and preventing the algorithm from getting stuck in local optima. Local search with the concept of exploitation leads to the ability of the algorithm to achieve possible better solutions near the discovered solutions [10]. In addition to exploration and exploitation abilities, what leads to the success of metaheuristic algorithms in the optimization process is the balancing of exploration and exploitation during the search process [11]. The efforts of researchers to achieve more effective solutions for optimization problems have led to the design of numerous metaheuristic algorithms [12]. These algorithms are employed in various optimization applications in science, such as energy [13], [14], [15], [16], protection [17], energy carriers [18], [19], and electrical engineering [20], [21], [22], [23], [24], [25].

The main research question in the study of metaheuristic algorithms is that considering various algorithms presented so far, is there still a need to design newer metaheuristic algorithms? In response to this question, No Free Lunch (NFL) theorem [26] explains that there is no specific metaheuristic algorithm to be considered as the best optimizer for all optimization problems. In fact, the optimal performance of an algorithm in solving a set of optimization problems is not a guarantee for the similar performance of that algorithm in solving other optimization problems. According to NFL theorem, a successful algorithm in solving an optimization problem may even fail in solving another problem. Therefore, there is no guarantee for the success or failure of implementing an algorithm on a problem. By keeping open the field of metaheuristic algorithms study, NFL theorem encourages researchers to provide more effective solutions to optimization problems by designing newer metaheuristic algorithms.

The innovation and novelty of this paper is the introduction and design of a new metaheuristic algorithm called Red Panda Optimization (RPO) to solve optimization problems. The main contributions of this paper are as follows:

- The proposed RPO approach is based on the simulation of red panda behaviors in nature.
- The fundamental inspiration for RPO design is the foraging strategy and tree climbing ability of red pandas.
- The mathematical model of RPO is presented in two phases of exploration and exploitation.
- The efficiency of RPO in optimization has been evaluated on fifty-two benchmark functions consisting of unimodal, high-dimensional multimodal, and fixed-dimensional multimodal types, as well as CEC 2017 test suite.
- The performance of RPO is compared with twelve well-known metaheuristic algorithms.
- The effectiveness of RPO in handling real-world applications is examined on four engineering design problems.

The rest of this article is organized in this way, first the literature review is presented in section II. Then, the proposed Red Panda Optimization (RPO) algorithm is introduced and mathematically modeled in section III. Simulation studies and results are presented in section IV. The performance of the proposed RPO in solving real-world applications is evaluated in section V. Finally, conclusions and several proposals for future studies are provided in section VI.

II. LECTURE REVIEW

Metaheuristic algorithms have been developed with inspiration from various natural phenomena, animal life in nature, biological sciences, physical laws and phenomena, rules of games, human interactions, and other evolutionary processes. Based on the idea used in the design, metaheuristic algorithms can be broadly classified into five groups: swarm-based, evolutionary-based, physics-based, human-based, and game-based approaches [27].

Swarm-based metaheuristic algorithms are developed based on simulating the natural swarm behavior of animals, birds, aquatic animals, insects, plants, and other living organisms in nature. Among the well-known algorithms of this group, one can mention Particle Swarm Optimization (PSO) [28], Ant Colony Optimization (ACO) [29], Artificial Bee Colony (ABC) [30], and Firefly Algorithm (FA) [31]. ACO was proposed based on modeling the ability of ant swarm to identify the shortest communication path between nests and food sources. ABC was designed based on simulating interactions and natural behaviors of colony bees in obtaining food resources. FA was inspired by the communication feature of flashing light in the firefly's swarm. Finding food resources, migration, and chasing are common natural behaviors among animals, whose simulation has inspired researchers to design several swarm-based algorithms such as: Coati Optimization Algorithm (COA) [32], Golden Jackal Optimization (GJO) [33], White Shark Optimizer (WSO) [34], Marine Predator Algorithm (MPA) [35], African Vultures Optimization Algorithm (AVOA) [36], Pelican Optimization Algorithm (POA) [37], Tunicate Swarm Algorithm (TSA) [38], Honey Badger Algorithm (HBA) [39], Whale Optimization Algorithm (WOA) [40], Reptile Search Algorithm (RSA) [41], Green Anaconda Optimization (GAO) [42], Cuckoo Search Algorithm (CSA) [43], and Grey Wolf Optimizer (GWO) [44].

Evolutionary-based metaheuristic algorithms are introduced based on the concepts of genetics, biology, natural selection, and survival of the fittest. Genetic Algorithm (GA) [45] and Differential Evolution (DE) [46] are widely used approaches in this group. GA and DE were developed based on reproductive process modeling, biology concepts and stochastic operators such as selection, crossover, and mutation. Artificial Immune Systems (AISs) is another evolutionary approach that has been introduced based on the ability of the human body's defense system against diseases and microbes [47]. Some other evolutionary-based metaheuristic algorithms are: Evolution Strategy (ES) [48], Genetic programming (GP) [49], and Cultural Algorithm (CA) [50].

Physics-based metaheuristic algorithms are designed based on simulating concepts, phenomena, and laws in physics. Simulated Annealing (SA) [51] is one of the most widely used physics-based approaches. SA was developed based on the modeling of metal annealing phenomenon in physics, where metals are melted under heat and then cooled in order to achieve ideal crystal. The modeling of physical forces has been the starting point for the introduction of several physics-based algorithms, such as: Spring Search Algorithm (SSA) [52], Momentum Search Algorithm (MSA) [53], and Gravitational Search Algorithm (GSA) [54]. SSA was introduced based on the simulation of Hooke's law, spring elastic force, and Newton's laws of motion in a system consisting of weights connected by springs. MSA was proposed based on the modeling of the force resulting from the momentum between the bullets. GSA was designed based on simulating the gravitational force that masses exert on each other at different distances. The physical phenomenon of matter state transitions for water was employed in design of the Water Cycle Algorithm (WCA) [55]. Some other physics-based metaheuristic algorithms are: Nuclear Reaction Optimization (NRO) [56], Lichtenberg Algorithm (LA) [57], Archimedes Optimization Algorithm (AOA) [58], Equilibrium Optimizer (EO) [59], Multi-Verse Optimizer (MVO) [60], and Electro-Magnetism Optimization (EMO) [61].

Human-based metaheuristic algorithms have been developed based on the simulation of human interactions, communication, thinking, and decision-making in social and individual lives. Teaching-Learning Based Optimization (TLBO) [62] is one of the most widely used human-based algorithms. The basic inspiration in its design was modelling the educational interactions of students and teachers in the classroom. The economic activities of both the poor and the rich sections of the society, who are trying to improve their economic conditions, have been a source of inspiration in the design of Poor and Rich Optimization (PRO) [63]. Therapeutic interactions between patients and physicians were employed in the design of Doctor and Patients Optimization (DPO) [64]. The cooperation between the members of a team who are trying to achieve the team goal was the basic idea in the design of Teamwork Optimization Algorithm (TOA) [65]. Some other human-based metaheuristic algorithms are: Ali Baba and the Forty Thieves (AFT) [66], Skill Optimization Algorithm (SOA) [67], Language Education Optimization (LEO) [68], Coronavirus Herd Immunity Optimizer (CHIO) [69], War Strategy Optimization (WSO) [70], and Driving Training-Based Optimization (DTBO) [71].

Game-based metaheuristic algorithms have been introduced based on modeling the rules of various individual and group games, the strategy of players, coaches, referees, and other influential persons of the games. Football Game Based Optimization (FGBO) [72] and Volleyball Premier League (VPL) [73] are two game-based approaches, which were designed based on the simulation of competitions between clubs in soccer and volleyball leagues. The skill of the players in putting together the pieces of the puzzle has been a source of inspiration in the design of Puzzle Optimization Algorithm (POA) [74]. The strategy of players in throwing darts and collecting points in the darts game was employed in the design of Darts Game Optimizer (DGO) [75]. Some other game-based metaheuristic algorithms are: Tug of War Optimization (TWO) [76], Billiards Optimization Algorithm (BOA) [77], Dice Game Optimization (DGO) [78], Ring Toss Game-Based Optimization (RTGBO) [79], Orientation Search Algorithm (OSA) [80], and Archery Algorithm (AA) [81].

Based on the best knowledge obtained from the literature review, no metaheuristic algorithm has been designed based on simulating the natural behavior of the red panda. Meanwhile, the behavior of foraging and resting on trees among red pandas are intelligent activities that has the potential to design a metaheuristic algorithm. In order to address this research gap in the studies of metaheuristic algorithms, in this paper, a new metaheuristic algorithm is introduced based on the mathematical modeling of the natural behavior of the red panda, which is discussed in the next section.

Algorithm 1 Pseudocode of RPO

Start RPO.
Input: The problem information (variables, objective function, and constraints).
Set RPO population size (N) and the total number of iterations (T).
Generate the initial population matrix at random using (1) and (2).
Evaluate the objective function by (3).
For $t = 1$ to T
For $i = 1$ to N
Phase 1: : The strategy of red pandas in foraging
Update food positions set for the ith RPO member using (4). $PFS_i \leftarrow \{X_k | k \in \{1, 2, \ldots, N\} \cap F_k < F_i\} \cup \{X_{best}\}$
Determine the selected food by the ith red panda at random.
Calculate new position of the ith RPO member based on the 1[st] phase of RPO using (5). $x_{i,j}^{P1} \leftarrow x_{i,j} + r \cdot (SFS_{i,j} - I \cdot x_{i,j})$
Update the ith RPO member using (6). $X_i \leftarrow \begin{cases} X_i^{P1}, & F_i^{P1} < F_i \\ X_i, & else \end{cases}$
Phase 2: Skill in climbing and resting on the tree
Calculate new position of the ith RPO member based on the 2[nd] phase of RPO using (7). $x_{i,j}^{P2} \leftarrow x_{i,j} + \frac{lb_j + r \cdot (ub_j - lb_j)}{t}$
Update the ith RPO member using (8). $X_i \leftarrow \begin{cases} X_i^{P2}, & F_i^{P2} < F_i \\ X_i, & else \end{cases}$
end
Save the best candidate solution found so far.
end
Output: The best solution obtained by RPO.
End RPO.

FIGURE 1. Photo of a red panda; downloaded from free media Wikimedia Commons.

III. RED PANDA OPTIMIZATION ALGORITHM

In this section, the proposed Red Panda Optimization (RPO) algorithm is introduced, then its mathematical modeling is presented.

A. INSPIRATION OF RPO

The red panda is a small endemic animal of the southern China and eastern Himalayas. It has dense reddish-brown hair on its body and legs, a black belly and legs, white-lined ears, a mainly white muzzle, and a ringed tail. It has a head-to-body length of 51-63.5 cm, a tail length of 28-48.5 cm, and weighs between 3.2 and 15 kg. Because of its flexible joints and curved semi-retractile claws, it is well adapted to climbing [82]. The red panda inhabits temperate broadleaf and mixed forests as well as coniferous forests, favoring steep slopes with dense bamboo cover close to water sources. It is largely arboreal and solitary [83]. A picture of the red panda is shown in Figure 1.

Red panda is largely herbivorous and eats mainly bamboo leaves and shoots, as well as blooms and fruits. The red panda has a good sense of sight, smell, and hearing and uses its long white whiskers to search for food at night [84]. According to observations, the red panda is a nocturnal animal. Due to its high ability to climb, it sleeps and rests in high places, especially trees during the day [85].

Among the natural behaviors of the red panda, its foraging strategy based on its high ability of hearing, sight, and smell, as well as the high skill of this animal in climbing trees, is much more impressive. Mathematical modeling of these natural behaviors of the red panda is the basis for the design of the proposed RPO approach, which is explained below.

B. MATHEMATICAL MODELLING

In this subsection, first the initialization of the proposed RPO approach is described, then based on the simulation of the natural behaviors of the red panda, the mathematical model of updating the candidate solutions in two phases of exploration and exploitation is presented.

1) INITIALIZATION

The proposed RPO approach is a population-based metaheuristic algorithm, whose members consist of red pandas. In RPO design, each red panda is a candidate solution to the problem, which suggests certain values for the problem variables based on its position in the search space. Therefore, from a mathematical point of view, each red panda (i.e., candidate solution) is modeled using a vector. Together, the red pandas of the algorithm population can be mathematically modeled using a matrix according to (1). Each row of this matrix represents a red panda (i.e., candidate

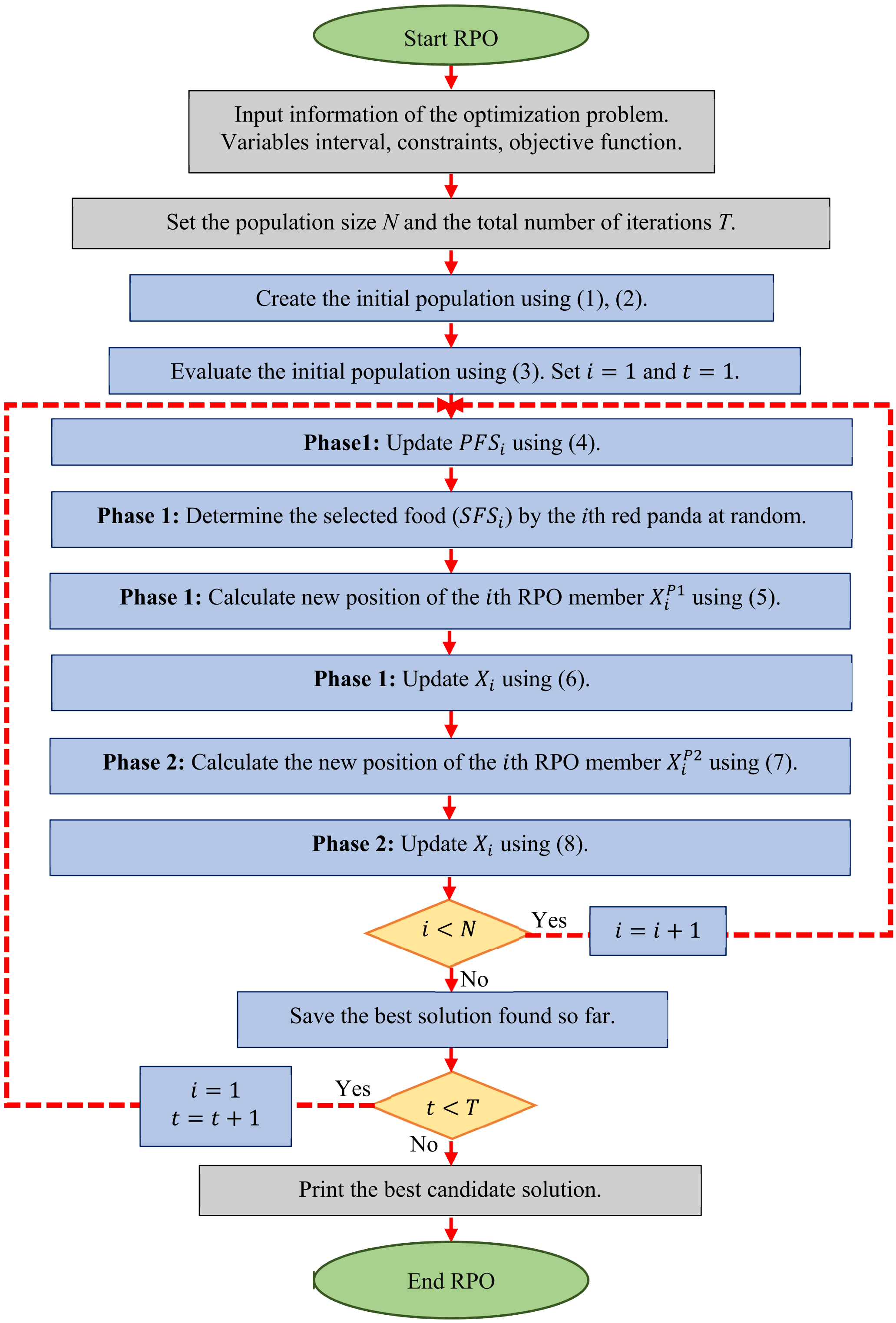

FIGURE 2. **Flowchart of RPO.**

solution) and each column of this matrix represents the suggested values for the corresponding variable of the given problem. At the beginning of RPO execution, the position of red pandas in the search space is randomly initialized

TABLE 1. Control parameter values for the competitor algorithms.

Algorithm	parameter	value
GA		
	Type	Real coded.
	Selection	Roulette wheel (Proportionate).
	Crossover	Whole arithmetic (Probability = 0.8, $\alpha \in [-0.5, 1.5]$).
	Mutation	Gaussian (Probability = 0.05).
PSO		
	Topology	Fully connected.
	Cognitive and social constant	$(C_1, C_2) = (2, 2)$.
	Inertia weight	Linear reduction from 0.9 to 0.1
	Velocity limit	10% of the dimension range.
GSA		
	Alpha, G_0, R_{norm}, R_{power}	20, 100, 2, 1
TLBO		
	T_F: the teaching factor	T_F = round $[(1 + rand)]$.
	random number $rand$	$rand$ is a random number from the interval [0,1].
GWO		
	Convergence parameter (a)	a: Linear reduction from 2 to 0.
MVO		
	wormhole existence probability (WEP)	$\text{Min}(WEP) = 0.2$ and $\text{Max}(WEP) = 1$.
	Exploitation accuracy over the iterations (p)	$p = 6$.
WOA		
	Convergence parameter a	a: Linear reduction from 2 to 0.
	Parameters r and l	r is a random vector in [0,1], l is a random number in $[-1,1]$.
TSA		
	P_{min} and P_{max}	1, 4
	c_1, c_2, c_3	random numbers lie in the range [0,1].
MPA		
	Constant number	$P = 0.5$
	Random vector	R is a vector of uniform random numbers from [0,1].
	Fish Aggregating Devices ($FADs$)	$FADs = 0.2$
	Binary vector	U = 0 or 1
RSA		
	Sensitive parameter	$\beta = 0.01$
	Sensitive parameter	$\alpha = 0.1$
	Evolutionary Sense (ES)	ES are randomly decreasing values between 2 and -2

TABLE 1. *(Continued.)* Control parameter values for the competitor algorithms.

