Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 9107547, 19 pages
https://doi.org/10.1155/2021/9107547

Research Article

A Bio-Inspired Method for Engineering Design Optimization Inspired by Dingoes Hunting Strategies

Hernán Peraza-Vázquez,[1] **Adrián F. Peña-Delgado**,[2] **Gustavo Echavarría-Castillo**,[1] **Ana Beatriz Morales-Cepeda**,[3] **Jonás Velasco-Álvarez**,[4] **and Fernando Ruiz-Perez**[1]

[1]*Instituto Politécnico Nacional-CICATA Altamira, km.14.5 carretera Tampico-Puerto Industrial Altamira, Altamira 89600, Tamaulipas, Mexico*
[2]*Universidad Tecnológica de Altamira, Boulevard de los Ríos Km. 3 + 100, Puerto Industrial Altamira, Altamira 89601, Tamaulipas, Mexico*
[3]*TecNM/Instituto Tecnológico de Ciudad Madero, Juventino Rosas y Jesús Urueta s/n Col. Los Mangos, Madero 89318, Tamaulipas, Mexico*
[4]*Centro de Investigación en Matemáticas (CIMAT).A.C,CONACyT, Bartolomé de las Casas 314, Guanajuato 20259, Mexico*

Correspondence should be addressed to Hernán Peraza-Vázquez; hperaza@ipn.mx and Adrián F. Peña-Delgado; apea@utaltamira.edu.mx

Received 20 May 2021; Revised 15 July 2021; Accepted 23 August 2021; Published 21 September 2021

Academic Editor: José Francisco Gómez Aguilar

A novel bio-inspired algorithm, namely, Dingo Optimization Algorithm (DOA), is proposed for solving optimization problems. The DOA mimics the social behavior of the Australian dingo dog. The algorithm is inspired by the hunting strategies of dingoes which are attacking by persecution, grouping tactics, and scavenging behavior. In order to increment the overall efficiency and performance of this method, three search strategies associated with four rules were formulated in the DOA. These strategies and rules provide a fine balance between intensification (exploitation) and diversification (exploration) over the search space. The proposed method is verified using several benchmark problems commonly used in the optimization field, classical design engineering problems, and optimal tuning of a Proportional-Integral-Derivative (PID) controller are also presented. Furthermore, the DOA's performance is tested against five popular evolutionary algorithms. The results have shown that the DOA is highly competitive with other metaheuristics, beating them at the majority of the test functions.

1. Introduction

Typically, a constrained optimization problem can be described as a nonlinear programming problem (NLP) [1], as shown below.

$$\begin{aligned} &\text{Minimize} && f(x), \\ &\text{Subject to} && g_i(x) \le 0, i = 1, \ldots, p, \\ &&& h_j(x) = 0, j = 1, \ldots, q, \\ &&& x_k^{(l)} \le x_k \le x_k^{(u)}, k = 1, \ldots, D. \end{aligned} \tag{1}$$

In the above NLP problem, the function f is the objective function, where $f(x)$: $R^D \longrightarrow R$, there are D variables, $x = (x_1, \ldots, x_D)$, is a vector of size D, $x \in R^D$, R^D representing the whole search space, g_i are the inequality constraints, h_j are the equality constraints, and $x_k^{(l)}$, $x_k^{(u)}$ are the lower bound constraints and upper bound constraints, respectively, where p and q are defined as the number of inequality and equality constraints, respectively. Thus, the optimization goal is to find a feasible vector x to minimize the objective function. When the vector x contains a subset of μ and ν vectors of continuous real and integer variables,

respectively, $|\mu| + |\nu| = D$, then the NLP problem becomes a mixed−integer nonlinear programming problem (MINLP). Nonconvex NLPs and MINLPs are commonly found in real-world situations. Therefore, the scientific community continues developing new approaches to obtain optimal solutions with acceptable computation time in various engineering, industrial, and science fields. For example, on design optimization, the design objective could simply be to minimize the cost or maximize the efficiency of production. However, the objective could be more complex, e.g., controlling the highly nonlinear behavior of the pH neutralization process in a chemical plant. The need to solve practical NLP/MINLP problems has led to the development of a large number of heuristics and metaheuristics over the last two decades [2, 3]. Metaheuristics, which are emerging as effective alternatives for solving nondeterministic Polynomial Time hard (NP-hard) optimization problems, are strategies for designing or improving very general heuristics procedures with high performance in order to find (near) optimal solutions. The goal of the metaheuristics is efficient exploration (diversification) and exploitation (intensification) of the search space, where an effective algorithm sets a good ratio between this two parameters. For example, we can take advantage of the search experience to guide search engines by applying learning strategies or incorporating probabilistic decisions. Metaheuristics can be classified in two main groups, local search and population based. In the first group, the search process starts with one candidate solution and then it is improved in each iteration over the runtime. Algorithms such as variable neighborhood search (VNS) [4], Tabu search (TS) [5], Simulated annealing (SA) [6], and Iterated local search [7] are considered as part of the local search metaheuristics group. On the other hand, in the second group, among the population-based metaheuristics, evolutionary algorithms are metaheuristics inspired by the process of natural selection. Genetic Algorithms (GA) [8], Genetic Programming [9], Differential Evolution (DE) [10], and Evolution Strategy (ES) [11] are considered as state-of-the-art population-based evolutionary algorithms. Moreover, Ant Colony Optimization (ACO) [12], Cuckoo Search Algorithm (CSA) [13], and Particle Swarm Algorithm (PSO) [14] are some representative population-based metaheuristics categorized as swarm based [15]. Even though there is a great amount of research on metaheuristics, it continues to be used in many fields, e.g., cluster analysis, scheduling, artificial intelligence, process engineering, etc., with flattering results. However, there is no particular heuristic algorithm suitable for all optimization problems [16]. Therefore, designing new optimization techniques is an active research field within the scientific community [17]. A survey of some of the most relevant animal or natured based bio-inspired algorithms includes but is not limited to Virus Colony Search (VCS) [18], Plant Propagation algorithms [19], Lightning Search Algorithm (LSA) [20], Ant Lion Optimizer (ALO) [21], Lion Optimizer Algorithm (LOA) [22], Spotted Hyena Optimizer (SHO) [23], Harris Hawks Optimization (HHO) [24], Dragonfly Algorithm (DFA) [25], Grey Wolf Optimizer (GWO) [26], Dolphin Echolocation Algorithm [27], Water Strider Algorithm (WSA), [27], Slime Mould Algorithm (SMA) [28], Moth Search Algorithm (MSA) [29], Colony Predation Algorithm (CPA) [30], Black Widow Optimization Algorithm (BWOA) [31], Grasshopper Optimization Algorithm (Goa) [32], and the Hunger games search (HGS) [33]. Additionally, some outstanding physical phenomena based bio-inspired algorithm for optimization are [27]: Magnetic Charged System Search (MCSS), Colliding Bodies Optimization (CBO), Water Evaporation Optimization (WEO), Vibrating Particles System (VPS), Thermal Exchange Optimization (TEO), Cyclical Parthenogenesis Algorithm (CPA), among others.