AVOA		
	L_1, L_2	$(L_1, L_2) = (0.8, 0.2)$.
	w	$w = 2.5$
	P_1, P_2, P_3	$(P_1, P_2, P_3) = (0.6, 0.4, 0.6)$
WSO		
	F_{min} and F_{max}	$(F_{min}, F_{max}) = (0.07, 0.75)$.
	τ, a_0, a_1, a_2	$(\tau, a_0, a_1, a_2) = (4.125, 6.25, 100, 0.0005)$.

using (2).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix}_{N \times m} \quad (1)$$

$$x_{i,j} = lb_j + r_{i,j} \cdot \left(ub_j - lb_j\right), \quad i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, m, \quad (2)$$

where, X is the population matrix of red pandas' locations, X_i is the ith red panda (i.e., candidate solution), $x_{i,j}$ is its jth dimension (problem variable), N is the number of red pandas, m indicates the number of problem variables, $r_{i,j}$ are random numbers in the interval [0, 1], lb_j, and ub_j are the lower bound and upper bound of the jth problem variable, respectively.

Considering that the position of each red panda is a candidate solution for the problem, the objective function of the problem corresponding to each of these candidate solutions can be evaluated. The set of evaluated values for the objective function can be represented using a matrix according to (3).

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \quad (3)$$

where F is the objective function values vector and F_i denotes the value of the objective function obtained by the ith red panda.

The evaluated values for the objective function of the problem are the main criterion in determining the quality of the candidate solutions. In other words, the best obtained value

TABLE 2. Optimization results of the unimodal test functions.

		RPO	WSO	AVOA	RSA	MPA	TSA	WOA	MVO	GWO	TLBO	GSA	PSO	GA
F1	mean	0	323.1097	0	0.00E+00	9.29E-50	3.20E-47	5.80E-155	0.151165	4.55E-59	9.04E-75	1.15E-16	0.127076	34.63222
	best	0	72.27965	0	0.00E+00	7.49E-52	1.71E-49	6.00E-167	0.099666	1.72E-60	3.06E-77	4.90E-17	8.49E-05	16.62765
	worst	0	901.2379	0	0.00E+00	6.72E-49	2.09E-46	6.00E-154	0.247637	3.48E-58	7.49E-74	3.20E-16	1.90874	78.29486
	std	0	238.7674	0	0.00E+00	1.88E-49	6.32E-47	1.60E-154	0.042235	8.56E-59	2.06E-74	6.07E-17	0.424792	16.41014
	median	0	282.8347	0	0.00E+00	2.03E-50	3.66E-48	7.00E-161	0.135998	1.94E-59	8.83E-76	1.04E-16	0.004411	27.27738
	rank	1	11	1	1	5	6	2	9	4	3	7	8	10
F2	mean	0	3.776208	1.40E-283	0	1.04E-27	1.90E-28	6.40E-104	0.26047	1.64E-34	1.06E-38	5.37E-08	1.541729	2.942119
	best	0	0.756212	0	0	5.61E-30	1.02E-30	1.90E-114	0.179542	7.38E-36	2.62E-40	3.07E-08	0.048828	1.967443
	worst	0	8.958666	2.80E-282	0	3.43E-27	2.01E-27	1.20E-102	0.434829	6.42E-34	8.43E-38	9.17E-08	9.3403	5.171726
	std	0.00E+00	2.044082	0	0	1.21E-27	4.90E-28	2.70E-103	0.070258	2.04E-34	1.94E-38	1.56E-08	2.219601	0.732368
	median	0	3.136077	3.70E-297	0	6.28E-28	2.46E-29	3.40E-108	0.24991	7.85E-35	3.90E-39	5.17E-08	0.710123	2.762897
	rank	1	12	2	1	7	6	3	9	5	4	8	10	11
F3	mean	0	2636.013	0	0	1.33E-11	2.30E-11	22123.03	14.63423	2.41E-14	3.18E-25	499.7092	550.1655	2062.237
	best	0	1088.637	0	0	2.94E-20	8.92E-21	5630.916	6.012495	1.58E-19	1.81E-29	182.4369	36.57526	1157.779
	worst	0	5234.592	0	0	2.04E-10	3.05E-10	45040.6	26.20651	3.46E-13	3.35E-24	771.4986	5811.157	3532.252
	std	0	1107.562	0	0	4.56E-11	6.88E-11	9979.735	5.595845	7.92E-14	8.70E-25	148.1048	1256.767	648.7051
	median	0	2473.189	0	0	1.68E-14	1.47E-14	20656.93	11.89704	4.69E-16	8.95E-27	467.5755	248.454	2075.032
	rank	1	10	1	1	4	5	11	6	3	2	7	8	9
F4	mean	0	23.45023	4.60E-255	0	2.45E-19	0.00841	39.92337	0.589797	1.98E-14	2.74E-30	1.277679	6.85801	3.235365
	best	0	18.2143	0	0	5.06E-20	5.99E-05	0.10012	0.230471	8.70E-16	2.80E-31	1.51E-08	2.419381	2.007645
	worst	0	29.70858	9.20E-254	0	7.36E-19	0.039162	84.62575	0.978307	1.31E-13	1.45E-29	4.579257	9.763827	4.534911
	std	0	3.863479	0	0	1.84E-19	0.010586	30.0497	0.190986	3.05E-14	3.30E-30	1.120839	2.054112	0.65642
	median	0	23.18626	1.40E-284	0	1.93E-19	0.003723	38.0023	0.614103	8.45E-15	1.68E-30	0.992464	7.091329	3.274724
	rank	1	11	2	1	4	6	12	7	5	3	8	10	9
F5	mean	1.48E-09	28398.06	1.43E-05	5.798028	23.40807	28.60948	27.19269	456.4057	26.68578	26.85244	31.08724	105.5889	466.1947
	best	0	2267.064	2.18E-06	2.02E-28	22.82986	27.91037	26.43629	27.96275	25.24989	25.62028	25.42943	20.4739	170.799
	worst	2.67E-08	109106.7	4.26E-05	28.99019	24.03788	29.46094	28.73292	2504.324	28.51931	28.74462	91.45478	499.2191	987.7947
	std	5.97E-09	31180.39	1.12E-05	11.8973	0.393641	0.450018	0.604601	673.5282	0.826454	0.850991	15.68442	103.444	246.1823
	median	3.50E-18	20920.32	1.29E-05	3.74E-27	23.35764	28.8377	26.96467	132.3027	26.21564	26.7838	26.31103	84.06385	378.3332
	rank	1	13	2	3	4	8	7	11	5	6	9	10	12
F6	mean	0	343.063	4.50E-08	6.944537	1.54E-09	3.719839	0.065683	0.155721	0.721085	1.24652	1.30E-16	0.036419	31.48493
	best	0	57.61476	1.12E-08	4.57633	6.22E-10	2.816758	0.009389	0.108064	2.88E-05	0.72013	6.32E-17	5.38E-05	19.22687
	worst	0	1155.801	1.17E-07	7.250012	3.49E-09	4.55303	0.372513	0.211356	1.254424	1.934547	3.27E-16	0.287775	63.83688
	std	0	308.0894	2.43E-08	0.690273	8.01E-10	0.511655	0.088652	0.032372	0.345344	0.307855	5.98E-17	0.072692	11.37853
	median	0	265.7647	4.24E-08	7.250001	1.31E-09	3.796628	0.033701	0.148686	0.743237	1.183863	1.13E-16	0.003354	27.82404
	rank	1	13	4	11	3	10	6	7	8	9	2	5	12
F7	mean	2.16E-06	9.35E-05	3.77E-05	9.63E-05	0.000767	0.004149	0.001871	0.012721	0.000789	0.00154	0.058546	0.153672	0.010002
	best	1.89E-07	1.53E-06	1.82E-06	8.26E-06	0.000334	0.0021	5.32E-05	0.005047	0.000353	0.000295	0.025667	0.050064	0.00528
	worst	9.04E-06	0.000396	0.000138	0.000246	0.001661	0.007804	0.008107	0.0263	0.001578	0.003987	0.089886	0.31669	0.018763
	std	2.02E-06	0.000108	3.61E-05	6.97E-05	0.000444	0.001679	0.002254	0.005127	0.000346	0.001198	0.020316	0.057489	0.003841
	median	1.73E-06	6.40E-05	2.65E-05	9.10E-05	0.000582	0.003973	0.001069	0.011738	0.000757	0.001089	0.061168	0.152469	0.009263
	rank	1	3	2	4	5	9	8	11	6	7	12	13	10
Sum rank		7	73	14	22	32	50	49	60	36	34	53	64	73
Mean rank		1	10.4285	2	3.1428	4.5714	7.1428	7	8.5714	5.1428	4.8571	7.5714	9.1428	10.4285
Total rank		1	12	2	3	4	8	7	10	6	5	9	11	12

for the objective function corresponds to the best candidate solution and similarly the worst value obtained for the objective function corresponds to the worst candidate solution. Since the candidate solutions are updated in each iteration, the best and worst candidate solutions must also be updated in each iteration. After the implementation of the algorithm, the best candidate solution obtained during the iterations of the algorithm is presented as a solution to the problem. The process of updating candidate solutions in the proposed RPO consists of two phases of exploration and exploitation, which are described as follows.

2) PHASE 1: THE STRATEGY OF RED PANDAS IN FORAGING (EXPLORATION)

The position of red pandas in the first phase of RPO is modeled based on their movement in order to forage in the wild. Red pandas are highly skilled in identifying and moving towards the location of food sources using their high abilities

TABLE 3. Optimization results of the high-dimensional multimodal test functions.

		RPO	WSO	AVOA	RSA	MPA	TSA	WOA	MVO	GWO	TLBO	GSA	PSO	GA
F8	mean	-12474.4	-6585.34	-12378	-5503.43	-9607.07	-6149.15	-11384.8	-7831.31	-6026.15	-5485.59	-2655.59	-6859.48	-8586.32
	best	-12474.4	-8082.13	-12569.5	-5655.12	-10195.9	-7589.76	-12568.7	-8752.78	-7136.57	-6913.77	-3460.74	-7952.19	-9885.55
	worst	-12474.4	-5339.49	-11464	-5.02E+03	-8487.95	-5152.22	-8654.92	-6960.7	-4398.37	-4764.14	-2191.52	-5771.23	-7127.7
	std	3.73E-12	717.4584	372.4878	158.8838	442.981	618.3085	1421.809	496.9466	694.8304	559.5748	315.5151	549.7553	708.3909
	median	-12474.4	-6553.85	-12569.5	-5537.38	-9575.73	-6058.25	-11960.3	-7846.27	-5799.57	-5491.83	-2619.13	-6864.01	-8660.87
	rank	1	8	2	11	4	9	3	6	10	12	13	7	5
F9	mean	0	33.71054	0	0.00E+00	0	1.65E+02	2.84E-15	113.0038	8.53E-15	0	2.45E+01	65.08238	56.08746
	best	0	19.32847	0	0.00E+00	0	9.94E+01	0	64.7574	0	0	15.91933	34.8272	28.44008
	worst	0	60.08227	0	0.00E+00	0	2.23E+02	5.68E-14	159.2712	5.68E-14	0	4.18E+01	114.491	96.39632
	std	0	11.24004	0	0.00E+00	0	3.24E+01	1.27E-14	29.65035	2.08E-14	0	6.72E+00	22.0399	17.7611
	median	0	34.6458	0	0.00E+00	0	1.70E+02	0	116.4866	0	0	23.879	59.20977	51.70003
	rank	1	5	1	1	1	9	2	8	3	1	4	7	6
F10	mean	8.88E-16	6.990967	8.88E-16	8.88E-16	4.09E-15	0.933101	3.91E-15	0.541256	1.63E-14	4.44E-15	7.99E-09	3.039996	3.70709
	best	8.88E-16	4.218409	8.88E-16	8.88E-16	8.88E-16	7.99E-15	8.88E-16	0.095271	1.15E-14	4.44E-15	6.06E-09	1.898455	3.145878
	worst	8.88E-16	10.71603	8.88E-16	8.88E-16	4.44E-15	3.647303	7.99E-15	1.704507	2.22E-14	4.44E-15	1.18E-08	4.054739	4.671155
	std	0	1.547191	0	0	1.09E-15	1.474687	2.09E-15	0.512215	3.51E-15	0	1.61E-09	0.566816	0.489157
	median	8.88E-16	6.704537	8.88E-16	8.88E-16	4.44E-15	1.51E-14	4.44E-15	0.324594	1.51E-14	4.44E-15	7.90E-09	2.958008	3.534577
	rank	1	11	1	1	3	8	2	7	5	4	6	9	10
F11	mean	0	4.54676	0	0	0	0.008977	0	0.423722	0.003124	0	8.977727	0.161128	1.537545
	best	0	1.167062	0	0	0	0	0	0.254389	0	0	3.897009	0.001351	1.236638
	worst	0	9.53335	0	0	0	0.021749	0	0.603397	0.033615	0	16.17909	0.674206	2.380255
	std	0	2.58369	0	0	0	0.007396	0	0.10199	0.008572	0	3.484564	0.191215	0.235609
	median	0	4.297575	0	0	0	0.010512	0	0.417193	0	0	8.093364	0.091004	1.500304
	rank	1	7	1	1	1	3	1	5	2	1	8	4	6
F12	mean	6.16E-15	8938.535	3.13E-09	1.35E+00	1.87E-10	7.024464	0.009343	1.008126	0.032199	0.080382	0.188433	1.361297	0.228629
	best	1.57E-32	1.520943	1.01E-09	0.730526	1.05E-10	1.21049	0.000893	0.00082	0.003646	0.055022	4.20E-19	0.000304	0.033411
	worst	1.22E-13	130992.2	8.50E-09	1.66E+00	3.14E-10	2.01E+01	0.028761	3.676108	0.069328	0.110581	0.728748	4.775259	1.014181
	std	2.74E-14	30118.49	2.08E-09	0.310746	7.07E-11	4.177653	0.008396	1.099896	0.014526	0.017974	0.241821	1.285973	0.224657
	median	1.08E-20	7.565528	2.54E-09	1.52E+00	1.60E-10	6.557718	0.007814	0.807789	0.031831	0.077297	0.107447	1.249149	0.192671
	rank	1	13	3	10	2	12	4	9	5	6	7	11	8
F13	mean	1.02E-28	37270.51	1.50E-08	2.66E-01	0.002222	2.89E+00	0.216461	0.036717	0.569041	1.115878	0.153505	6.191399	2.461518
	best	1.35E-32	19.34587	1.65E-09	2.79E-31	8.10E-10	1.503915	0.04073	0.013226	0.192399	0.541372	6.58E-18	0.087294	1.184657
	worst	5.68E-25	448852.6	1.02E-07	2.73E+00	0.01337	3.59E+00	0.756973	0.081024	0.925017	1.615842	1.22246	15.60791	5.511804
	std	4.29E-16	104177.3	2.24E-08	8.20E-01	0.004285	4.97E-01	0.16901	0.019829	0.201783	0.304186	0.337971	4.455618	1.239427
	median	6.32E-30	299.0376	9.91E-09	1.07E-30	3.28E-09	2.83E+00	0.182965	0.030446	0.511049	1.128007	1.40E-17	6.088677	2.118455
	rank	1	13	2	7	3	11	6	4	8	9	5	12	10
Sum rank		6	57	10	31	14	52	18	39	33	33	43	50	45
Mean rank		1	9.5	1.6667	5.166667	2.333333	8.666667	3	6.5	5.5	5.5	7.166667	8.333333	7.5
Total rank		1	12	2	5	3	11	4	7	6	6	8	10	9

in smell, hearing, and vision. In RPO design, for each red panda, the location of other red pandas that lead to better objective function values is considered as the location of food resources. The set of proposed food resource positions for each red panda based on the comparison of the objective function values is modeled using (4). Among these proposed positions, one position is randomly determined as the food position selected by the corresponding red panda.

$$PFS_i = \{X_k | k \in \{1, 2, \ldots, N\} \; and \; F_k < F_i\} \cup \{X_{best}\}, \tag{4}$$

where PFS_i is the set of proposed food sources for ith red panda and X_{best} is the location of the red panda with best value for the objective function (best candidate solution).

Moving towards the food source leads to big changes in the position of red pandas, which improves the capability of the proposed algorithm in exploration and global search in the problem-solving space. In order to model the behavior of red pandas during foraging, first a new position is calculated for each red panda based on movement towards the location of food source (the best candidate solution) using (5). Then, if the value of the objective function is improved in the new position, the position of the red panda is updated to the position calculated in the exploration phase using (6).

$$X_i^{P1} : x_{i,j}^{P1} = x_{i,j} + r \cdot (SFS_{i,j} - I \cdot x_{i,j}) \tag{5}$$

TABLE 4. Optimization results of the fixed-dimensional multimodal test functions.

		RPO	WSO	AVOA	RSA	MPA	TSA	WOA	MVO	GWO	TLBO	GSA	PSO	GA
F14	mean	0.998004	1.343751	1.097209	5.227334	1.73999	7.630947	3.693804	0.998004	4.480656	0.998005	4.290907	5.155666	1.033146
	best	0.998004	0.998004	0.998004	1.002758	0.998004	0.998004	0.998004	0.998004	0.998004	0.998004	1.363452	0.998004	0.998004
	worst	0.998004	5.928845	2.982105	1.27E+01	6.903336	12.67051	10.76318	0.998004	10.76318	0.998014	13.94578	15.50382	1.550495
	std	5.09E-17	1.16662	0.443659	3.67734	1.426005	5.051667	3.825977	3.58E-12	3.828346	2.49E-06	3.427131	4.532665	0.124072
	median	0.998004	0.998004	0.998004	2.982156	0.998004	7.365715	1.992031	0.998004	2.982105	0.998004	3.147991	3.96825	0.998006
	rank	1	6	5	12	7	13	8	2	10	3	9	11	4
F15	mean	0.000307	0.000366	0.000366	1.40E-03	0.000686	7.57E-03	5.62E-04	0.002615	4.37E-03	0.003429	2.57E-03	0.001462	0.007961
	best	0.000307	0.000307	0.000307	5.07E-04	0.000324	3.08E-04	0.000321	0.000308	0.000307	0.000309	0.000584	0.000307	0.00084
	worst	0.000307	0.000672	0.001223	3.67E-03	0.001593	2.09E-02	1.43E-03	0.020363	2.04E-02	0.020364	6.95E-03	0.020363	0.028106
	std	2.49E-19	0.000115	0.000205	6.94E-04	0.000352	9.76E-03	2.35E-04	0.006075	8.21E-03	0.007303	1.41E-03	0.004464	0.009646
	median	0.000307	0.000307	0.000311	1.29E-03	0.000562	5.01E-04	0.000478	0.000671	0.000308	0.00033	0.00219	0.000307	0.003474
	rank	1	2	3	6	5	12	4	9	11	10	8	7	13
F16	mean	-1.03E+00	-1.03163	-1.03E+00	-9.77E-01	-1.03E+00	-1.02847	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.03163
	best	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.03163
	worst	-1.03E+00	-1.03163	-1.03E+00	0.00E+00	-1.03E+00	-1	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.0316
	std	2.22E-16	2.28E-16	1.25E-16	0.230081	6.40E-04	0.009735	7.04E-11	4.45E-08	6.24E-09	2.12E-06	1.02E-16	1.35E-16	6.97E-06
	median	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.03E+00	-1.03E+00	-1.03E+00	-1.03163	-1.03163
	rank	1	1	1	9	7	8	2	4	3	5	1	1	6
F17	mean	0.397887	0.397899	0.397887	0.41522	0.398296	0.397911	0.397889	0.397887	0.397896	0.40432	0.397887	0.776311	0.821605
	best	0.397887	0.397887	0.397887	0.398605	0.397887	0.397888	0.397887	0.397887	0.397887	0.39789	0.397887	0.397887	0.397887
	worst	0.397887	0.398059	0.397887	0.520792	0.402341	0.397959	0.39791	0.397888	0.398048	0.524448	0.397887	2.320028	2.435846
	std	0	3.96E-05	0	0.028731	0.001049	2.38E-05	4.99E-06	6.69E-08	3.58E-05	0.028275	0	0.70116	0.719311
	median	0.397887	0.397887	0.397887	0.403394	0.397888	0.3979	0.397888	0.397887	0.397888	0.397976	0.397887	0.397887	0.398001
	rank	1	5	1	9	7	6	3	2	4	8	1	10	11
F18	mean	3.00E+00	3	3.00E+00	7.17E+00	3.00E+00	10.15176	3.000009	3	3.000009	3.000001	3	3	3.002563
	best	3.00E+00	3	3.00E+00	3	3.00E+00	3	3	3	3	3	3.00E+00	3	3
	worst	3.00E+00	3	3.00E+00	8.40E+01	3.00E+00	9.20E+01	3.000037	3.000002	3.000054	3.000004	3	3	3.01211
	std	1.30E-15	4.78E-16	8.52E-07	18.10082	4.90E-04	20.97897	1.15E-05	3.64E-07	1.37E-05	1.22E-06	3.40E-15	3.12E-15	0.004167
	median	3.00E+00	3	3.00E+00	3.00E+00	3.00E+00	3.00001	3.000002	3	3.000003	3.000001	3	3	3.000043
	rank	1	1	5	11	9	12	7	4	8	6	3	2	10
F19	mean	-3.86278	-3.86278	-3.86E+00	-3.80E+00	-3.8042	-3.86E+00	-3.86012	-3.86278	-3.8616	-3.86171	-3.86278	-3.82413	-3.86263
	best	-3.86E+00	-3.86278	-3.86E+00	-3.86E+00	-3.86E+00	-3.86278	-3.86278	-3.86278	-3.86278	-3.86275	-3.86E+00	-3.86278	-3.86278
	worst	-3.86278	-3.86278	-3.86E+00	-3.68E+00	-3.69874	-3.85E+00	-3.8549	-3.86278	-3.8549	-3.85477	-3.86278	-3.08976	-3.86083
	std	2.28E-15	2.28E-15	4.13E-13	5.49E-02	0.055686	1.76E-03	0.002997	1.26E-07	0.002471	0.002365	1.90E-15	0.172852	0.000438
	median	-3.86E+00	-3.86278	-3.86E+00	-3.81E+00	-3.82E+00	-3.86E+00	-3.86174	-3.86278	-3.86276	-3.86244	-3.86E+00	-3.86278	-3.86277
	rank	1	1	2	11	10	5	8	3	7	6	1	9	4
F20	mean	-3.322	-3.2794	-3.29227	-2.59297	-2.5525	-3.2547	-3.20977	-3.24457	-3.25561	-3.25343	-3.322	-3.26462	-3.18185
	best	-3.322	-3.322	-3.322	-3.0964	-3.11109	-3.32152	-3.32192	-3.32199	-3.32199	-3.31538	-3.322	-3.322	-3.31657
	worst	-3.322	-3.1903	-3.2031	-1.24082	-1.93849	-3.13678	-2.43159	-3.20235	-3.08668	-3.13436	-3.322	-3.13764	-2.87031
	std	3.95E-16	0.059076	0.05282	0.55936	0.396512	0.070032	0.202952	0.058293	0.080305	0.061594	4.20E-16	0.074972	0.125112
	median	-3.322	-3.32197	-3.322	-2.77999	-2.66695	-3.26088	-3.32058	-3.20303	-3.32199	-3.29898	-3.322	-3.322	-3.20784
	rank	1	3	2	11	12	6	9	8	5	7	1	4	10
F21	mean	-10.1532	-7.40685	-10.1532	-5.0552	-10.1532	-6.50499	-9.26468	-7.61784	-9.30677	-7.14369	-5.58745	-4.64513	-5.40415
	best	-10.1532	-10.1532	-10.1532	-5.0552	-10.1532	-10.1043	-10.1529	-10.1532	-10.1531	-9.79821	-10.1532	-10.1532	-9.02834
	worst	-10.1532	-2.68286	-10.1532	-5.0552	-10.1532	-2.61136	-2.63044	-5.05518	-3.33802	-4.91024	-2.68286	-2.63047	-2.34608
	std	1.95E-15	3.517193	7.20E-15	2.64E-07	3.35E-07	3.010769	2.209817	2.601227	2.092307	1.522661	3.502858	2.98588	2.109838
	median	-10.1532	-10.1532	-10.1532	-5.0552	-10.1532	-5.06631	-10.1504	-7.62691	-10.1527	-7.69573	-3.29594	-2.68286	-5.36517
	rank	1	7	2	12	3	9	5	6	4	8	10	13	11
F22	mean	-10.4029	-8.15674	-10.4029	-5.08767	-10.4029	-6.68047	-9.33407	-8.16688	-10.4024	-7.28041	-10.0705	-7.5143	-6.12192
	best	-10.4029	-10.4029	-10.4029	-5.08767	-10.4029	-10.3275	-10.4028	-10.4029	-10.4029	-9.74769	-10.4029	-10.4029	-10.0174
	worst	-10.4029	-2.75193	-10.4029	-5.08767	-10.4028	-2.72446	-5.08766	-2.76589	-10.4019	-4.04886	-3.7544	-1.83759	-2.36544
	std	3.51E-15	3.525634	1.07E-14	7.96E-07	4.99E-05	3.412059	2.178407	2.854208	0.000307	1.804726	1.48666	3.679456	3.289457
	median	-10.4029	-10.4029	-10.4029	-5.08767	-10.4029	-5.07435	-10.399	-10.4028	-10.4025	-7.88932	-10.4029	-10.4029	-6.0991
	rank	1	8	2	13	3	11	6	7	4	10	5	9	12
F23	mean	-10.5364	-9.43483	-10.5364	-5.12847	-10.5363	-6.96616	-8.2365	-9.72988	-9.85988	-8.37514	-10.2396	-6.29865	-6.74887
	best	-10.5364	-10.5364	-10.5364	-5.12848	-10.5364	-10.485	-10.5363	-10.5364	-10.5364	-9.72222	-10.5364	-10.5364	-10.1183
	worst	-10.5364	-2.87114	-10.5364	-5.12847	-10.5363	-1.67401	-2.42166	-5.12847	-2.42171	-4.67125	-4.60022	-2.42173	-2.74409

TABLE 4. ***(Continued.)*** **Optimization results of the fixed-dimensional multimodal test functions.**

	std	2.79E-15	2.696453	6.89E-15	1.79E-06	3.83E-05	3.898186	2.944939	1.969662	2.126773	1.123547	1.327372	3.942886	2.517185
	median	-10.5364	-10.5364	-10.5364	-5.12847	-10.5363	-10.209	-10.5317	-10.5363	-10.536	-8.61385	-10.5364	-3.35328	-7.38736
	rank	1	7	2	13	3	10	9	6	5	8	4	12	11
Sum rank		10	41	25	107	66	92	61	51	61	71	43	78	92
Mean rank		1	4.1	2.5	10.7	6.6	9.2	6.1	5.1	6.1	7.1	4.3	7.8	9.2
Total rank		1	3	2	11	7	10	6	5	6	8	4	9	10

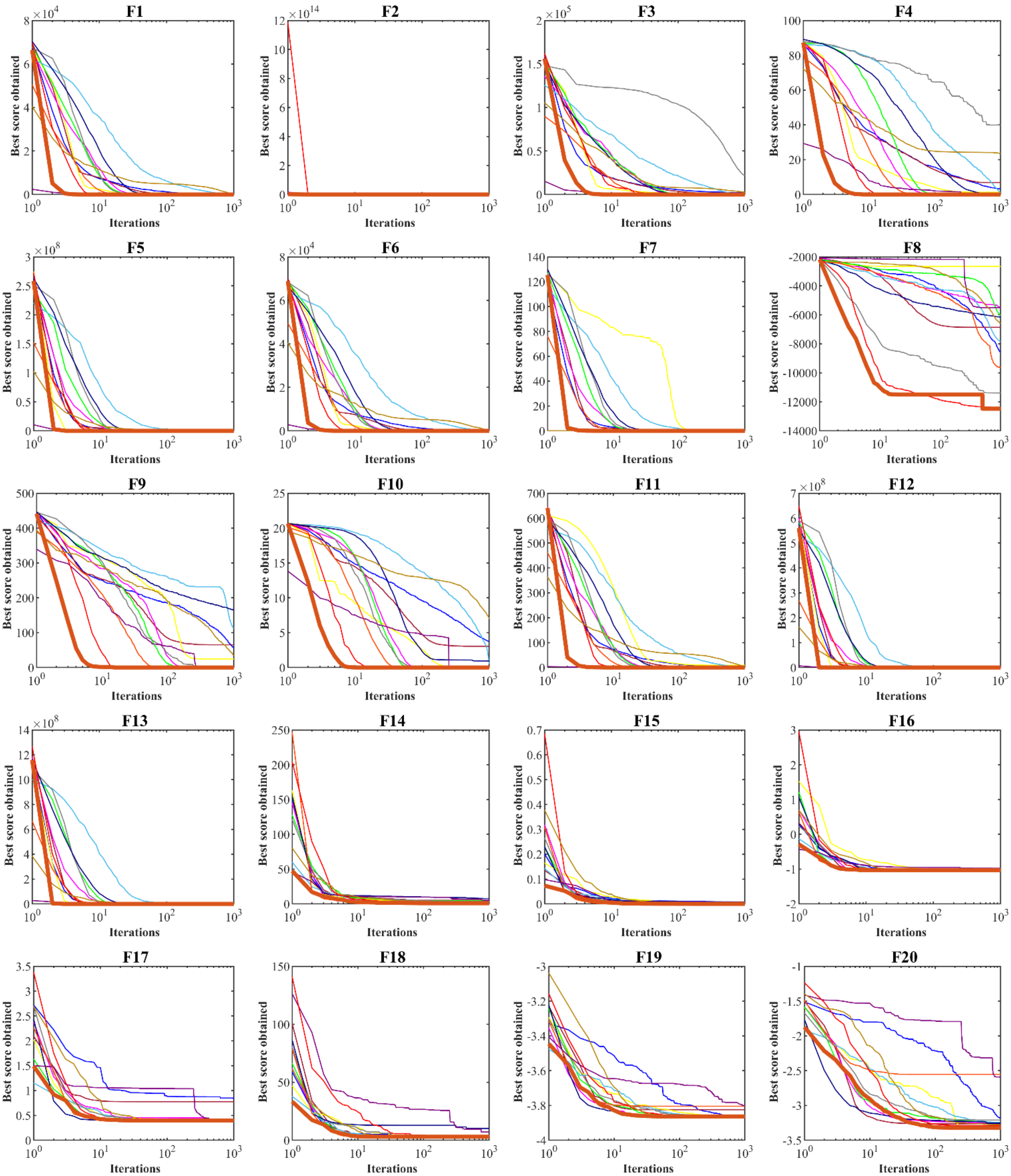

FIGURE 3. **Convergence curves of RPO and the competitor algorithms performances for F1 to F23 test functions.**

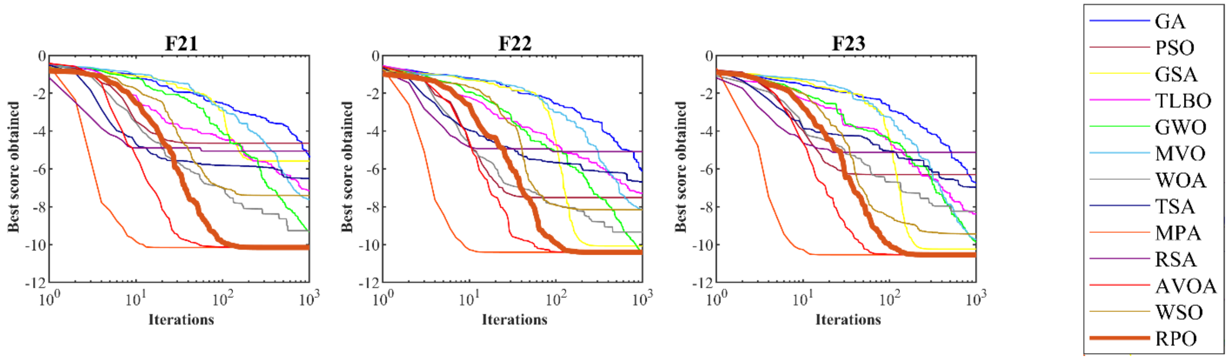

FIGURE 3. *(Continued.)* **Convergence curves of RPO and the competitor algorithms performances for F1 to F23 test functions.**

$$X_i = \begin{cases} X_i^{P1}, & F_i^{P1} < F_i; \\ X_i, & else, \end{cases} \tag{6}$$

where, X_i^{P1} is the new position of the ith red panda based on the first phase of RPO, $x_{i,j}^{P1}$ is its jth dimension, F_i^{P1} represents its objective function value, SFS_i is the selected food source for ith red panda, $SFS_{i,j}$ denotes its jth dimension, r is a random number in the interval $[0, 1]$, and I is a random number selected from the set $\{1, 2\}$ randomly.

3) PHASE 2: SKILL IN CLIMBING AND RESTING ON THE TREE (EXPLOITATION)

The position of red pandas in the second phase of the RPO is modeled based on the skill of this animal in climbing trees and resting on them. Red pandas spend most of their time resting on trees. After foraging on the ground, this animal climbs the nearby trees. Moving towards the tree and climbing it leads to small changes in the position of red pandas, which increases the capability of the proposed RPO algorithm in exploitation and local search in promising areas. In order to mathematically model the natural behavior of red pandas in climbing trees, first a new position is calculated for each red panda using (7). Then, if the value of the objective function is improved, this new position replaces the previous position of the corresponding red panda using (8).

$$x_{i,j}^{P2} = x_{i,j} + \frac{lb_j + r \cdot (ub_j - lb_j)}{t}, \quad i = 1, 2, \ldots, N, \\ j = 1, 2, \ldots, m, \quad and\ t = 1, 2, \ldots, T, \tag{7}$$

$$X_i = \begin{cases} X_i^{P2}, & F_i^{P2} < F_i; \\ X_i, & else, \end{cases} \tag{8}$$

where X_i^{P2} is the new position of the ith red panda based on the second phase of RPO, $x_{i,j}^{P2}$ is its jth dimension, F_i^{P2} indicates its objective function value, r is a random number in the interval $[0, 1]$, t represents the iteration counter of the algorithm, and T is the maximum number of iterations.