Here, a novel bio-inspired algorithm, namely Dingo Optimization Algorithm (DOA), is proposed for solving optimization tasks. It is based on the simulation of the hunting strategies of Dingoes, which are attacking by chasing, grouping tactics, and scavenging behavior. The remainder of this paper is organized as follows. Section 2 illustrates the DOA details, including the inspiration and mathematical model. In order to illustrate the proficiency and robustness of the proposed approach, several numerical examples and their comparison with state-of-the-art metaheuristics are presented in Section 3. Finally, Section 4 summarizes our findings and concludes the paper with a brief discussion on the scope for future work.

2. Dingo Optimization Algorithm (DOA)

In this section, the inspiration of the proposed method is first discussed. Then, the mathematical model is provided.

2.1. Biological Fundamentals. The dingo is Australia's native largest mammalian carnivore, and their scientific name is **Canis lupus dingo**. Several studies have been conducted to study the dingoes' feeding behavior and diet, showing that these canines prey on several species such as mammals, birds, vegetation (seeds), reptiles, insects, fish, crabs, and frogs, just to mention some [34]. They are opportunistic hunters but will also scavenge food when they are exploring new territories and suddenly find dead prey. Their hunting behavior can be variable. Usually, they pursue and attack their prey from behind. Group attack is their most used hunting strategy in which they surround the prey inside a perimeter and begin to chase it until they fatigue it. Further details about the dingoes' behavior can be found in [35, 36].

2.2. Mathematical Model and Optimization Algorithm. In this section, the mathematical model of Dingoes different hunting strategies is first provided. The DOA algorithm is then proposed. The hunting strategies considered are attacking by persecution, grouping tactics, and scavenging behavior. In addition, dingoes' survival probability is also considered.

2.3. Strategy 1: Group Attack. Predators often use highly intelligent hunting techniques. Dingoes usually hunt small prey, such as rabbits, individually, but when hunting large prey such as kangaroos, they gather in groups. Dingoes can

find the location of the prey and surround it, such as wolves, see Figure 1. This behavior is represented by the following equation:

$$\vec{x}_i(t+1) = \beta_1 \sum_{k=1}^{na} \frac{\left[\overrightarrow{\varphi_k(t)} - \vec{x}_i(t)\right]}{na} - \vec{x}_*(t), \quad (2)$$

where $\vec{x}(t+1)$ is the new position of a search agent (indicates dingoes' movement), *na* is a random integer number generated in the invertal of [2, SizePop/2], where SizePop is the total size of the population of dingoes. $\vec{\varphi}_k(t)$, is a subset of search agents (dingoes that will attack) where $\varphi \subset X$, X is the dingoes' population randomly generated, $\vec{x}_i(t)$ is the current search agent, $\vec{x}_*(t)$ is the best search agent found from the previous iteration, and β_1 is a random number uniformly generated in the interval of $[-2, 2]$; it is a scale factor that changes the magnitude and sense of the dingoes' trajectories. The Group attack pseudocode is shown in Algorithm 1.

2.4. Strategy 2: Persecution. Dingoes usually hunt small prey, which is chased until it is caught individually. The following equation models this behavior:

$$\vec{x}_i(t+1) = \vec{x}_*(t) + \beta_1 * e^{\beta_2} * \left(\vec{x}_{r_1}(t) - \vec{x}_i(t)\right), \quad (3)$$

where $\vec{x}(t+1)$ indicates the dingoes' movement, $\vec{x}_i(t)$ is the current search agent, $\vec{x}_*(t)$ is the best search agent found from the previous iteration, β_1 has the same value as in equation (2), β_2 is a random number uniformly generated in the interval of $[-1, 1]$, r_1 is the random number generated in the interval from 1 to the size of a maximum of search agents (dingoes), and $\vec{x}_{r1}(t)$ is the r_1-th search agent selected, where $i \neq r_1$.

Equation (3) is used to represent the dingoes' trajectories while hunting for their prey. At the same time, Figure 2 is used to graphically illustrate its parameters.

2.5. Strategy 3: Scavenger. Scavenger behavior is defined as the action when dingoes find carrion to eat when they are randomly walking in their habitat. Equation (4) is used to model this behavior, which is graphically displayed in Figure 3

$$\vec{x}_i(t+1) = \frac{1}{2}\left[e^{\beta_2} * \vec{x}_{r_1}(t) - (-1)^{\sigma} * \vec{x}_i(t)\right], \quad (4)$$

where $\vec{x}(t+1)$ indicates the dingoes' movement, β_2 has the same value as in equation (3), r_1 is the random number generated in the interval from 1 to the size of a maximum of search agents (dingoes), $\vec{x}_{r_1}(t)$ is the r_1-th search agent selected, $\vec{x}_i(t)$ is the current search agent, where $i \neq r_1$ and σ is a binary number randomly generated by Algorithm 2, $\sigma \in \{0, 1\}$.

2.6. Strategy 4: Dingoes' Survival Rates. The Australian dingo dog is at risk of extinction mainly due to illegal hunting. In the DOA, the dingoes' survival rate value is provided by the following equation:

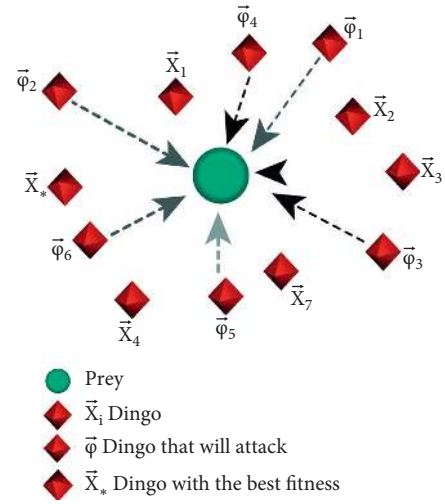

Figure 1: Group attack strategy.

$$\text{survival}(i) = \frac{\text{fitness}_{\max} - \text{fitness}(i)}{\text{fitness}_{\max} - \text{fitness}_{\min}}, \quad (5)$$

where $\text{fitness}_{\max}$ and $\text{fitness}_{\min}$ are the worst and the best fitness value in the current generation, respectively, whereas $\text{fitness}(i)$ is the current fitness value of the $i-th$ search agent. The survival vector in equation (5) contains the normalized fitness in the interval of $[0, 1]$. Equation (6) is applied for low survival rates by Algorithm 3, e.g., for survival rates values equal to or less than 0.3.

$$\vec{x}_i(t) = \vec{x}_*(t) + \frac{1}{2}\left[\vec{x}_{r_1}(t) - (-1)^{\sigma} * \vec{x}_{r_2}(t)\right], \quad (6)$$

where $\vec{x}_i(t)$ is the search agent with low survival rates that will be updated, r_1 and r_2 are random numbers generated in the interval from 1 to the size maximum of search agents (dingoes), with $r_1 \neq r_2$, $\vec{x}_{r_1}(t)$ and $\vec{x}_{r_2}(t)$, are the r_1, r_2-th search agents selected, $\vec{x}_*(t)$ is the best search agent found from the previous iteration and σ is a binary number randomly generated by the second algorithm, $\sigma \in \{0, 1\}$. Note that equation (6) is an addition or subtraction of vectors, defined by the random value of σ.