C. REPETITIONS PROCESS, FLOWCHART, AND PSEUDO-CODE OF RPO

The proposed RPO approach is an iteration-based metaheuristic algorithm. After updating the position of all red pandas based on the exploration and exploitation phases, the first iteration of the RPO is completed. Then, based on the new values, the algorithm enters the next iteration and the process of updating the position of the red pandas is repeated using (4) to (8) until the last iteration of the algorithm. After completion of RPO implementation, the position of the best red panda, which results in the best value for the objective function, is presented as the solution of the problem. The implementation steps of RPO are presented in the form of a flowchart in Figure 2 and its pseudocode is given in Algorithm 1.

D. COMPUTATIONAL COMPLEXITY

In this subsection, the computational complexity analysis of the proposed RPO approach is discussed. RPO initialization has a computational complexity equal to $O(Nm)$, where N is the number of red pandas and m denotes the number of problem variables. In each iteration of the algorithm, the position of red pandas is updated in the two phases of exploration and exploitation. Therefore, the red pandas update process has a computational complexity equal to $O(2NmT)$, where T is the maximum number of the algorithm iterations. Therefore, the total computational complexity of RPO is equal to $O(Nm(1 + 2T))$.

IV. SIMULATION STUDIES AND DISCUSSION

In this section, simulation studies on the performance of the proposed RPO in solving optimization problems are presented. For this purpose, fifty-two standard benchmark functions consisting of unimodal, high-dimensional multimodal, and fixed-dimensional multimodal types as well as CEC 2017 test suite [86] are employed. Also, in order to analyze the quality of RPO in providing appropriate solutions, the results obtained from the proposed approach are compared with the performance of twelve well-known metaheuristic algorithms including: GA, PSO, GSA, TLBO, GWO, MVO,

TABLE 5. Optimization results of the CEC-2017 test suite.

		RPO	WSO	AVOA	RSA	MPA	TSA	WOA	MVO	GWO	TLBO	GSA	PSO	GA
C17-F1	mean	100	3.07E+03	4.09E+03	1.09E+10	6.59E+07	1.86E+09	6.88E+06	8014.981	9.41E+07	1.57E+08	7.90E+02	3.35E+03	1.26E+07
	best	100	5.01E+02	1.17E+02	9.41E+09	2.08E+04	3.98E+08	5.01E+06	5095.679	2.96E+04	6.99E+07	1.00E+02	3.62E+02	6.55E+06
	worst	100	7.40E+03	1.27E+04	1.30E+10	2.39E+08	4.04E+09	9.06E+06	11813.26	3.42E+08	3.79E+08	1.90E+03	9.92E+03	1.81E+07
	std	2.08E-06	3.23E+03	5.87E+03	1.60E+09	1.16E+08	1.62E+09	1.71E+06	3143.464	1.66E+08	1.49E+08	7.79E+02	4.42E+03	4.84E+06
	median	100	2.20E+03	1.78E+03	1.06E+10	1.21E+07	1.49E+09	6.73E+06	7575.492	1.72E+07	8.97E+07	5.78E+02	1.55E+03	1.29E+07
	rank	1	3	5	13	9	12	7	6	10	11	2	4	8
C17-F3	mean	300	464.5135	302.0192	10267.95	2366.725	11925.54	1824.674	300.0582	3252.465	754.5378	10918.38	300	15733.23
	best	300	300.1927	300	5527.263	1216.811	4528.989	640.4615	300.0135	1609.729	482.5901	6863.28	300	4618.157
	worst	300	844.4021	304.319	13744.58	4470.517	16868.69	3531.535	300.1326	6257.882	932.1847	14847.31	300	24879.84
	std	5.56E-12	254.5224	2.340144	3753.137	1499.508	5234.208	1360.552	0.052255	2142.154	196.8718	3289.345	4.64E-14	10572.55
	median	300	356.7296	301.8789	10899.98	1889.787	13152.24	1563.35	300.0434	2571.124	801.6882	10981.47	300	16717.47
	rank	2	5	4	10	8	12	7	3	9	6	11	1	13
C17-F4	mean	400	407.2957	405.0707	1414.847	410.9365	588.2747	426.8392	403.5586	412.5268	409.7871	404.8591	421.6779	415.7083
	best	400	406.6334	401.3246	874.8043	404.5605	483.0731	406.8749	401.7011	406.4989	408.9495	403.8009	400.1128	412.4635
	worst	400	407.9785	406.9653	1943.823	421.2244	711.1053	478.5016	405.2243	430.2669	410.3159	406.4845	475.105	419.6784
	std	2.08E-10	0.610202	2.654651	455.8057	7.631059	111.6505	34.50837	1.829878	11.81396	0.585232	1.229366	35.94524	3.154122
	median	400	407.2854	405.9964	1420.38	408.9806	579.4602	410.9901	403.6545	406.6707	409.9415	404.5755	405.7469	415.3457
	rank	1	5	4	13	7	12	11	2	8	6	3	10	9
C17-F5	mean	505.8467	517.9215	547.3819	578.3821	514.8584	569.274	544.0672	525.4579	513.9576	536.616	557.956	529.9867	530.1072
	best	503.4832	514.9671	528.8537	562.5519	509.3634	546.5185	525.2044	510.9493	509.1122	530.7196	552.7326	511.9395	525.052
	worst	507.4645	520.8979	567.6569	594.5714	521.7037	603.7437	582.6718	540.7958	521.8296	540.4401	570.6416	555.7175	536.3365
	std	1.962468	2.912836	20.36247	17.73972	5.735235	25.35835	26.88804	12.44919	5.487105	4.268117	8.563761	20.1965	5.10332
	median	506.2196	517.9106	546.5084	578.2025	514.1833	563.4168	534.1962	525.0432	512.4443	537.6521	554.225	526.145	529.5201
	rank	1	4	10	13	3	12	9	5	2	8	11	6	7
C17-F6	mean	600.0001	600.8465	618.7414	644.0479	601.6577	626.8685	625.0691	602.3263	601.2195	607.426	618.6179	608.0398	611.1025
	best	600	600.0026	617.6546	640.5716	601.0979	616.3121	608.1441	600.5108	600.645	605.1494	603.1557	601.4659	607.472
	worst	600.0001	601.6626	621.5024	648.6505	602.9781	643.7389	648.9105	604.6673	601.8601	610.9759	639.1099	620.8405	615.6957
	std	3.34E-05	0.915818	1.84396	3.62651	0.894333	11.81854	17.14696	1.865805	0.502461	2.653415	16.6149	8.778418	3.641276
	median	600.0001	600.8604	617.9043	643.4847	601.2773	623.7115	621.6108	602.0636	601.1865	606.7894	616.1031	604.9264	610.6212
	rank	1	2	10	13	4	12	11	5	3	6	9	7	8
C17-F7	mean	716.9658	724.7542	770.0335	812.0262	727.0803	838.1264	766.2735	732.5036	727.2388	755.4194	717.6126	734.5213	738.9913
	best	714.6235	715.3458	746.6222	797.7337	720.3352	794.7635	754.3765	717.639	717.9756	750.5447	715.0813	726.7135	727.7346
	worst	719.6695	737.9637	800.0727	825.7699	738.8916	883.0503	798.0781	753.3847	746.2284	764.2491	721.6935	747.0726	743.9886
	std	2.158199	10.44666	24.57847	13.15284	8.349296	38.31833	21.24208	15.01745	12.97004	6.139167	2.854204	9.260546	7.616713
	median	716.7852	722.8537	766.7196	812.3006	724.5471	837.3459	756.3197	729.4954	722.3756	753.4418	716.8378	732.1495	742.121
	rank	1	3	11	12	4	13	10	6	5	9	2	7	8
C17-F8	mean	805.0735	808.7077	833.5798	858.0199	816.1139	852.1631	839.2534	812.6909	817.042	840.7119	821.3916	824.5363	818.0629
	best	803.4824	803.9798	821.8891	845.8319	811.1057	834.5879	820.0417	807.9622	811.3169	833.2718	812.9345	816.9143	813.7883
	worst	806.364	815.9193	850.7427	863.7385	819.4207	873.1022	852.425	817.9172	822.4848	849.4086	829.8487	831.483	826.546
	std	1.204995	5.219796	12.18224	8.24759	3.666985	17.09959	13.8905	4.089768	4.665544	8.247461	7.189688	7.204848	5.746715
	median	805.2239	807.4659	830.8436	861.2546	816.9645	850.4811	842.2735	812.4422	817.1831	840.0837	821.3916	824.8739	815.9586
	rank	1	2	9	13	4	12	10	3	5	11	7	8	6
C17-F9	mean	900	929.3131	1211.93	1514.186	909.5045	1420.443	1414.578	900.8677	912.9217	912.8054	900	904.5932	905.5344
	best	900	900.0012	958.169	1410.468	900.5408	1190.18	1088.519	900.0011	900.6207	907.8307	900	900.9737	903.0297
	worst	900	971.5927	1724.097	1663.035	925.2083	1732.526	1718.904	903.3723	935.8728	921.6602	900	913.3387	909.8295
	std	1.93E-12	34.85359	354.5162	107.4221	11.28278	234.418	265.082	1.668413	16.51996	6.071066	0	5.898045	3.072024
	median	900	922.8293	1082.726	1491.62	906.1344	1379.533	1425.445	900.0487	907.5966	910.8653	900	902.0301	904.6392
	rank	2	9	10	13	6	12	11	3	8	7	1	4	5
C17-F10	mean	1144.138	1525.5	1833.3	2692.02	1701.744	2106.677	2098.525	1836.484	1777.094	2256.198	2369.988	2013.59	1766.646
	best	1064.073	1118.755	1517.483	2508.239	1514.869	1811.861	1480.705	1488.613	1576.796	1836.742	2069.602	1599.934	1443.475
	worst	1245.127	2173.755	2515.046	3078.344	1914.964	2376.805	2661.325	2375.001	2063.577	2565.7	2484.022	2449.501	2191.037
	std	74.81271	454.5228	467.8448	264.9682	163.8881	298.1701	569.8336	429.0835	206.5988	309.6127	200.4254	348.6038	320.0971
	median	1133.676	1404.745	1650.336	2590.748	1688.571	2119.02	2126.036	1741.161	1734	2311.176	2463.163	2002.462	1716.037
	rank	1	2	6	13	3	10	9	7	5	11	12	8	4
C17-F11	mean	1100.626	1124.026	1151.953	4189.355	1146.355	5769.669	1154.582	1129.463	1159.204	1154.533	1141.981	1146.624	2474.015
	best	1100.497	1117.954	1118.262	1484.117	1120.789	5610.898	1113.882	1105.941	1123.16	1140.523	1121.042	1134.545	1116.115
	worst	1100.995	1133.078	1209.026	6861.324	1203.616	5856.783	1178.303	1152.392	1237.505	1177.45	1173.496	1169.648	6326.297
	std	0.245975	6.452637	39.9058	2414.009	38.85384	109.1639	29.71527	23.18336	53.245	15.92274	22.36238	15.79061	2565.955

TABLE 5. *(Continued.)* Optimization results of the CEC-2017 test suite.

	median	1100.505	1122.536	1140.263	4205.989	1130.508	5805.498	1163.071	1129.76	1138.074	1150.079	1136.692	1141.152	1226.824
	rank	1	2	7	12	5	13	9	3	10	8	4	6	11
C17-F12	mean	1294.321	2.34E+03	1.18E+06	7.58E+08	1.07E+06	1.12E+06	2.53E+06	1.11E+06	1.52E+06	5.43E+06	1.10E+06	8.59E+03	6.50E+05
	best	1259.427	1.36E+03	3.82E+05	1.68E+08	3.55E+04	5.79E+05	1.84E+05	9.39E+03	4.87E+04	1.45E+06	5.10E+05	2.61E+03	1.88E+05
	worst	1378.957	4.30E+03	2.14E+06	1.32E+09	1.67E+06	1.37E+06	4.19E+06	3.47E+06	2.38E+06	9.61E+06	1.85E+06	1.48E+04	1.15E+06
	std	56.63799	1.32E+03	8.23E+05	5.84E+08	718201	3.73E+05	1.86E+06	1.60E+06	1.03E+06	4.31E+06	5.68E+05	5.57E+03	3.93E+05
	median	1269.45	1.86E+03	1.10E+06	7.69E+08	1.28E+06	1.26E+06	2.87E+06	4.70E+05	1.83E+06	5.32E+06	1.01E+06	8.45E+03	6.32E+05
	rank	1	2	9	13	5	8	11	7	10	12	6	3	4
C17-F13	mean	1305.316	1.33E+03	19590.41	3.69E+07	8636.952	1.36E+04	8.04E+03	7129.179	1.10E+04	1.79E+04	1.07E+04	7.01E+03	5.84E+04
	best	1303.049	1.31E+03	2827.835	3.07E+06	5576.112	8.05E+03	3.43E+03	1392.062	6891.887	1.69E+04	5.32E+03	2.46E+03	9.08E+03
	worst	1307.325	1.36E+03	33636.44	1.23E+08	11255.07	2.16E+04	1.62E+04	13197.76	1.54E+04	2.03E+04	1.51E+04	1.79E+04	1.93E+05
	std	1.760128	2.32E+01	15909.58	5.72E+07	2457.335	5.83E+03	5.81E+03	6108.204	3.46E+03	1.64E+03	4.14E+03	7.30E+03	89872.51
	median	1305.445	1.32E+03	20948.69	1.10E+07	8858.314	1.24E+04	6.28E+03	6963.447	1.08E+04	1.71E+04	1.12E+04	3.87E+03	1.56E+04
	rank	1	2	11	13	6	9	5	4	8	10	7	3	12
C17-F14	mean	1402.488	1423.187	2068.49	5643.2	2284.602	3535.701	1527.934	1584.776	2417.442	1605.133	5878.186	3115.468	13829.69
	best	1401.492	1406.965	1699.949	4926.752	1459.264	1494.468	1487.913	1424.777	1466.935	1524.797	4842.303	1434.97	3901.489
	worst	1403.483	1435.919	2935.299	7310.2	4164.993	5897.609	1570.738	2038.104	5230.982	1637.676	8015.687	7252.678	27663.3
	std	0.907419	12.2306	581.5339	1118.367	1279.926	2339.604	42.25814	301.9551	1873.64	53.75547	1485.2	2777.326	10054.71
	median	1402.487	1424.932	1819.356	5167.924	1757.076	3375.364	1526.542	1438.111	1485.926	1629.029	5327.376	1887.112	11876.98
	rank	1	2	6	11	7	10	3	4	8	5	12	9	13
C17-F15	mean	1500.608	1.52E+03	5583.045	1.48E+04	5552.776	7.42E+03	6572.399	1544.964	6.14E+03	1.72E+03	2.56E+04	9.56E+03	4778.883
	best	1500.306	1511.684	2115.426	2.83E+03	4019.915	2.38E+03	2053.065	1527.841	3725.494	1.59E+03	1.20E+04	2974.495	1919.88
	worst	1500.795	1.53E+03	13461.88	3.25E+04	6808.759	1.34E+04	1.43E+04	1557.896	7.31E+03	1.82E+03	3.84E+04	1.58E+04	8502.898
	std	0.211074	7.53E+00	5287.435	1.30E+04	1148.546	4.72E+03	5351.432	13.11231	1.64E+03	1.13E+02	1.26E+04	5.35E+03	3269.257
	median	1500.665	1.52E+03	3377.437	1.19E+04	5691.214	6.95E+03	4946.423	1547.06	6.76E+03	1.74E+03	2.59E+04	9736.253	4346.377
	rank	1	2	7	12	6	10	9	3	8	4	13	11	5
C17-F16	mean	1600.854	1631.963	1825.2	2047.53	1731.61	2080.719	1976.657	1832.371	1738.057	1682.598	2108.525	1947.807	1817.44
	best	1600.578	1602.024	1645.364	1835.697	1649.67	1881.961	1777.435	1735.911	1616.927	1654.683	1972.998	1839.08	1727.547
	worst	1601.25	1719.592	1950.565	2341.888	1795.524	2279.121	2114.447	1898.671	1842.279	1740.941	2317.949	2119.513	1850.824
	std	0.293201	58.35722	128.4311	213.5445	60.57862	179.9981	160.0168	68.76372	92.88131	40.15491	156.6607	129.7601	59.91419
	median	1600.794	1603.117	1852.436	2006.267	1740.623	2080.897	2007.374	1847.452	1746.51	1667.383	2071.577	1916.318	1845.695
	rank	1	2	7	11	4	12	10	8	5	3	13	9	6
C17-F17	mean	1703.369	1738.518	1754.813	1827.274	1758.536	1809.876	1852.567	1853.514	1773.682	1762.775	1857.806	1756.302	1760.194
	best	1700.908	1710.808	1736.971	1809.096	1727.373	1793.527	1779.149	1784.392	1726.278	1751.879	1751.548	1749.136	1756.835
	worst	1710.327	1753.339	1802.155	1837.197	1835.37	1821.594	1903.594	1969.318	1884.512	1773.479	1993.65	1763.5	1762.82
	std	4.635324	18.90624	31.60965	12.47599	51.31799	12.04372	53.99291	87.45844	74.18254	10.6869	123.3269	6.135559	2.704442
	median	1701.121	1744.963	1740.062	1831.402	1735.7	1812.191	1863.761	1830.172	1741.97	1762.871	1843.013	1756.286	1760.561
	rank	1	2	3	10	5	9	11	12	8	7	13	4	6
C17-F18	mean	1803.022	1821.362	12583.12	6.11E+06	17541.67	12800.24	24860.12	22329.02	21212.84	31504.64	10284.31	23325.42	13609.97
	best	1800.238	1808.227	5064.639	302305.5	5784.651	7875.487	6783.878	9199.011	6651.07	25593.54	6724.069	2958.931	3554.508
	worst	1810.861	1830.204	16593.21	1.77E+07	26620.5	17334.67	39122.45	36015.86	35885.27	39435.71	12582.88	43550.36	19687.74
	std	5.221664	10.38503	5162.968	8068130	10305.49	3929.41	15565.23	12610.84	14808.73	6360.041	2497.85	20931.74	7038.805
	median	1800.495	1823.509	14337.31	3199777	18880.75	12995.41	26767.08	22050.61	21157.51	30494.65	10915.15	23396.19	15598.82
	rank	1	2	4	13	7	5	11	9	8	12	3	10	6
C17-F19	mean	1900.523	1.91E+03	7055.795	7.55E+05	6986.061	1.34E+05	3.72E+04	1915.789	5.64E+03	4.90E+03	4.32E+04	2.66E+04	6492.073
	best	1900.034	1902.859	2196.819	4.90E+04	2397.607	1.95E+03	8075.745	1910.099	1947.783	2.05E+03	1.18E+04	2676.945	2235.907
	worst	1900.849	1.91E+03	14063.29	1.62E+06	12223.18	2.69E+05	6.82E+04	1926.057	1.47E+04	1.33E+04	6.27E+04	8.23E+04	10459.09
	std	0.345586	4.05E+00	5763.931	7.08E+05	4846.576	1.53E+05	24647.74	7.523698	6.08E+03	5.56E+03	2.28E+04	3.75E+04	3388.981
	median	1900.604	1.90E+03	5981.538	6.74E+05	6661.726	1.34E+05	3.62E+04	1913.501	2.96E+03	2.14E+03	4.92E+04	1.07E+04	6636.648
	rank	1	2	8	13	7	12	10	3	5	4	11	9	6
C17-F20	mean	2008.144	2042.536	2182.848	2239.101	2148.82	2222.326	2221.485	2149.609	2182.156	2077.176	2271.99	2181.152	2053.815
	best	2000.468	2020.622	2033.57	2176.378	2119.893	2114.382	2105.405	2050.305	2140.341	2065.354	2201.309	2155.286	2038.374
	worst	2012.302	2065.244	2315.735	2298.524	2204.511	2344.047	2308.642	2265.274	2263.739	2088.349	2371.812	2215.298	2062.141
	std	5.268838	23.46593	126.7974	60.04463	37.89064	97.18204	97.04466	88.16954	55.56519	9.630337	82.86021	29.79371	10.94569
	median	2009.903	2042.138	2191.043	2240.75	2135.437	2215.437	2235.947	2141.428	2162.272	2077.501	2257.419	2177.013	2057.373
	rank	1	2	9	12	5	11	10	6	8	4	13	7	3
C17-F21	mean	2200	2257.378	2214.815	2272.021	2297.838	2334.302	2317.873	2257.022	2321.56	2306.962	2380.593	2327.466	2305.331
	best	2200	2201.776	2204.429	2225.704	2293.813	2222.779	2219.735	2200.008	2317.045	2203.991	2361.841	2318.817	2228.495

TABLE 5. *(Continued.)* **Optimization results of the CEC-2017 test suite.**

	worst	2200	2315.502	2241.859	2298.356	2302.16	2384.687	2365.29	2315.463	2326.892	2348.473	2399.185	2335.569	2342.504
	std	9.77E-10	63.27764	18.06558	32.09582	3.467277	75.54728	66.17809	65.77033	4.045273	69.06333	15.58921	8.230613	51.82018
	median	2200	2256.116	2206.485	2282.012	2297.69	2364.871	2343.233	2256.308	2321.151	2337.692	2380.674	2327.739	2325.162
	rank	1	4	2	5	6	12	9	3	10	8	13	11	7
C17-F22	mean	2300.288	2304.053	2309.64	2961.228	2307.985	2744.358	2325.549	2284.723	2309.229	2321.01	2300	2314.243	2319.245
	best	2300	2302.865	2304.685	2736.917	2301.181	2460.133	2320.545	2224.239	2301.36	2314.271	2300	2300.685	2316.136
	worst	2300.464	2306.702	2311.968	3125.843	2317.27	2967.549	2333.762	2305.661	2324.061	2333.618	2300	2348.82	2324.026
	std	0.205084	1.779339	3.343529	163.5919	6.715539	226.2123	5.902336	40.29034	10.43003	8.838917	4.78E-11	23.07451	3.366513
	median	2300.345	2303.322	2310.953	2991.077	2306.745	2774.876	2323.944	2304.496	2305.747	2318.076	2300	2303.733	2318.408
	rank	3	4	7	13	5	12	11	1	6	10	2	8	9
C17-F23	mean	2605.152	2629.183	2645.231	2708.158	2616.937	2732.714	2652.379	2621.726	2614.711	2645.742	2806.265	2647.616	2660.372
	best	2603.937	2617.394	2632.674	2676.838	2611.829	2636.949	2633.196	2607.675	2608.178	2634.093	2736.331	2639.73	2638.96
	worst	2606.764	2640.741	2664.384	2751.868	2621.692	2780.571	2673.95	2634.222	2621.981	2655.608	2955.197	2660.513	2669.42
	std	1.240791	10.54252	14.89692	35.0257	4.746674	64.79782	22.0189	11.57352	7.056325	9.545508	102.7808	9.3625	14.47615
	median	2604.954	2629.299	2641.933	2701.962	2617.113	2756.668	2651.185	2622.504	2614.343	2646.634	2766.766	2645.111	2666.553
	rank	1	5	6	11	3	12	9	4	2	7	13	8	10
C17-F24	mean	2565.244	2628.429	2777.971	2866.473	2707.981	2671.146	2770.447	2687.309	2757.72	2765.308	2756.309	2775.772	2730.119
	best	2508.339	2500.026	2749.551	2831.736	2696.258	2510.075	2750.764	2500.181	2740.206	2760.859	2500	2757.415	2523.272
	worst	2616.16	2756.728	2810.839	2925.445	2723.964	2818.253	2796.389	2761.816	2780.826	2769.47	2910.085	2790.675	2817.267
	std	58.18573	147.9519	27.60845	40.69663	11.88365	164.3451	19.39768	124.9074	17.86333	3.52295	177.381	14.02608	138.3439
	median	2568.238	2628.481	2775.747	2854.356	2705.851	2678.128	2767.317	2743.62	2754.924	2765.451	2807.575	2777.5	2789.97
	rank	1	2	12	13	5	3	10	4	8	9	7	11	6
C17-F25	mean	2915.192	2922.277	2912.059	3302.497	2929.217	3148.634	2905.665	2921.314	2939.149	2933.596	2921.498	2922.641	2953.708
	best	2897.895	2897.934	2899.173	3227.936	2917.834	2903.049	2751.205	2897.783	2919.463	2913.96	2899.585	2898.714	2941.202
	worst	2921.769	2946.585	2949.468	3386.778	2936.37	3714.425	2963.697	2945.881	2947.069	2952.974	2943.426	2946.61	2964.225
	std	11.53237	27.1265	24.91391	65.67926	7.987409	380.9019	103.0084	27.11908	13.17237	20.718	25.27439	27.08611	9.792523
	median	2920.551	2922.294	2899.797	3297.637	2931.332	2988.53	2953.878	2920.797	2945.032	2933.724	2921.49	2922.619	2954.702
	rank	3	6	2	13	8	12	1	4	10	9	5	7	11
C17-F26	mean	2862.5	2913.423	2985.864	3820.788	3160.412	3675.06	3204.325	2900.159	3292.822	3229.716	3934.065	2904.365	2897.007
	best	2850	2817.326	2800	3472.539	2943.853	3162.602	2929.263	2900.122	2974.439	2912.962	2800	2800	2692.913
	worst	2900	3035.347	3176.148	4183.224	3648.109	4372.747	3646.386	2900.208	3983.013	3949.226	4458.174	3017.461	3125.39
	std	24.9722	90.14812	214.409	306.0894	327.0156	591.086	313.1848	0.03843	463.8503	482.2908	767.4771	88.82214	218.7994
	median	2850	2900.509	2983.654	3813.695	3024.843	3582.446	3120.825	2900.153	3106.919	3028.338	4239.043	2900	2884.863
	rank	1	5	6	12	7	11	8	3	10	9	13	4	2
C17-F27	mean	3089.39	3108.049	3122.299	3241.788	3114.49	3186.307	3202.862	3091.787	3118.113	3117.018	3236.309	3139.581	3165.314
	best	3089.262	3097.47	3095.743	3130.053	3094.036	3103.401	3185.771	3089.725	3094.808	3095.827	3223.231	3097.666	3121.59
	worst	3089.518	3119.426	3187.808	3448.33	3165.297	3231.746	3215.51	3095.374	3183.353	3177.422	3259.486	3190.464	3228.683
	std	0.147771	9.897975	43.75125	140.8504	33.97803	58.08242	12.39809	2.654449	43.48797	40.23446	16.11557	38.98141	45.22737
	median	3089.39	3107.651	3102.823	3194.385	3099.313	3205.041	3205.083	3091.024	3097.146	3097.412	3231.259	3135.097	3155.491
	rank	1	3	7	13	4	10	11	2	6	5	12	8	9
C17-F28	mean	3100	3206.371	3246.182	3829.484	3306.237	3622.266	3300.63	3248.995	3363.069	3341.745	3476.706	3320.885	3257.183
	best	3100	3101.23	3100	3741.083	3202.395	3435.628	3156.592	3100.133	3201.7	3222.407	3462.464	3182.82	3148.201
	worst	3100	3383.883	3411.822	3893.295	3354.588	3846.951	3412.371	3411.823	3435.126	3412.081	3496.499	3412.053	3543.984
	std	2.07E-09	125.0449	137.7736	70.57054	70.05455	213.0956	131.3067	171.998	108.3037	90.4326	15.75328	103.8144	191.7169
	median	3100	3170.186	3236.453	3841.779	3333.982	3603.242	3316.778	3242.012	3407.726	3366.247	3473.932	3344.334	3168.273
	rank	1	2	3	13	7	12	6	4	10	9	11	8	5
C17-F29	mean	3136.88	3179.875	3296.049	3393.557	3248.593	3244.114	3365.261	3208.026	3275.244	3218.734	3362.024	3276.182	3245.186
	best	3131.922	3164.605	3216.476	3316.959	3186.363	3168.556	3243.693	3143.443	3194.144	3168.356	3241.182	3170.757	3192.605
	worst	3142.083	3205.527	3382.815	3465.973	3330.312	3319.615	3523.176	3298.305	3397.893	3242.962	3673.067	3365.159	3298.151
	std	4.1702	18.63679	85.53742	76.69589	66.72629	61.76374	117.1158	65.54629	96.7765	34.97041	208.0018	88.24954	44.32259
	median	3136.758	3174.685	3292.453	3395.648	3238.849	3244.142	3347.088	3195.179	3254.47	3231.809	3266.924	3284.406	3244.995
	rank	1	2	10	13	7	5	12	3	8	4	11	9	6
C17-F30	mean	3408.483	7.10E+03	315325.2	3.94E+06	743329.5	6.58E+05	1.06E+06	3.24E+05	1.00E+06	6.47E+04	8.38E+05	4.14E+05	1.64E+06
	best	3394.769	4.03E+03	111836.1	8.86E+05	26515.96	1.20E+05	4.54E+03	7725.682	3.57E+04	3.11E+04	6.44E+05	6.60E+03	5.63E+05
	worst	3425.209	1.57E+04	821835.5	6.22E+06	1082695	1.39E+06	4.01E+06	1.24E+06	1.45E+06	1.09E+05	1.07E+06	8.22E+05	3.73E+06
	std	15.82787	5.75E+03	338216.9	2.23E+06	487446.5	5.39E+05	1.97E+06	607606.9	6.64E+05	3.79E+04	1.77E+05	4.69E+05	1489000
	median	3406.978	4.32E+03	163814.6	4.32E+06	932053.7	5.60E+05	1.17E+05	2.62E+04	1.26E+06	5.94E+04	8.19E+05	4.15E+05	1.13E+06
	rank	1	2	4	13	8	7	11	5	10	3	9	6	12

TABLE 5. ***(Continued.)*** **Optimization results of the CEC-2017 test suite.**

Sum rank	35	90	199	352	165	302	262	132	213	217	249	206	217
Mean rank	1.206897	3.103448	6.862069	12.13793	5.689655	10.41379	9.034483	4.551724	7.344828	7.482759	8.586207	7.103448	7.482759
Total rank	1	2	5	12	4	11	9	3	7	8	10	6	8

WOA, TSA, MPA, RSA, WSO, and AVOA. The values of the control parameters for these competitor algorithms are specified in Table 1. Simulation results are reported using six statistical indicators: mean, best, worst, standard deviation (std), median, and rank. The ranking criterion for metaheuristic algorithms in solving each benchmark function is to provide a better value for the mean index.

A. EVALUATION OF UNIMODAL TEST FUNCTIONS

Seven benchmark functions F1 to F7 are selected from the unimodal type. Because these functions have no local optima, they are suitable options for evaluating the exploitation power of metaheuristic algorithms. The optimization results for the functions F1 to F7 using RPO and the competitor algorithms are presented in Table 2. Based on the optimization results, RPO with high exploitation ability has converged to the global optima in solving functions F1, F2, F3, F4, F5, and F6. In solving the function F7, RPO is the first-best optimizer. The analysis of the simulation results shows that the proposed RPO approach has provided better results and in total by winning the first rank, it has achieved a superior performance in the optimization of unimodal benchmark functions compared to the competitor algorithms.

B. EVALUATION OF HIGH-DIMENSIONAL MULTIMODAL TEST FUNCTIONS

Six benchmark functions F8 to F13 are selected from high-dimensional multimodal type. In addition to the global optima, these functions have a large number of local optima, and for this reason, they are suitable options for evaluating the exploration power of metaheuristic algorithms. The implementation results of RPO and the competitor algorithms for the functions F8 to F13 are reported in Table 3. The optimization results show that RPO with high exploration ability has converged to the global optima in the optimization of F9 and F11 functions in addition to identifying the main optimal area in the search space. In solving the functions F8, F10, F12, and F13, RPO has provided suitable solutions with high exploration ability and is the first-best optimizer for these functions. The comparison of the simulation results indicates that the proposed RPO approach, with a high exploration ability in the case of high-dimensional multimodal functions, has obtained superior performance over the competitor algorithms.

C. EVALUATION OF FIXED-DIMENSIONAL MULTIMODAL TEST FUNCTIONS

Ten benchmark functions F14 to F23 have been selected from the fixed-dimensional multimodal type. These functions, compared to functions F8 to F13, have a lower number of local optima. Functions F14 to F23 are suitable options for evaluating the ability of metaheuristic algorithms in balancing exploration and exploitation features during the search process. The results of using RPO and the competitor algorithms for optimizing the functions F14 to F23 are presented in Table 4. Based on the optimization results, RPO is the first-best optimizer for the functions F14, F15, F21, F22, and F23. In solving the functions F16, F17, F18, F19, and F20, RPO has the same conditions as some of the competing algorithms considering the mean index criterion. However, RPO has provided a more effective performance in handling these functions by providing better results from the std index viewpoint. What is evident from the analysis of simulation results, RPO has achieved better results in solving fixed-dimensional multimodal functions with an appropriate ability to balance exploration and exploitation, and compared to the competitor algorithms, it has provided superior performance in optimizing these functions.

The performance of RPO and the competitor algorithms in solving benchmark functions F1 to F23 is illustrated in the form of convergence curves in Figure 3.

D. EVALUATION OF CEC 2017 TEST SUITE

RPO's performance in solving optimization problems is evaluated on CEC 2017 test suite. This test suite has thirty benchmark functions consisting of three unimodal functions of C17-F1 to C17-F3, seven multimodal functions of C17-F4 to C17-F10, ten hybrid functions of C17-F11 to C17-F20, and ten composition functions of C17-F21 to C17-F30. The C17-F2 function has been excluded from the simulation studies due to its unstable behavior. The full description of CEC 2017 test suite is provided in [86]. The implementation results of RPO and the competitor algorithms on CEC 2017 test suite are reported in Table 5. Based on the optimization result, RPO is the first best optimizer for functions C17-F1, C17-F4 to C17-F8, C17-F10 to C17-F21, C17-F23, C17-F24, and C17-F26 to C17-F30. The performance of RPO and the competitor algorithms in solving the CEC 2017 test suite is drawn as boxplot diagrams in Figure 4. Analysis of the simulation results shows that RPO has provided better results for most of the benchmark functions of CEC 2017 test suite and overall, by winning the first rank, it has provided superior performance over the competitor algorithms in solving CEC 2017 test suite.