2.7. Pseudocode for DOA. The pseudocode of the DOA is explained in Algorithm 4, whereas the overall flow is shown in Figure 4.

3. DOA Algorithm Analysis

In this section, an in-depth analysis of the DOA algorithm is carried out. This analysis includes the DOA's time complexity, parameters settings studies, the hunting strategies analysis, and the associated effects of the population size in the algorithm's performance.

3.1. Time Complexity. Without any loss of generality, let **f** be any optimization problem and suppose that **O(f)** is the computational time complexity of evaluating its function value. Thereby, the DOA computational time complexity is

(1) Begin procedure
(2) Generate a random integer, named **na**, **na** $\in$ [2, SizePop/2], answer the question: How many dingoes will attack?
(3) Generate k-index vector of random integers, with size **na**, k-index $\in\in$ [1, SizePop], answer the question: Which dingoes will attack?
(4) Generate the φ subset from Population using the k-index vector of positions, $\varphi \subset$ Population, φ are the dingoes that will attack
(5) Apply equation (7).
(6) Return the new positions of dingoes
(7) End procedure

Algorithm 1: Group attack procedure.

(1) Begin procedure
(2) **if** rand ≤ 0.5 **then**
(3) return 0
(4) **else**
(5) return 1
(6) **end if**
(7) End procedure

Algorithm 2: σ procedure.

(1) Begin procedure
(2) **for** $i = 1$ to sizePopulation **do**
(3) **if** survival $(i) \leq 0.3$ **then**
(4) Strategy 4: $\vec{x}_i$ search agent updated by equation (6).
(5) **end if**
(6) **end for**
(7) End procedure

Algorithm 3: Survival procedure.

(1) **procedure** DOA
(2) Initialization of parameters
(3) $P = 0.5$, probability of hunting or scavenger strategy
(4) $Q = 0.7$, probability of Strategy 1 (group attack) or Strategy 2 (persecution attack)
(5) Generate the initial population
(6) **while** iteration < Max Number of Iterations **do**
(7) **if** random $< P$ **then**
(8) **if** random $< Q$ **then**
(9) Strategy 1: Group Attack Procedure, Algorithm 1, equation (2).
(10) **else**
(11) : Strategy 2: Persecution, (3).
(12) **end if**
(13) **else**
(14) Strategy 3: Scavenger, (4).
(15) **end if**
(16) Update search agents that have low survival value, Algorithm 3, equation (6)
(17) Calculate x_{new}, the fitness value of the new search agents
(18) **if** $x_{\text{new}} < x_*$ **then**
(19) $x_* = x_{\text{new}}$
(20) **end if**
(21) iteration = iteration + 1
(22) **end while**
(23) Display x_*, the best optimal solution
(24) **end procedure**

Algorithm 4: Dingo Optimizer Algorithm.

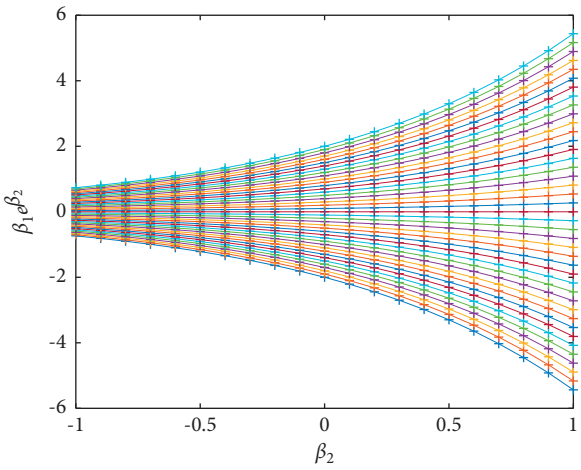

Figure 2: Effects of β_1 and β_2 on the evaluation of equation (3) (dingoes' hunting trajectories).

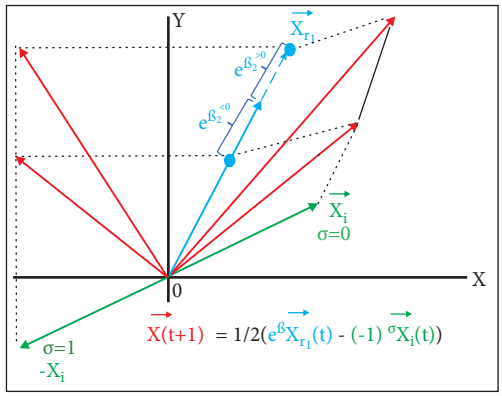

Figure 3: Scavenger vectorial trajectory, taken from the evaluation of equation (4). The red line represents the possible trajectory of the resulting vector when randomly varying β ($\beta > 0$ or $\beta < 0$) and σ ($\sigma = 1$ or $\sigma = 0$), blue and green lines, respectively.

defined as **O(tMx ∗ nDg ∗ f)**, where *tMx* is the maximum number of iterations and *nDg* is the number of dingoes (population size).

3.2. Hunter Strategies Analysis. To better understand the performance of each of the hunting strategies separately, a unimodal and a multimodal problem was conducted. The algorithm was modified to use only one strategy on each run time and executed with the number of iterations set as 500 and population size (search agents) of 100. The output of this analysis is displayed in Figure 5. Note that the group attack strategy has obtained the best performance for the unimodal function F2, while the persecution strategy has the worst, while, for the multimodal function F14, the persecution strategy overcomes the scavenger strategy and shows competitive results compared to the attack strategy.

3.3. Population Size Analysis. The effects of the population size on the performance of the DOA algorithm are studied by fixing the number of iterations to 100 and then varying the population size initially at 15, then 30, 40, 50, 100, and 200 for the F2 F14 functions. The output of these tests is summarized in Table 1 and Figure 6, where the best optimal value is found at a population size of 100.

Notice that from the population size analysis results described in Table 1, row two corresponding to size 30 shows that the DOA algorithm outperformed, in F2 and F14, the algorithms reported in Table 2. Nevertheless, Table 1 data were conducted with a fixed population size of 100, whereas Table 2 data were calculated with a population size of 500. Moreover, for a smaller population size of 15, it still outperforms the other algorithms.

3.4. P and Q Parameters Analysis. The DOA algorithm uses two parameters, P and Q. P is a fixed value that indicates the probability of the algorithm to choose between the hunting or scavenger strategy. If the hunting strategy is selected by the algorithm, then a fixed Q value indicates its probability to choose between group attack or persecution strategy.