E. STATISTICAL ANALYSIS

In this subsection, statistical analysis is presented on the performance of RPO and the competitor algorithms to determine

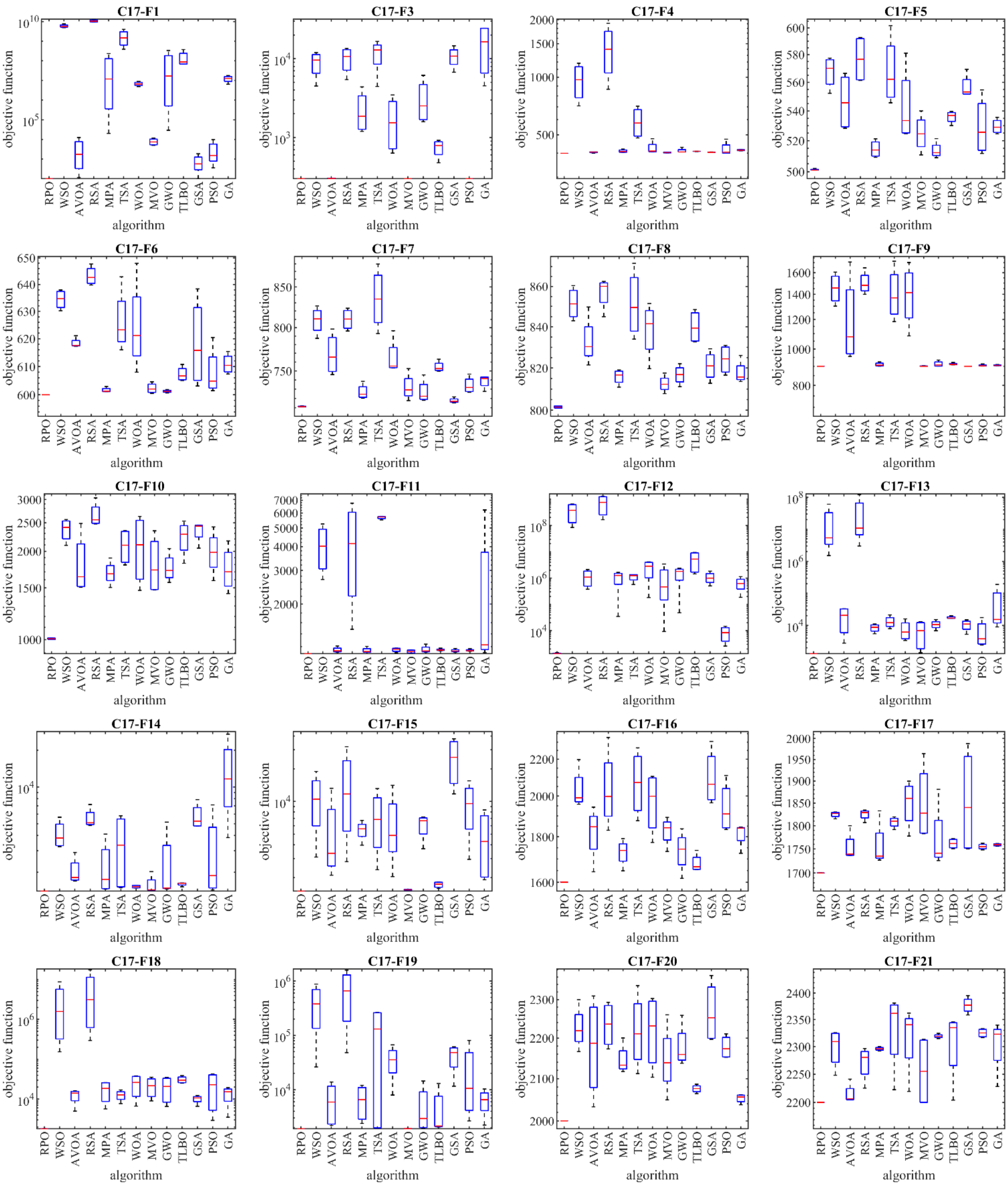

FIGURE 4. Boxplot diagrams of RPO and the competitor algorithms performances for the CEC-2017 test suite.

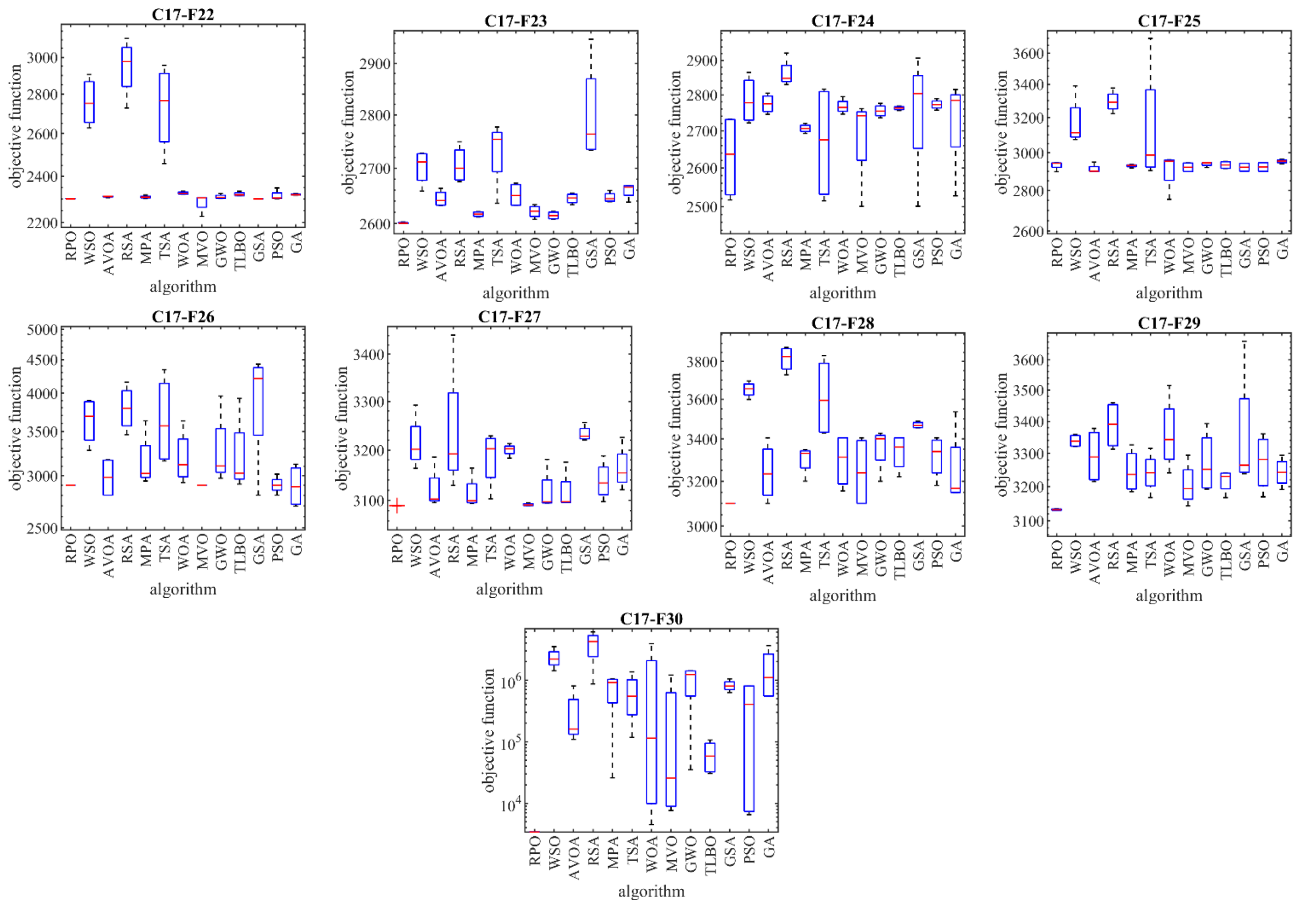

FIGURE 4. *(Continued.)* **Boxplot diagrams of RPO and the competitor algorithms performances for the CEC-2017 test suite.**

whether RPO has a significant statistical superiority or not. For this purpose, Wilcoxon rank sum test [87] is employed, which is a non-parametric statistical test to determine the significant difference between the average of two data samples. In Wilcoxon rank sum test, an index called p-value is utilized to evaluate whether RPO is significantly superior over any of the competitor algorithms from a statistical point of view.

The results of the statistical analysis on the performance of RPO and the competitor algorithms are reported in Table 6. Based on the results, in cases where the p-value is less than 0.05, RPO has a significant statistical superiority over the corresponding competitor algorithm.

TABLE 6. ***p*-values obtained from Wilcoxon rank sum test.**

Compared algorithm	Objective function type			
	F1-F7	F8-F13	F14-F23	CEC 2017
RPO vs. WSO	1.08E-24	1.97E-21	6.49E-12	1.14E-17
RPO vs. AVOA	5.03E-17	0.000129	3.76E-21	9.21E-20
RPO vs. RSA	5.06E-09	3.82E-08	1.44E-34	1.97E-21
RPO vs. MPA	1.01E-24	4.66E-09	9.13E-33	2.47E-21
RPO vs. TSA	1.01E-24	2.78E-20	1.44E-34	4.62E-21
RPO vs. WOA	1.01E-24	1.04E-07	1.44E-34	8.39E-21
RPO vs. MVO	1.01E-24	1.97E-21	1.44E-34	5.13E-19
RPO vs. GWO	1.01E-24	7.84E-16	1.44E-34	2.67E-21
RPO vs. TLBO	1.01E-24	6.98E-15	1.44E-34	2.29E-21
RPO vs. GSA	1.01E-24	3.05E-19	1.03E-13	7.1E-19
RPO vs. PSO	1.01E-24	1.97E-21	9.75E-17	2.14E-20
RPO vs. GA	1.01E-24	2.14E-20	1.44E-34	1.8E-19

V. RPO FOR REAL-WORLD APPLICATION

In this section, the effectiveness of the proposed RPO approach for solving optimization problems in real-world applications is tested. In this regard, RPO is implemented on four engineering design problems.

A. PRESSURE VESSEL DESIGN PROBLEM

Pressure vessel design is an engineering minimization problem with the aim of reducing design cost. The schematic of this design is presented in Figure 5. Pressure vessel design mathematical model is as follows [88]:

$$\begin{aligned}
&\mathit{Consider}\colon X = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L]\,.\\
&\mathit{Minimize}\colon f(x) = 0.6224x_1x_3x_4 + 1.778x_2x_3^2\\
&\qquad\qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3.\\
&\mathit{Subject\ to}\colon g_1(x) = -x_1 + 0.0193x_3 \le 0,\\
&\qquad\qquad g_2(x) = -x_2 + 0.00954x_3 \le 0,
\end{aligned}$$

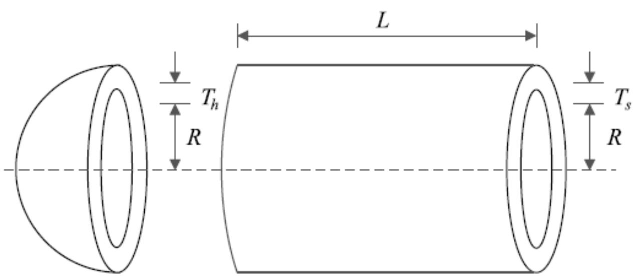

FIGURE 5. Schematic view of pressure vessel design problem.

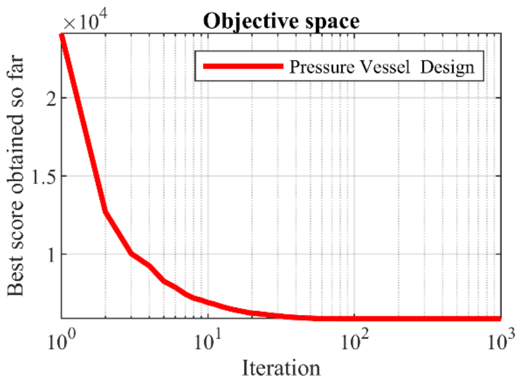

FIGURE 6. Convergence analysis of the RPO for the pressure vessel design optimization problem.

$$g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0,$$
$$g_4(x) = x_4 - 240 \leq 0.$$

with

$$0 \leq x_1,\ x_2 \leq 100 \quad \text{and } 10 \leq x_3,\ x_4 \leq 200.$$

The results of implementing RPO and the competitor algorithms for solving pressure vessel design problem are reported in Tables 7 and 8. Based on the obtained results, RPO has provided the optimal solution of this design with the design values equal to (0.778027, 0.384579, 40.31228, 200) and the corresponding objective function value is equal to (5882.895). The convergence curve of RPO while achieving the solution for pressure vessel design is drawn in Figure 6. Based on the simulation results, RPO has provided superior performance in pressure vessel design optimization compared to the competitor algorithms.

B. SPEED REDUCER DESIGN PROBLEM

Speed reducer design is an engineering minimization problem with the aim of reducing the weight of the speed reducer. The schematic of this design is illustrated in Figure 7. The mathematical model of speed reducer design is as follows [89], [90]:

$$Consider: X = [x_1, x_2, x_3, x_4, x_5, x_6, x_7] = [b, M, p, l_1, l_2, d_1, d_2].$$

$$Minimize: f(x) = 0.7854 x_1 x_2^2$$

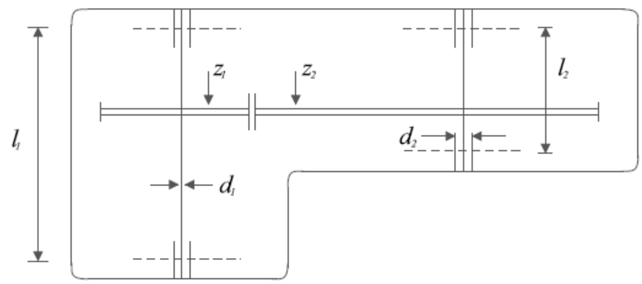

FIGURE 7. Schematic view of speed reducer design problem.

$$\times \left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777 \times \left(x_6^3 + x_7^3\right) + 0.7854(x_4x_6^2 + x_5x_7^2).$$

$$Subject\ to: g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0,$$
$$g_2(x) = \frac{397.5}{x_1x_2^2x_3} - 1 \leq 0,$$
$$g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0,$$
$$g_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0,$$
$$g_5(x) = \frac{1}{110x_6^3}\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \cdot 10^6} - 1 \leq 0,$$
$$g_6(x) = \frac{1}{85x_7^3}\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5 \cdot 10^6} - 1 \leq 0,$$
$$g_7(x) = \frac{x_2xn_3}{40} - 1 \leq 0,$$
$$g_8(x) = \frac{5x_2}{x_1} - 1 \leq 0,$$
$$g_9(x) = \frac{x_1}{12x_2} - 1 \leq 0,$$
$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0,$$
$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0.$$

with

$$2.6 \leq x_1 \leq 3.6, \quad 0.7 \leq x_2 \leq 0.8, \quad 17 \leq x_3 \leq 28,$$
$$7.3 \leq x_4 \leq 8.3, \quad 7.8 \leq x_5 \leq 8.3, \quad 2.9 \leq x_6 \leq 3.9,$$
$$\text{and } 5 \leq x_7 \leq 5.5.$$

Speed reducer design optimization results using RPO and the competitor algorithms are reported in Tables 9 and 10. Based on the obtained results, RPO has provided the optimal solution of this design with the design values equal to (3.5, 0.7, 17, 7.3, 7.8, 3.350215, 5.286683) and the corresponding

TABLE 7. Performance of optimization algorithms for pressure vessel design problem.

Algorithm	Optimum variables				Optimum cost
	T_s	T_h	R	L	
RPO	0.778027	0.384579	40.31228	200	5882.895
WSO	0.778027	0.384579	40.31228	200	5882.901
AVOA	0.778032	0.384581	40.31252	199.9967	5882.909
RSA	1.291554	0.699601	65.23446	12.84939	8194.355
MPA	0.778027	0.384579	40.31228	200	5882.901
TSA	0.77984	0.386106	40.4037	200	5915.504
WOA	0.942422	0.466466	47.60099	118.5224	6360.68
MVO	0.847484	0.423773	43.89043	155.5507	6031.851
GWO	0.77856	0.386098	40.32256	199.9562	5891.903
TLBO	1.743819	0.503695	49.40541	107.2033	11947.44
GSA	1.211502	1.336607	44.9893	188.6548	13397
PSO	1.728796	0.678351	68.42428	15.00819	10951.49
GA	1.551898	0.87558	62.40696	44.7856	12086.51

TABLE 8. Statistical results of optimization algorithms for pressure vessel design problem.