In order to determine the effects of *P* and *Q* parameters on the DOA performance, an analysis of said variables is carried out by means of the benchmark problems F1 to F23; see Tables 3–5. The methodology consists of setting *P* fixed at 0.5 while *Q* starts on 0.25 and is incremented on 0.25 steps during four runtimes, one for each *Q* value until 1 is achieved. Afterward, a similar approach is conducted, leaving *Q* fixed at 0.5 while *P* starts on 0.25 and vary in 0.25 increments until *P* is equal to 1. The convergence analysis results of this parameters test are shown in Figures 7 and 8. It is to be noticed that regardless of P and Q values, the algorithm converges to the solution reported in Table 2. This is due to the incorporation of the **survival strategy**, which improves the quality of search agents by updating those with low survival values.

Based on this, for the rest of the paper, the DOA tests will be conducted with P and *Q* fixed at 0.5 and 0.70, respectively.

4. Experimental Setup

In this section, 23 classical benchmark functions, reported in the literature [37], are optimized to investigate the effectiveness, efficiency, and stability of the DOA algorithm. The functions are categorized as unimodal, multimodal, and fixed-dimension multimodal. Table 3 shows unimodal functions, labeled from F1 to F7, whereas functions F8 to F13 are considered as multimodal, as shown in Table 4. Additionally, functions F14 to F23 are defined as fixed-dimension multimodal in Table 4. Unimodal functions allow testing the exploitation ability since they only have one global optimum, whereas multimodal functions and fixed-dimension multimodal functions are able to test the exploration ability since they include many local optima. Tables 3–5 summarizes these benchmark functions where *Di m* indicates the dimension of the function, Interval is the boundary of the function's search space, and f_{min} is the optimum value. Figure 9 shows the typical 2D plots of the cost function for some test cases considered in this study.

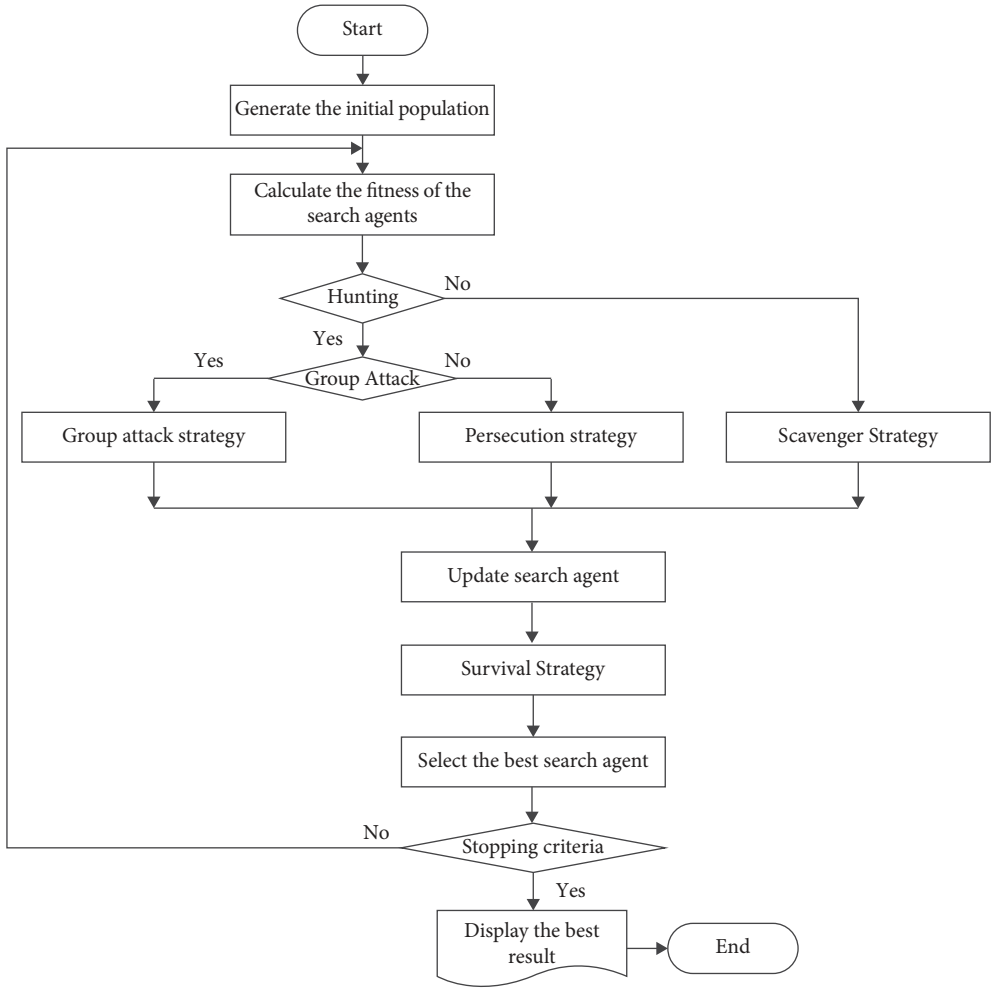

Figure 4: DOA Flowchart.

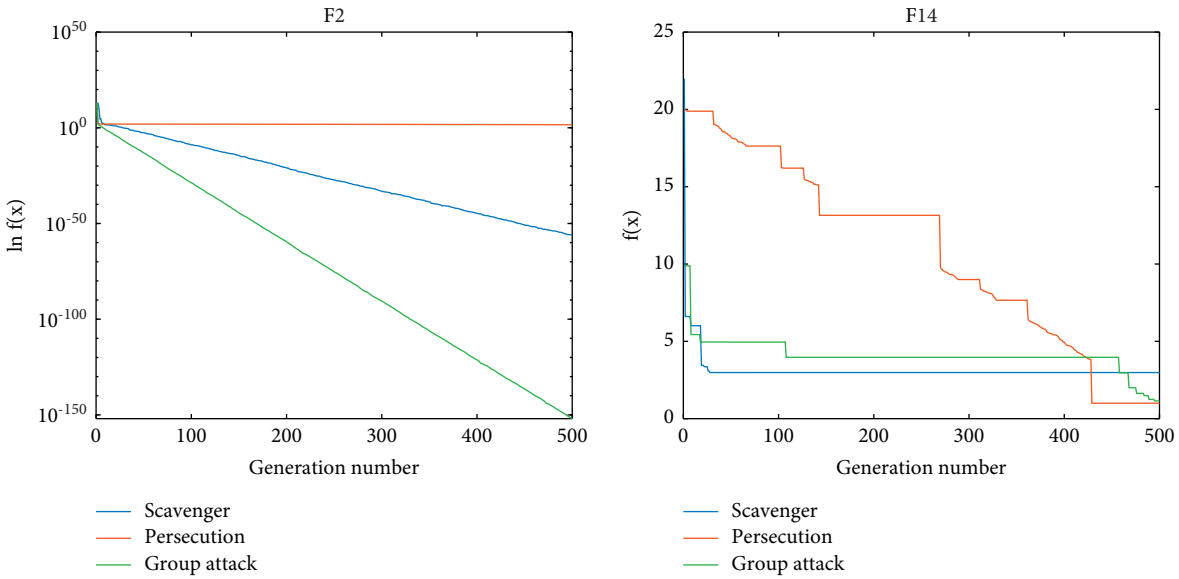

Figure 5: Hunter strategies analysis.

Table 1: Population size analysis.

Population size	F2 (optimal value: 0)				F14 (optimal value: 1)			
	Worst	Best	Ave	std	Worst	Best	Ave	std
15	$1.64E-17$	$1.62E-48$	$1.64E-18$	$5.18E-18$	$1.17E+00$	$9.96E-01$	$1.02E+00$	$5.52E-02$
30	$6.53E-20$	$2.85E-43$	$6.86E-21$	$2.06E-20$	$9.98E-01$	$9.98E-01$	$9.98E-01$	$1.49E-04$
50	$6.57E-27$	$5.87E-47$	$6.64E-28$	$2.08E-27$	$9.99E-01$	$9.98E-01$	$9.98E-01$	$3.25E-04$
100	$1.09E-29$	$3.96E-58$	$1.09E-30$	$3.43E-30$	$9.99E-01$	$9.98E-01$	$9.98E-01$	$3.57E-04$
200	$4.34E-39$	$3.96E-53$	$4.84E-40$	$1.36E-39$	$9.99E-01$	$9.98E-01$	$9.98E-01$	$3.37E-04$

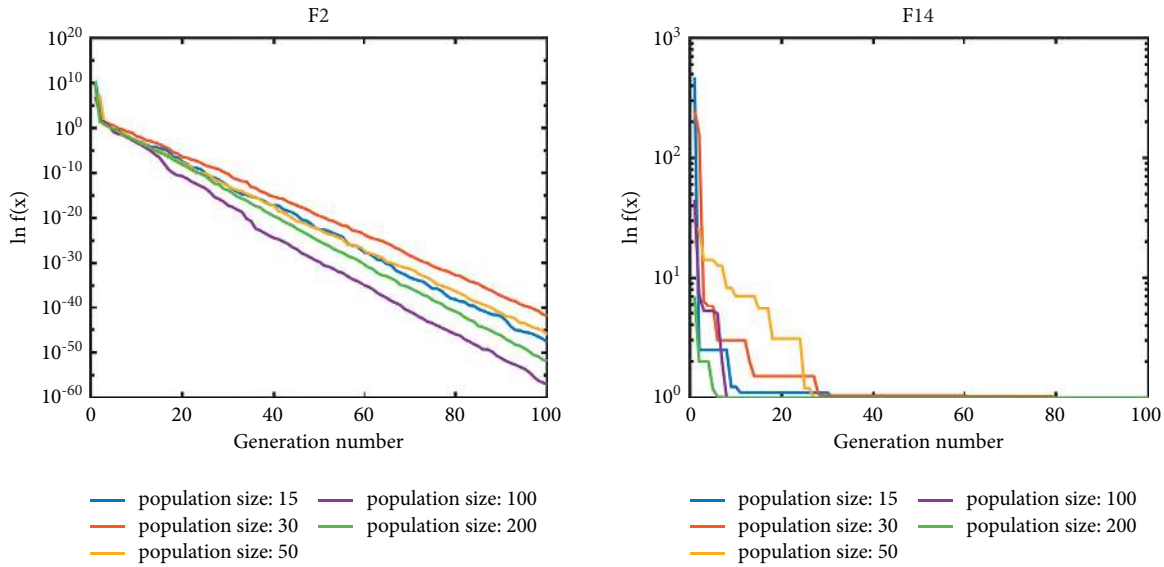

Figure 6: Population size analysis.

For each benchmark function, the DOA algorithm was run 30 times, the size of the population (search agents) was set to 30, while the number of iterations was defined as 500. Figure 10 shows the convergence graphs of all functions.

The DOA algorithm was compared with the following algorithms: the Whale Optimization Algorithm (WOA) [38], Particle Swarm Optimization (PSO) [39], Gravity Search Algorithm (GSA) [40], Differential Evolution (DE) [41], and Fast Evolutionary Programming (FEP) [42]. Our approach is implemented in MATLAB R2018a. All computations were carried out on a standard PC (Linux Kubuntu 18.04 LTS, Intel core i7, 2.50 GHz, 16 GB). The six algorithms were ranked by computing their Mean Absolute Error (MAE). MAE is a valid statistical criterion and an unambiguous measurement of the average error magnitude. It shows how far the results are from actual values. The MAE formula is as follows:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} |m_i - k_i|}{N}, \tag{7}$$

where m_i indicates the mean of the optimal values, k_i is the corresponding global optimal value, and N represents the number of test functions. Table 6 shows the average error rates obtained in the 23 benchmark functions. The ranking of all the algorithms based on their MAE calculations is illustrated in Table 7.

5. Results and Discussion

According to the statistical results given in Table 2, the Dingo Optimization Algorithm (DOA) is able to provide very competitive results. In the exploitation analysis, unimodal functions, the DOA outperforms all other algorithms, as Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO), Gravity Search Algorithm (GSA), and Fast Evolutionary Programming (FEP), in $F1$, $F2$, $F3$, and $F7$ functions and similar to Differential Evolution (DE), it found the optimal result in F4. Table 8 shows the exploitation capability results summary. We can see that this result represents an accumulated rate of 71.43 % that outperforms or ties over the others algorithms. Therefore, these results show DOA superior performance in terms of exploitation at the optimum. On the other hand, the exploration analysis shows that the DOA was most efficient in $F10$, $F15$, $F16$, $F17$, $F19$, $F20$, $F21$, and $F23$, multimodal and fixed-dimension multimodal functions (see Table 2). In addition, the DOA showed similar behavior to other metaheuristics to find the optimal result in $F9$, $F11$, $F14$, $F18$, and $F22$ functions. This result represents an accumulated rate of 81.25 % that outperforms or ties compared with other algorithms. Table 8 confirms that the DOA also has a very good exploration capability. We can see that the DOA algorithm is at least the second-best and frequently the most efficient on the majority of the test functions due to the

Table 2: Performance and comparison of DOA.