Algorithm	mean	best	worst	std	median	rank
RPO	5882.895	5882.895	5882.895	1.87E-12	5882.895	1
WSO	5893.153	5882.901	5984.052	27.36291	5882.902	3
AVOA	6297.475	5882.909	7315.667	433.9341	6085.848	5
RSA	13920.6	8194.355	23258.16	3851.022	12681.39	9
MPA	5882.901	5882.901	5882.901	4.53E-06	5882.901	2
TSA	6361.012	5915.504	7195.051	410.2889	6203.974	6
WOA	8488.458	6360.68	14408.04	2071.279	7972.449	8
MVO	6665.159	6031.851	7320.699	394.477	6731.769	7
GWO	6042.341	5891.903	6853.448	294.8064	5902.172	4
TLBO	33457.01	11947.44	72912.61	16989.52	29395.68	12
GSA	24060.88	13397	38176.31	8265.507	23059.42	10
PSO	35198.67	10951.49	61090.9	15905.38	38920.01	13
GA	29952.66	12086.51	54709.96	13335.83	26410.2	11

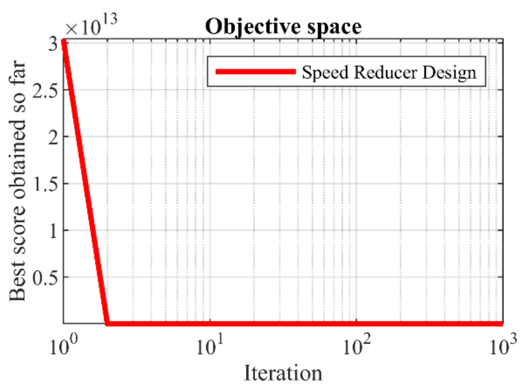

FIGURE 8. Convergence analysis of the RPO for the speed reducer design optimization problem.

objective function value is equal to (2996.348). The RPO convergence curve during solving the speed reducer design is drawn in Figure 8. Analysis of the simulation results shows that RPO is superior over the competitor algorithms by providing better results in the optimization of speed reducer design.

C. WELDED BEAM DESIGN PROBLEM

Welded beam design is a real-world application with the aim of minimizing the fabrication cost of the welded beam. The schematic of this design is shown in Figure 9. The mathematical model of the welded beam design is as follows [40]:

$$\begin{aligned}
&Consider: X = [x_1, x_2, x_3, x_4] = [h, l, t, b]\,.\\
&Minimize: f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2).\\
&Subject\ to: g_1(x) = \tau(x) - 13600 \le 0,\\
&\qquad g_2(x) = \sigma(x) - 30000 \le 0,\\
&\qquad g_3(x) = x_1 - x_4 \le 0,\\
&\qquad g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2)\\
&\qquad\qquad - 5.0 \le 0,\\
&\qquad g_5(x) = 0.125 - x_1 \le 0, \quad g_6(x) = \delta(x)\\
&\qquad\qquad - 0.25 \le 0,\\
&\qquad g_7(x) = 6000 - p_c(x) \le 0.
\end{aligned}$$

where

$$\tau(x) = \sqrt{(\tau')^2 + (2\tau\tau')\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{6000}{\sqrt{2}x_1x_2},$$

$$\tau'' = \frac{MR}{J}, \quad M = 6000\left(14 + \frac{x_2}{2}\right),$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},$$

TABLE 9. Performance of optimization algorithms for speed reducer design problem.

Algorithm	Optimum variables							Optimum cost
	b	M	p	l_1	l_2	d_1	d_2	
RPO	3.5	0.7	17	7.3	7.8	3.350215	5.286683	2996.348
WSO	3.500001	0.7	17	7.300011	7.8	3.350215	5.286683	2996.348
AVOA	3.5	0.7	17	7.300001	7.8	3.350215	5.286683	2996.348
RSA	3.6	0.7	17	8.3	8.3	3.356126	5.5	3198.674
MPA	3.5	0.7	17	7.3	7.8	3.350215	5.286683	2996.348
TSA	3.513994	0.7	17	7.3	8.3	3.350568	5.290516	3015.365
WOA	3.594903	0.7	17	7.3	8.027113	3.36258	5.286762	3041.81
MVO	3.502443	0.7	17	7.3	8.091898	3.371241	5.286899	3009.244
GWO	3.500696	0.7	17	7.30558	7.8	3.365114	5.288991	3001.953
TLBO	3.560863	0.704337	27.11575	8.169454	8.174528	3.690042	5.343833	5463.451
GSA	3.524856	0.702987	17.4005	7.864753	7.897223	3.41375	5.394368	3184.453
PSO	3.508879	0.700078	18.18874	7.407452	7.87381	3.616286	5.348895	3328.551
GA	3.584642	0.706038	17.88296	7.780183	7.860589	3.731411	5.351402	3376.635

TABLE 10. Statistical results of optimization algorithms for speed reducer design problem.

Algorithm	mean	best	worst	std	median	rank
RPO	2996.348	2996.348	2996.348	9.33E-13	2996.348	1
WSO	2996.656	2996.348	2999.008	0.634541	2996.366	3
AVOA	3001.239	2996.348	3012.326	4.30514	3001.131	4
RSA	3300.61	3198.674	3363.873	62.40256	3316.752	9
MPA	2996.348	2996.348	2996.348	3.46E-06	2996.348	2
TSA	3035.173	3015.365	3050.071	11.00124	3037.113	7
WOA	3163.113	3041.81	3483.243	115.3297	3126.942	8
MVO	3032.667	3009.244	3076.451	14.38386	3033.144	6
GWO	3005.324	3001.953	3011.796	2.720315	3004.763	5
TLBO	7.55E+13	5463.451	5.46E+14	1.26E+14	2.96E+13	12
GSA	3493.588	3184.453	4167.382	284.4832	3352.923	10
PSO	1.11E+14	3328.551	5.64E+14	1.35E+14	7.97E+13	13
GA	5.36E+13	3376.635	3.46E+14	8.45E+13	2.15E+13	11

$$J = 2\sqrt{2}\, x_1 x_2 \left(\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right), \quad \sigma(x) = \frac{504000}{x_4 x_3^2},$$

$$\delta(x) = \frac{65856000}{(30 \cdot 10^6)\, x_4 x_3^3},$$

$$p_c(x) = \frac{4.013\,(30 \cdot 10^6)\, x_3 x_4^3}{1176}\left(1 - \frac{x_3}{28}\sqrt{\frac{30 \cdot 10^6}{4(12 \cdot 10^6)}}\right).$$

with

$$0.1 \leq x_1,\ x_4 \leq 2 \quad \text{and}\ 0.1 \leq x_2,\ x_3 \leq 10.$$

The results of employing RPO and the competitor algorithms in handling the welded beam design problem are presented in Tables 11 and 12. Based on the obtained results, RPO has provided the optimal solution of this design with the design values equal to (0.20573, 3.470489, 9.036624, 0.20573) and the corresponding objective function value is equal to (1.72468). The convergence curve of RPO while reaching the solution for welded beam design is drawn in Figure 10. What is evident from the simulation results, RPO has a higher ability compared to the competitor algorithms in dealing with the welded beam design problem.

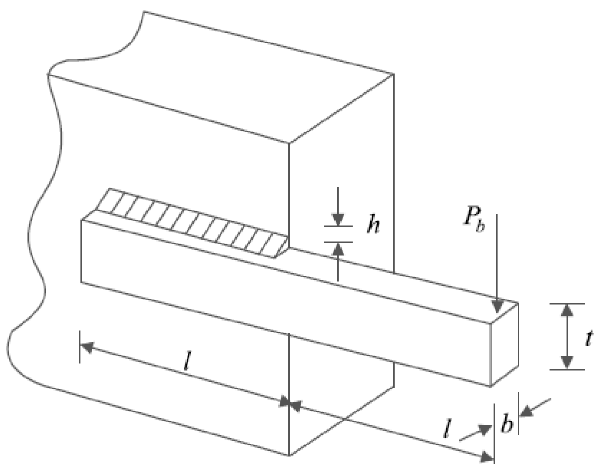

FIGURE 9. Schematic view of the welded beam design problem.

D. TENSION/COMPRESSION SPRING DESIGN PROBLEM

Tension/compression spring design is a real-world application aimed at minimizing the weight of tension/compression spring. The schematic of this design is presented in Figure 11. The mathematical model of tension/compression spring design is as follows [40]:

$$Consider: X = [x_1, x_2, x_3] = [d, D, P].$$

TABLE 11. Performance of optimization algorithms for welded beam design problem.

Algorithm	Optimum variables				Optimum cost
	h	l	t	b	
RPO	0.20573	3.470489	9.036624	0.20573	1.72468
WSO	0.20573	3.470489	9.036624	0.20573	1.724852
AVOA	0.2049	3.48848	9.036508	0.205735	1.72601
RSA	0.19593	3.540107	10	0.218818	1.99664
MPA	0.20573	3.470489	9.036624	0.20573	1.724852
TSA	0.204066	3.497483	9.06652	0.206192	1.734605
WOA	0.214405	3.317836	8.968509	0.222288	1.829474
MVO	0.206015	3.46433	9.045367	0.206083	1.728662
GWO	0.20558	3.473912	9.036208	0.205805	1.72558
TLBO	0.324507	4.501924	6.608532	0.443622	3.133294
GSA	0.301279	2.658468	7.284759	0.316577	2.11484
PSO	0.386623	3.420812	7.201733	0.60504	4.216826
GA	0.225878	7.205324	7.655917	0.312695	2.848415

TABLE 12. Statistical results of optimization algorithms for welded beam design problem.

Algorithm	mean	best	worst	std	median	rank
RPO	1.72468	1.72468	1.72468	2.28E-16	1.72468	1
WSO	1.724853	1.724852	1.724858	1.36E-06	1.724852	3
AVOA	1.764281	1.72601	1.852621	0.039545	1.749218	7
RSA	2.22015	1.99664	2.597104	0.156301	2.193001	8
MPA	1.724852	1.724852	1.724852	3.64E-09	1.724852	2
TSA	1.744692	1.734605	1.75466	0.006079	1.744797	6
WOA	2.360132	1.829474	4.241941	0.695843	2.116216	9
MVO	1.742604	1.728662	1.779283	0.014917	1.73819	5
GWO	1.727455	1.72558	1.73184	0.001478	1.727189	4
TLBO	3.61E+13	3.133294	3.48E+14	8.8E+13	6.026046	12
GSA	2.504335	2.11484	2.838938	0.20767	2.536371	10
PSO	4.97E+13	4.216826	3.01E+14	9.5E+13	7.152137	13
GA	1.22E+13	2.848415	1.32E+14	3.75E+13	5.989825	11

TABLE 13. Performance of optimization algorithms for tension/compression spring design problem.

Algorithm	Optimum variables			Optimum cost
	d	D	P	
RPO	0.051689	0.356718	11.28897	0.012602
WSO	0.051687	0.356666	11.292	0.012665
AVOA	0.05115	0.343882	12.08318	0.012671
RSA	0.05	0.310577	15	0.0132
MPA	0.051691	0.356762	11.28639	0.012665
TSA	0.05093	0.338698	12.43709	0.012683
WOA	0.051122	0.343232	12.12573	0.012671
MVO	0.05	0.316862	14.10401	0.012757
GWO	0.051979	0.363704	10.89486	0.012671
TLBO	0.069083	0.936806	2	0.017884
GSA	0.055399	0.448234	7.528543	0.013108
PSO	0.068994	0.933432	2	0.017773
GA	0.06959	0.945261	2	0.018311

$$Minimize: f(x) = (x_3 + 2)\, x_2 x_1^2.$$

$$Subject\ to: g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0,$$

$$g_2(x) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3)} + \frac{1}{5108 x_1^2} - 1 \leq 0,$$

$$g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0,$$

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0.$$

with

$$0.05 \leq x_1 \leq 2, \quad 0.25 \leq x_2 \leq 1.3 \quad \text{and } 2 \leq x_3 \leq 15.$$

The implementation results of RPO and the competitor algorithms for the tension/compression spring design problem are reported in Tables 13 and 14. Based on the obtained

TABLE 14. Statistical results of optimization algorithms for tension/compression spring design problem.

Algorithm	mean	best	worst	std	median	rank
RPO	0.012602	0.012602	0.012602	6.88E-18	0.012602	1
WSO	0.012677	0.012665	0.012837	3.84E-05	0.012666	3
AVOA	0.01339	0.012671	0.014257	0.000596	0.013317	8
RSA	0.013287	0.0132	0.013441	7.42E-05	0.013264	6
MPA	0.012665	0.012665	0.012665	3.05E-09	0.012665	2
TSA	0.012983	0.012683	0.013586	0.000258	0.012904	5
WOA	0.013315	0.012671	0.014627	0.000646	0.013103	7
MVO	0.016741	0.012757	0.018279	0.001762	0.017721	9
GWO	0.012727	0.012671	0.012966	5.92E-05	0.012724	4
TLBO	0.018452	0.017884	0.019102	0.000383	0.018405	10
GSA	0.019897	0.013108	0.033424	0.004558	0.019441	11
PSO	2.24E+13	0.017773	3.97E+14	8.89E+13	0.017773	13
GA	1.75E+12	0.018311	1.81E+13	5.22E+12	0.026458	12

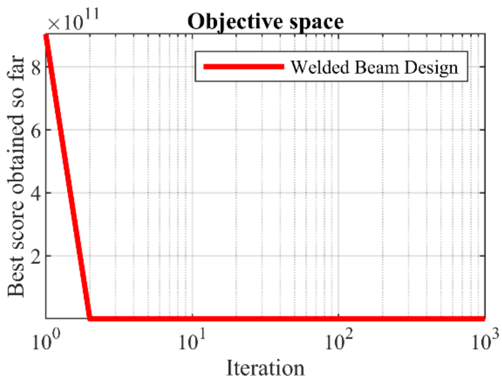

FIGURE 10. Convergence analysis of the RPO for the welded beam design optimization problem.

FIGURE 11. Schematic view of tension/compression spring design problem.

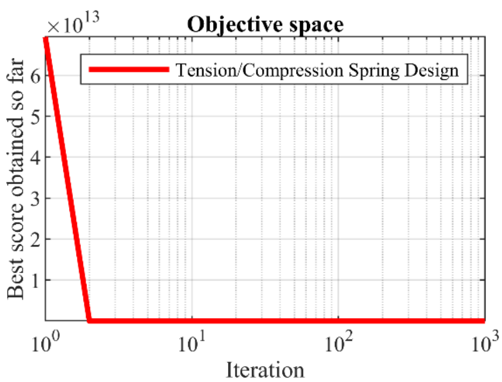

FIGURE 12. Convergence analysis of the RPO for the tension/compression spring design optimization problem.

results, RPO has provided the optimal solution of this design with the design values equal to (0.051689, 0.356718, 11.28897) and the corresponding objective function value is equal to (0.012602). The convergence curve of RPO while achieving the optimal solution for tension/compression spring design is drawn in Figure 12. What can be concluded from the simulation results, RPO has provided a more effective performance compared to the competitor algorithms in solving the tension/compression spring design problem.

VI. CONCLUSION AND FUTURE WORKS

In this paper, a new bio-inspired metaheuristic algorithm called Red Panda Optimization (RPO) was introduced, which can be applied to solve optimization problems. The fundamental inspiration of RPO is simulation of the behavior of red pandas when foraging and their ability to climb trees to rest. The implementation steps of RPO were described and mathematically modeled in two phases of exploration and exploitation. The effectiveness of RPO in solving optimization problems was evaluated considering fifty-two benchmark functions consisting of unimodal, high-dimensional multimodal, fixed-dimensional multimodal, and CEC 2017 test suite. The optimization results of unimodal functions indicated the high ability of RPO in local search and exploitation. The optimization results of multimodal functions showed that RPO has a high ability in global search and exploration. Also, the optimization results of CEC 2017 test suite showed the high capability of the proposed RPO approach in providing simultaneous exploration and exploitation during the search process. The results obtained from the implementation of RPO were compared with the performance of twelve well-known metaheuristic algorithms. The simulation results showed that the proposed RPO approach by balancing exploration and exploitation features during the search process, has provided superior performance over the competitor algorithms. Based on the simulation results, the proposed RPO approach provided better results compared to the competitor algorithms in 100% of unimodal functions, 100% of high-dimensional multimodal functions, 100% of fixed-dimensional multimodal functions, and 86.2% of CEC 2017 test suite benchmark functions. In addition, the implementation of RPO on four engineering design problems showed that the proposed algorithm has a high ability to solve optimization problems in real-world applications.

Following the introduction of the proposed RPO approach, several research paths are activated for further studies. The development of binary and multi-objective versions of RPO is one of the most significant research potentials in this regard. The use of RPO for solving optimization problems in various fields of science as well as optimization tasks in real-world applications is one of the other suggestions of this paper for future investigations.

ACKNOWLEDGMENT

The authors thank Štěpán Hubálovský from the University of Hradec Králové for fruitful and informative discussions.

REFERENCES

[1] H. Xian, C. Yang, H. Wang, and X. Yang, "A modified sine cosine algorithm with teacher supervision learning for global optimization," *IEEE Access*, vol. 9, pp. 17744–17766, 2021.

[2] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, "Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 7, pp. 1501–1529, Jul. 2020.

[3] A. S. Assiri, A. G. Hussien, and M. Amin, "Ant lion optimization: Variants, hybrids, and applications," *IEEE Access*, vol. 8, pp. 77746–77764, 2020.

[4] P. Coufal, Š. Hubálovský, M. Hubálovská, and Z. Balogh, "Snow leopard optimization algorithm: A new nature-based optimization algorithm for solving optimization problems," *Mathematics*, vol. 9, no. 21, p. 2832, Nov. 2021.

[5] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: Sparrow search algorithm," *Syst. Sci. Control Eng.*, vol. 8, no. 1, pp. 22–34, Jan. 2020.

[6] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017.

[7] S. Mirjalili, "The ant lion optimizer," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, May 2015.

[8] M. Cavazzuti, "Deterministic optimization," *Optimization Methods: From Theory to Design Scientific and Technological Aspects in Mechanics*. Berlin, Germany: Springer, 2013, pp. 77–102.

[9] K. Iba, "Reactive power optimization by genetic algorithm," *IEEE Trans. Power Syst.*, vol. 9, no. 2, pp. 685–692, May 1994.