F	Value	DOA		WOA		PSO		GSA		DE		FEP	
		ave	std	ave	std	ave	std	Ave	std	ave	std	ave	std
F1	0	**0**	0	1.41*E* − 30	4.91*E* − 30	0.000136	0.000202	2.53*E* − 16	9.67*E* − 17	8.2*E* − 14	5.9*E* − 14	0.00057	0.00013
F2	0	**0**	0	1.06*E* − 21	2.39*E* − 21	0.042144	0.045421	0.055655	0.194074	1.5*E* − 09	9.9*E* − 10	0.0081	0.00077
F3	0	**0**	0	5.39*E* − 07	2.93*E* − 06	70.12562	22.11924	896.5347	318.9559	6.8*E* − 11	7.4*E* − 11	0.016	0.014
F4	0	**0**	0	0.072581	0.39747	1.086481	0.317039	7.35487	1.741452	**0**	0	0.3	0.5
F5	0	28.9	0.043075118	27.86558	0.763626	96.71832	60.11559	67.54309	62.22534	**0**	0	5.06	5.87
F6	0	5.0163	0.368832716	3.116266	0.532429	0.000102	8.28*E* − 05	2.5*E* − 16	1.74*E* − 16	**0**	0	**0**	0
F7	0	**1.20*E* − 05**	0.0029289	0.001425	0.001149	0.122854	0.044957	0.089441	0.04339	0.00463	0.0012	0.1415	0.3522
F8	−2094.9	−3410.152	1087.135358	−5080.76	695.7968	−4841.29	1152.814	**−2821.07**	493.0375	−11080.1	574.7	−12554.5	52.6
F9	0	**0**	0	**0**	0	46.70423	11.62938	25.96841	7.470068	69.2	38.8	0.046	0.012
F10	0	**8.8818*E* − 16**	2.9064*E* − 14	7.4043	9.897572	0.276015	0.50901	0.062087	0.23628	9.7*E* − 08	4.2*E* − 08	0.018	0.0021
F11	0	**0**	0	0.000289	0.001586	0.009215	0.007724	27.70154	5.040343	**0**	0	0.016	0.022
F12	0	0.30312	4.2*E* − 08	0.339676	0.214864	0.006917	0.026301	1.799617	0.95114	**7.9*E* − 15**	8*E* − 15	9.2*E* − 06	3.6*E* − 06
F13	0	2.9914	0.008907	1.889015	0.266088	0.006675	0.008907	8.899084	7.126241	**5.1*E* − 14**	4.8*E* − 14	0.00016	0.000073
F14	1	**0.998**	0	2.111973	2.498594	3.627168	2.560828	5.859838	3.831299	**0.998**	3.3*E* − 16	1.22	0.56
F15	0.0003	**0.00030749**	0.000036795	0.000572	0.000324	0.000577	0.000222	0.003673	0.001647	4.5*E* − 14	0.00033	0.0005	0.00032
F16	−1.0316	**−1.0316**	0	−1.03163	4.2*E* − 07	−1.03163	6.25*E* − 16	−1.03163	4.88*E* − 16	−1.03163	3.1*E* − 13	−1.03	4.9*E* − 07
F17	0.398	**0.39789**	0	0.397914	2.7*E* − 05	0.397887	0	0.397887	0	0.397887	9.9*E* − 09	0.398	1.5*E* − 07
F18	3	**3**	0	**3**	4.22*E* − 15	**3**	1.33*E* − 15	**3**	4.17*E* − 15	**3**	2*E* − 15	3.02	0.11
F19	−3.86	**−3.8628**	0	−3.85616	0.002706	−3.86278	2.58*E* − 15	−3.86278	2.29*E* − 15	N/A	N/A	−3.86	0.000014
F20	−3.32	**−3.322**	0.0000025	−2.98105	0.376653	−3.26634	0.060516	−3.31778	0.023081	N/A	N/A	−3.27	0.059
F21	−10.153	**−10.1532**	0.007724	−7.04918	3.629551	−6.8651	3.019644	−5.95512	3.737079	−10.1532	2.5*E* − 06	−5.52	1.59
F22	−10.403	**−10.4029**	5*E* − 05	−8.18178	3.829202	−8.45653	3.087094	−9.68447	2.014088	**−10.4029**	3.9*E* − 07	−5.53	2.12
F23	−10.536	**−10.5364**	0.0000025	−9.34238	2.414737	−9.95291	1.782786	**−10.5364**	2.6*E* − 15	**−10.5364**	1.9*E* − 07	−6.57	3.14

Table 3: Unimodal testbench functions [37].

Function	Dim	Interval	$f_{\min}$
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F_3(x) = \sum_{i=1} (\sum_{j-1}^{i} x_j)^2$	30	$[-100, 100]$	0
$F_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	30	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n-1} [100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	$[-100, 100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0

Table 4: Multimodal testbench functions [37].

Function	Dim	Interval	$f_{\min}$
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]$	−2094.9145
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i + 10)]$	30	$[-5.12, 5.12]$	0
$F_{10}(x) = -20 \exp(-0.2\sqrt{1/n \sum_{i=1}^{n} x_i^2}) - \exp(1/n \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e$	30	$[-32, 32]$	0
$F_{11}(x) = 1/4000 \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600, 600]$	0
$F_{12}(x) = \pi/n \{10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + (x_i + 1/4) u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0 - a < x_i < a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases}$	30	$[-50, 50]$	0
$F_{13}(x) = 0.1 \{\sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)]\}$ $+ \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

Table 5: Fixed-dimension multimodal testbench functions [37].

Function	Dim	Interval	$f_{\min}$
$F_{14}(x) = (1/500 + \sum j = 1^{25} 1/j + \sum i = 1^{2} (xi - aij)^6)^{-1}$	2	$[-65, 65]$	1
$F_{15}(x) = \sum_{i=1}^{11} [ai - x1 (bi^2 + bix2)/bi^2 + bix3 + x4]^2$	4	$[-5, 5]$	0.00030
$F_{16}(x) = 4x1^2 - 2.1x1^4 + (1/3)x1^6 + x1x2 - 4x2^2 + 42^4$	2	$[-5, 5]$	−1.0316
$F_{17}(x) = (x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6)^2 + 10(1 - (1/8\pi))\cos x_1 + 10$	2	$[-5, 5]$	0.398
$F_{18}(x) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2$ $\times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2)$	3	$[1, 3]$	−3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2)$	6	$[0, 1]$	−3.32
$F_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	−10.1532
$F_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	−10.4028
$F_{23}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	−10.5363

integrated mechanisms of exploitation and exploration leading the algorithm to the global optimum. The statistical results of the MAE test show that the DOA algorithm has the lowest value of the mean absolute error for the 23 classical benchmark functions and outperforms all the other algorithms (see Table 6), whereby DOA algorithm appears ranked in the first position (see Table 7). Additionally, a convergence analysis was carried out. The purpose of the convergence analysis is to understand and visualize the search on promising regions by the algorithm exploration and exploitation capabilities. The DOA and WOA algorithm are compared during the convergence analysis due to WOA better performance over the metaheuristics reported in [38]. Figure 11 illustrates the DOA convergence analysis results for selected test functions versus the highest-ranking algorithms taken from MAE test. We can see that the DOA

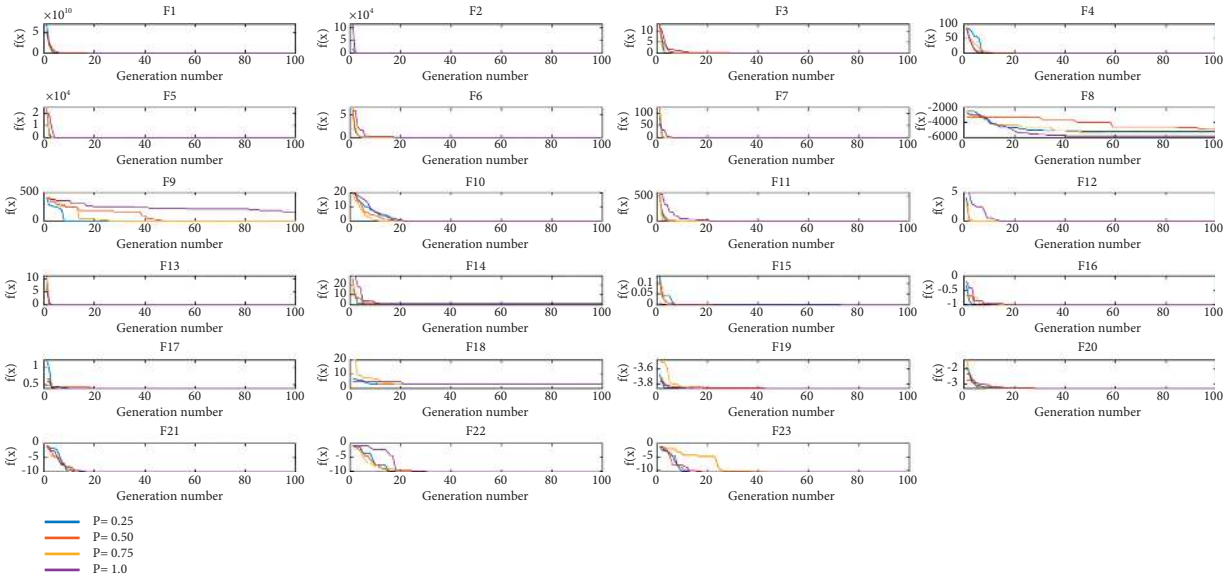

Figure 7: P and Q parameters analysis, first test: Q is set fixed at 0.5, while P starts on 0.25 and then, it increases on 0.25 until P is equal to 1.

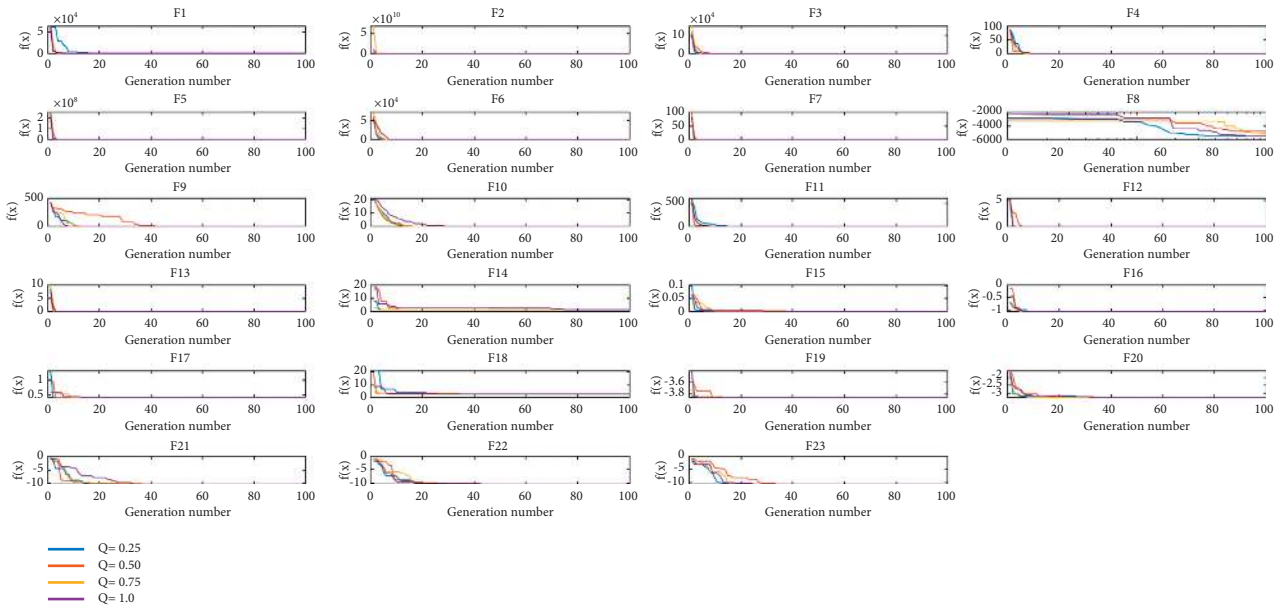

Figure 8: P and Q parameters analysis, second test: P is set fixed at 0.5, while Q starts on 0.25 and then increases on 0.25 until Q is equal to 1.

converges faster than WOA due to its adaptive mechanism. This behavior is illustrated on *F*1, *F*2, and *F*4 test benchmark functions; see Figure 11. Whereas in *F*7, *F*10, *F*14, *F*15, *F*22 and *F*23 test functions, the DOA converges rapidly from the initial stage of iteration. Based on this trends, we conclude that the DOA exploitation and exploration capabilities are quite effective finding the optimal.

6. Real-World Applications

In this section, a constrained optimization problem, typically represented by (1), is considered. The DOA algorithm was tested with four constrained engineering design problems: a cantilever beam, a three-bar truss, a pressure vessel, and a gear train design problem. The pressure vessel design