[10] S. S. Mohar, S. Goyal, and R. Kaur, "Localization of sensor nodes in wireless sensor networks using bat optimization algorithm with enhanced exploration and exploitation characteristics," *J. Supercomput.*, vol. 78, no. 9, pp. 11975–12023, Jun. 2022.

[11] G. Brunetti, C. Stumpp, and J. Šimunek, "Balancing exploitation and exploration: A novel hybrid global-local optimization strategy for hydrological model calibration," *Environ. Model. Softw.*, vol. 150, Apr. 2022, Art. no. 105341.

[12] M. Dehghani, Z. Montazeri, A. Dehghani, and O. P. Malik, "GO: Group optimization," *Gazi Univ. J. Sci.*, vol. 33, no. 2, pp. 381–392, Jun. 2020.

[13] M. Dehghani, Z. Montazeri, and O. P. Malik, "Energy commitment: A planning of energy carrier based on energy consumption," *Electr. Eng. Electromech.*, no. 4, pp. 69–72, Aug. 2019.

[14] M. Dehghani, M. Mardaneh, O. P. Malik, J. M. Guerrero, C. Sotelo, D. Sotelo, M. Nazari-Heris, K. Al-Haddad, and R. A. Ramirez-Mendoza, "Genetic algorithm for energy commitment in a power system supplied by multiple energy carriers," *Sustainability*, vol. 12, no. 23, p. 10053, Dec. 2020.

[15] M. Dehghani, M. Mardaneh, O. P. Malik, J. M. Guerrero, R. Morales-Menendez, R. A. Ramirez-Mendoza, J. Matas, and A. Abusorrah, "Energy commitment for a power system supplied by multiple energy carriers system using following optimization algorithm," *Appl. Sci.*, vol. 10, no. 17, p. 5862, Aug. 2020.

[16] H. Rezk, A. Fathy, M. Aly, and M. N. Ibrahim, "Energy management control strategy for renewable energy system based on spotted hyena optimizer," *Comput., Mater. Continua*, vol. 67, no. 2, pp. 2271–2281, 2021.

[17] A. Ehsanifar, M. Dehghani, and M. Allahbakhshi, "Calculating the leakage inductance for transformer inter-turn fault detection using finite element method," in *Proc. Iranian Conf. Electr. Eng. (ICEE)*, May 2017, pp. 1372–1377.

[18] M. Dehghani, Z. Montazeri, A. Ehsanifar, A. R. Seifi, M. J. Ebadi, and O. M. Grechko, "Planning of energy carriers based on final energy consumption using dynamic programming and particle swarm optimization," *Electr. Eng. Electromech.*, no. 5, pp. 62–71, Oct. 2018.

[19] Z. Montazeri and T. Niknam, "Energy carriers management based on energy consumption," in *Proc. IEEE 4th Int. Conf. Knowl.-Based Eng. Innov. (KBEI)*, Dec. 2017, pp. 539–543.

[20] M. Dehghani, Z. Montazeri, and O. P. Malik, "Optimal sizing and placement of capacitor banks and distributed generation in distribution systems using spring search algorithm," *Int. J. Emerg. Electr. Power Syst.*, vol. 21, no. 1, pp. 1–9, Jan. 2020.

[21] M. Dehghani, Z. Montazeri, O. P. Malik, K. Al-Haddad, J. M. Guerrero, and G. Dhiman, "A new methodology called dice game optimizer for capacitor placement in distribution systems," *Electr. Eng. Electromech.*, no. 1, pp. 61–64, Feb. 2020.

[22] S. Dehbozorgi, A. Ehsanifar, Z. Montazeri, M. Dehghani, and A. Seifi, "Line loss reduction and voltage profile improvement in radial distribution networks using battery energy storage system," in *Proc. IEEE 4th Int. Conf. Knowl.-Based Eng. Innov. (KBEI)*, Dec. 2017, pp. 215–219.

[23] Z. Montazeri and T. Niknam, "Optimal utilization of electrical energy from power plants based on final energy consumption using gravitational search algorithm," *Electr. Eng. Electromech.*, no. 4, pp. 70–73, Aug. 2018.

[24] M. Dehghani, M. Mardaneh, Z. Montazeri, A. Ehsanifar, M. J. Ebadi, and O. M. Grechko, "Spring search algorithm for simultaneous placement of distributed generation and capacitors," *Electr. Eng. Electromech.*, no. 6, pp. 68–73, Dec. 2018.

[25] M. Premkumar, R. Sowmya, P. Jangir, K. S. Nisar, and M. Aldhaifallah, "A new metaheuristic optimization algorithms for brushless direct current wheel motor design problem," *Comput., Mater. Continua*, vol. 67, no. 2, pp. 2227–2242, 2021.

[26] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[27] M. Dehghani and P. Trojovský, "Serval optimization algorithm: A new bio-inspired approach for solving optimization problems," *Biomimetics*, vol. 7, no. 4, p. 204, Nov. 2022.

[28] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, vol. 4, 1995, pp. 1942–1948.

[29] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996.

[30] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Proc. Int. Fuzzy Syst. Assoc. World Congr.*, 2007, pp. 789–798.

[31] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," 2010, *arXiv:1003.1409*.

[32] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 259, Jan. 2023, Art. no. 110011.

[33] N. Chopra and M. M. Ansari, "Golden jackal optimization: A novel nature-inspired optimizer for engineering applications," *Expert Syst. Appl.*, vol. 198, Jul. 2022, Art. no. 116924.

[34] M. Braik, A. Hammouri, J. Atwan, M. A. Al-Betar, and M. A. Awadallah, "White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems," *Knowl.-Based Syst.*, vol. 243, May 2022, Art. no. 108457.

[35] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: A nature-inspired metaheuristic," *Expert Syst. Appl.*, vol. 152, Aug. 2020, Art. no. 113377.

[36] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems," *Comput. Ind. Eng.*, vol. 158, Aug. 2021, Art. no. 107408.

[37] P. Trojovský and M. Dehghani, "Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications," *Sensors*, vol. 22, no. 3, p. 855, Jan. 2022.
[38] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization," *Eng. Appl. Artif. Intell.*, vol. 90, Apr. 2020, Art. no. 103541.
[39] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "Honey badger algorithm: New metaheuristic algorithm for solving optimization problems," *Math. Comput. Simul.*, vol. 192, pp. 84–110, Feb. 2022.
[40] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[41] L. Abualigah, M. A. Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, "Reptile search algorithm (RSA): A nature-inspired meta-heuristic optimizer," *Expert Syst. Appl.*, vol. 191, Apr. 2022, Art. no. 116158.
[42] M. Dehghani, P. Trojovský, and O. P. Malik, "Green anaconda optimization: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Biomimetics*, vol. 8, no. 1, p. 121, Mar. 2023.
[43] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems," *Eng. with Comput.*, vol. 29, no. 1, pp. 17–35, Jan. 2013.
[44] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[45] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Mach. Learn.*, vol. 3, nos. 2–3, pp. 95–99, 1988.
[46] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[47] L. N. D. Castro and J. I. Timmis, "Artificial immune systems as a novel soft computing paradigm," *Soft Comput.*, vol. 7, no. 8, pp. 526–544, Aug. 2003.
[48] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies—A comprehensive introduction," *Natural Comput.*, vol. 1, no. 1, pp. 3–52, 2002.
[49] J. R. Koza and J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.
[50] R. G. Reynolds, "An introduction to cultural algorithms," in *Proc. 3rd Annu. Conf. On Evol. Program.*, 1994, pp. 131–139.
[51] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[52] M. Dehghani, Z. Montazeri, A. Dehghani, and A. Seifi, "Spring search algorithm: A new meta-heuristic optimization algorithm inspired by Hooke's law," in *Proc. IEEE 4th Int. Conf. Knowl.-Based Eng. Innov. (KBEI)*, Dec. 2017, pp. 210–214.
[53] M. Dehghani and H. Samet, "Momentum search algorithm: A new meta-heuristic optimization algorithm inspired by momentum conservation law," *Social Netw. Appl. Sci.*, vol. 2, no. 10, pp. 1–15, Oct. 2020.
[54] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, Jun. 2009.
[55] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.*, vols. 110–111, pp. 151–166, Nov. 2012.
[56] Z. Wei, C. Huang, X. Wang, T. Han, and Y. Li, "Nuclear reaction optimization: A novel and powerful physics-based algorithm for global optimization," *IEEE Access*, vol. 7, pp. 66084–66109, 2019.
[57] J. L. J. Pereira, M. B. Francisco, C. A. Diniz, G. A. Oliver, S. S. Cunha, and G. F. Gomes, "Lichtenberg algorithm: A novel hybrid physics-based metaheuristic for global optimization," *Expert Syst. Appl.*, vol. 170, May 2021, Art. no. 114522.
[58] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, and W. Al-Atabany, "Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems," *Appl. Intell.*, vol. 51, no. 3, pp. 1531–1551, Mar. 2021.
[59] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, "Equilibrium optimizer: A novel optimization algorithm," *Knowl.-Based Syst.*, vol. 191, Mar. 2020, Art. no. 105190.
[60] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-verse optimizer: A nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 27, no. 2, pp. 495–513, Feb. 2016.
[61] E. Cuevas, D. Oliva, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "Circle detection using electro-magnetism optimization," *Inf. Sci.*, vol. 182, no. 1, pp. 40–55, Jan. 2012.
[62] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303–315, Mar. 2011.
[63] S. H. S. Moosavi and V. K. Bardsiri, "Poor and rich optimization algorithm: A new human-based and multi populations algorithm," *Eng. Appl. Artif. Intell.*, vol. 86, pp. 165–181, Nov. 2019.
[64] M. Dehghani, M. Mardaneh, J. M. Guerrero, O. P. Malik, R. A. Ramirez-Mendoza, J. Matas, J. C. Vasquez, and L. Parra-Arroyo, "A new 'doctor and patient' optimization algorithm: An application to energy commitment problem," *Appl. Sci.*, vol. 10, no. 17, p. 5791, Aug. 2020.
[65] M. Dehghani and P. Trojovský, "Teamwork optimization algorithm: A new optimization approach for function minimization/maximization," *Sensors*, vol. 21, no. 13, p. 4567, Jul. 2021.
[66] M. Braik, M. H. Ryalat, and H. Al-Zoubi, "A novel meta-heuristic algorithm for solving numerical optimization problems: Ali baba and the forty thieves," *Neural Comput. Appl.*, vol. 34, no. 1, pp. 409–455, Jan. 2022.
[67] H. Givi and M. Hubalovska, "Skill optimization algorithm: A new human-based metaheuristic technique," *Comput., Mater. Continua*, vol. 74, no. 1, pp. 179–202, 2023.
[68] P. Trojovský, M. Dehghani, E. Trojovská, and E. Milkova, "Language education optimization: A new human-based metaheuristic algorithm for solving optimization problems," *Comput. Model. Eng. Sci.*, vol. 136, no. 2, pp. 1527–1573, 2023.
[69] M. A. Al-Betar, Z. A. A. Alyasseri, M. A. Awadallah, and I. A. Doush, "Coronavirus herd immunity optimizer (CHIO)," *Neural Comput. Appl.*, vol. 33, no. 10, pp. 5011–5042, May 2021.
[70] T. S. L. V. Ayyarao, N. S. S. Ramakrishna, R. M. Elavarasan, N. Polumahanthi, M. Rambabu, G. Saini, B. Khan, and B. Alatas, "War strategy optimization algorithm: A new effective metaheuristic algorithm for global optimization," *IEEE Access*, vol. 10, pp. 25073–25105, 2022.
[71] M. Dehghani, E. Trojovská, and P. Trojovský, "A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process," *Sci. Rep.*, vol. 12, no. 1, p. 9924, Jun. 2022.
[72] M. Dehghani, M. Mardaneh, J. Guerrero, O. Malik, and V. Kumar, "Football game based optimization: An application to solve energy commitment problem," *Int. J. Intell. Eng. Syst.*, vol. 13, no. 5, pp. 514–523, Oct. 2020.
[73] R. Moghdani and K. Salimifard, "Volleyball premier league algorithm," *Appl. Soft Comput.*, vol. 64, pp. 161–185, Mar. 2018.
[74] F. A. Zeidabadi and M. Dehghani, "POA: Puzzle optimization algorithm," *Int. J. Intell. Eng. Syst.*, vol. 15, no. 1, pp. 273–281, 2022.
[75] M. Dehghani, Z. Montazeri, H. Givi, J. Guerrero, and G. Dhiman, "Darts game optimizer: A new optimization technique based on darts game," *Int. J. Intell. Eng. Syst.*, vol. 13, no. 5, pp. 286–294, Oct. 2020.
[76] A. Kaveh and A. Zolghadr, "A novel meta-heuristic algorithm: Tug of war optimization," *Iran Univ. Sci. Technol.*, vol. 6, no. 4, pp. 469–492, 2016.
[77] H. Givi and M. Hubáovská, "Billiards optimization algorithm: A new game-based metaheuristic approach," *Comput., Mater. Continua*, vol. 74, no. 3, pp. 5283–5300, 2023.
[78] M. Dehghani, Z. Montazeri, and O. P. Malik, "DGO: Dice game optimizer," *Gazi Univ. J. Sci.*, vol. 32, no. 3, pp. 871–882, Sep. 2019.
[79] S. Doumari, H. Givi, M. Dehghani, and O. Malik, "Ring toss game-based optimization algorithm for solving various optimization problems," *Int. J. Intell. Eng. Syst.*, vol. 14, no. 3, pp. 545–554, Jun. 2021.
[80] M. Dehghani, Z. Montazeri, O. P. Malik, A. Ehsanifar, and A. Dehghani, "OSA: Orientation search algorithm," *Int. J. Ind. Electron., Control Optim.*, vol. 2, no. 2, pp. 99–112, 2019.
[81] F. A. Zeidabadi, M. Dehghani, P. Trojovský, Š. Hubálovský, V. Leiva, and G. Dhiman, "Archery algorithm: A novel stochastic optimization algorithm for solving optimization problems," *Comput., Mater. Continua*, vol. 72, no. 1, pp. 399–416, 2022.
[82] A. R. Glatston, *Red Panda: Biology and Conservation of the First Panda*, 2nd ed. London, U.K.: Academic, 2021, pp. 81–93.
[83] F. Wei, Z. Feng, Z. Wang, and J. Hu, "Habitat use and separation between the giant panda and the red panda," *J. Mammal.*, vol. 81, no. 2, pp. 448–455, May 2000.
[84] J. Nijboer and E. S. Dierenfeld, "Red panda nutrition: How to feed a vegetarian carnivore," in *Red Panda*. Amsterdam, The Netherlands: Elsevier, 2022, pp. 225–238.
[85] M. S. Roberts and J. L. Gittleman, "Ailurus fulgens," *Mammalian Species*, no. 222, pp. 1–8, Nov. 1984. [Online]. Available: https://academic.oup.com/mspecies/article/doi/10.2307/3503840/2600576, doi: 10.2307/3503840.

[86] N. Awad, M. Ali, J. Liang, B. Qu, P. Suganthan, and P. Definitions, ''Evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization,'' Nanyang Technol. Univ., Singapore, Tech. Rep., Nov. 2016.
[87] F. Wilcoxon, ''Individual comparisons by ranking methods,'' in *Breakthroughs in Statistics*. New York, NY, USA: Springer, 1992, pp. 196–202. [Online]. Available: https://link.springer.com/chapter/10.1007/978-1-4612-4380-9_16
[88] B. K. Kannan and S. N. Kramer, ''An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design,'' *J. Mech. Des.*, vol. 116, no. 2, pp. 405–411, Jun. 1994.
[89] A. H. Gandomi and X.-S. Yang, ''Benchmark problems in structural optimization,'' in *Computational Optimization, Methods and Algorithms*. Berlin, Germany: Springer, 2011, pp. 259–281. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-20859-1_12
[90] E. Mezura-Montes and C. A. C. Coello, ''Useful infeasible solutions in engineering optimization with evolutionary algorithms,'' in *Proc. Mexican Int. Conf. Artif. Intell.*, 2005, pp. 652–662.

HADI GIVI (Member, IEEE) was born in Isfahan, Iran, in 1988. He received the B.Sc. degree in electrical power engineering from the Isfahan University of Technology, in 2010, the M.Sc. degree in electrical power engineering from the Amirkabir University of Technology, in 2012, and the Ph.D. degree in electrical power engineering from Shiraz University, in 2018. Since 2018, he has been an Assistant Professor with the Department of Electrical Engineering, Shahreza Campus, University of Isfahan. His main research interests include condition monitoring of power electronic converters, optimization algorithms as well as their applications in the power systems, and power quality analysis.

MOHAMMAD DEHGHANI received the B.S. degree in electrical engineering from Shahid Bahonar University, Kerman, Iran, in 2012, the M.S. degree in electrical engineering from Shiraz University, Shiraz, Iran, in 2016, and the Ph.D. degree from the Shiraz University of Technology, Shiraz, in 2020. His current research interests include optimization, metaheuristic algorithms, power systems, and energy commitment.

ŠTĚPÁN HUBÁLOVSKÝ received the M.Sc. and Ph.D. degrees from the Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic, in 1995 and 1998, respectively. He became an Associate Professor with the Faculty of Informatics and Management, University of Hradec Králové, Czech Republic, in 2012. He is currently the Vice-Dean of the Faculty of Science, University of Hradec Králové. His research interests include technical cybernetics, computer simulation and optimization, and big data processing.