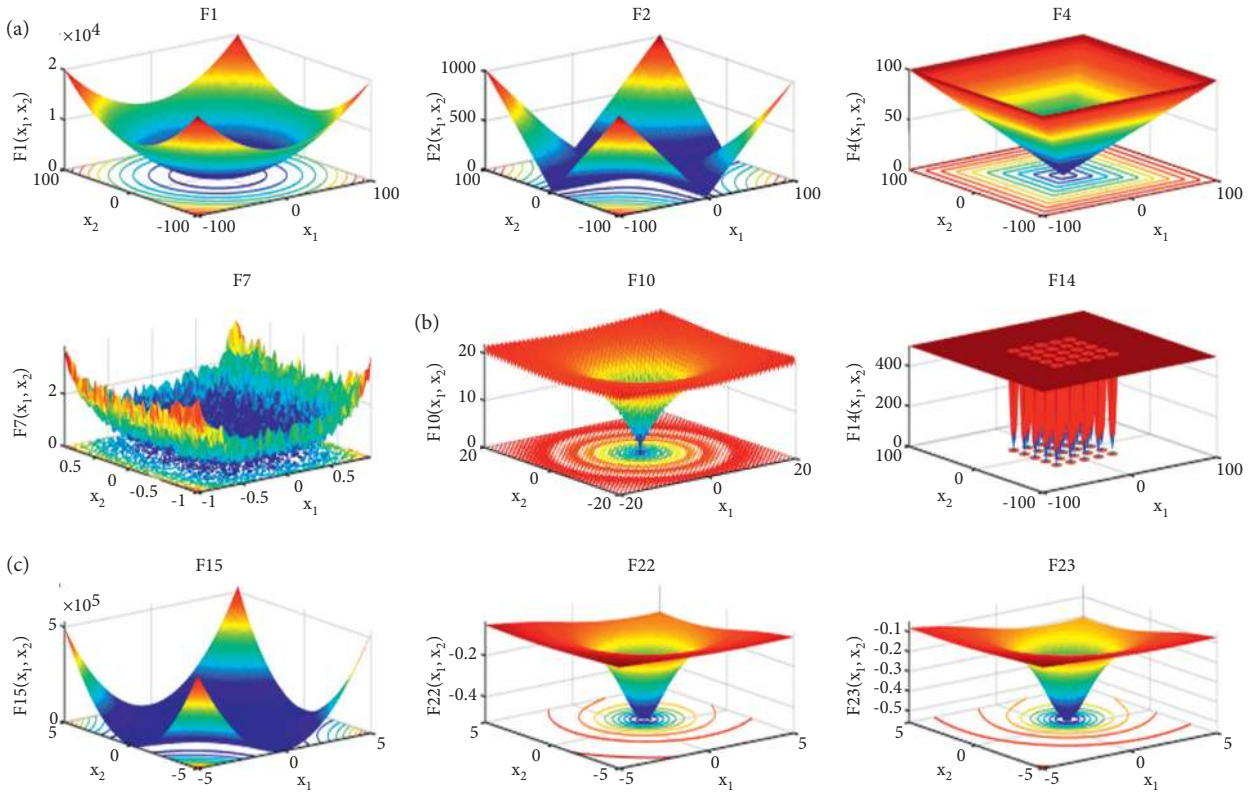

Figure 9: Typical 2D representations of benchmark mathematical functions. (a) Unimodal functions. (b) Multimodal functions. (c) Fixed-dimension multimodal functions.

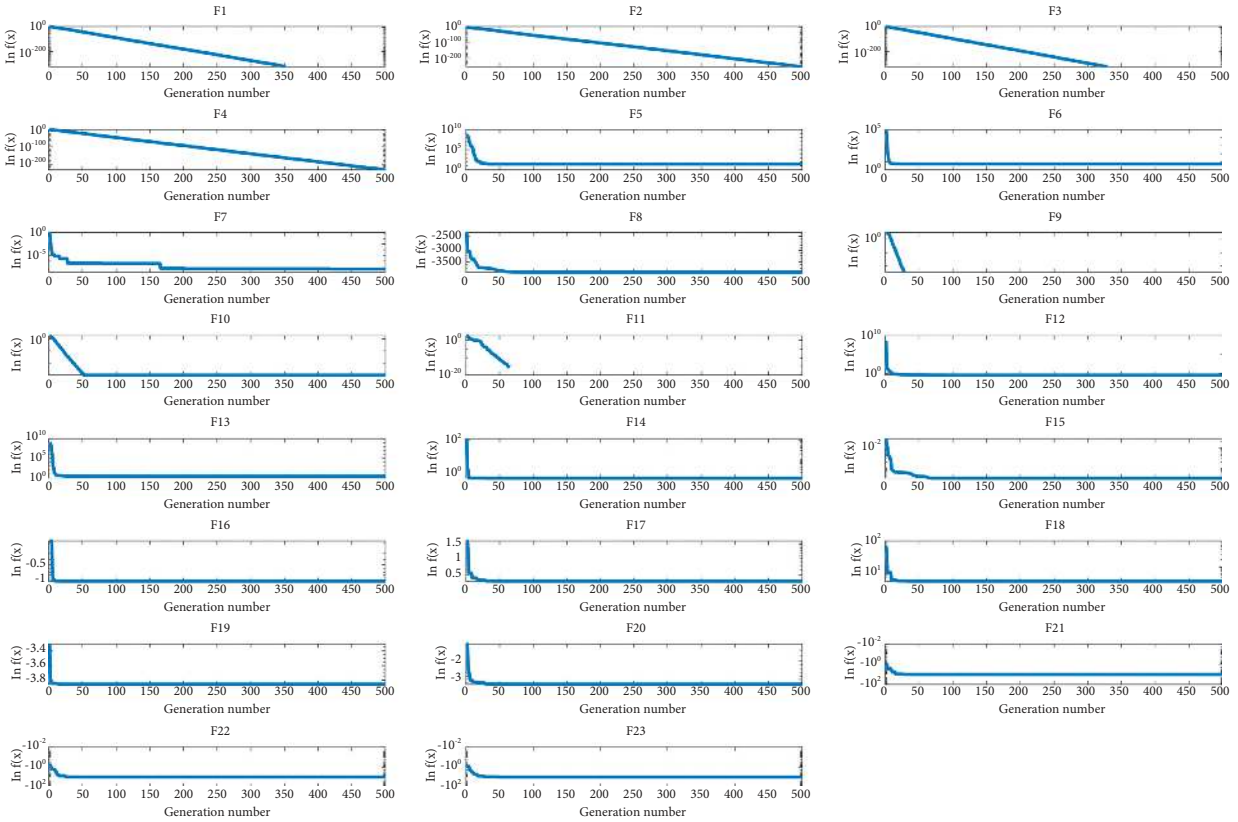

Figure 10: Convergence graphs.

Table 6: Average error rates obtained in the 23 benchmark problems.

F	DOA	WOA	PSO	GSA	DE	FEP
F1	**0.0000*E* + 00**	6.1304*E* − 32	5.9130*E* − 06	1.1000*E* − 17	3.9048*E* − 15	2.4783*E* − 05
F2	**0.0000*E* + 00**	4.6087*E* − 23	1.8323*E* − 03	2.4198*E* − 03	7.1429*E* − 11	3.5217*E* − 04
F3	**0.0000*E* + 00**	2.3435*E* − 08	3.0489*E* + 00	3.8980*E* + 01	3.2381*E* − 12	6.9565*E* − 04
F4	**0.0000*E* + 00**	3.1557*E* − 03	4.7238*E* − 02	3.1978*E* − 01	0.0000*E* + 00	1.3043*E* − 02
F5	1.2565*E* + 00	1.2115*E* + 00	4.2051*E* + 00	2.9367*E* + 00	0.0000*E* + 00	**2.2000*E* − 01**
F6	2.1810*E* − 01	1.3549*E* − 01	4.4348*E* − 06	1.0870*E* − 17	0.0000*E* + 00	**0.0000*E* + 00**
F7	**5.2317*E* − 07**	6.1957*E* − 05	5.3415*E* − 03	3.8887*E* − 03	2.2048*E* − 04	6.1522*E* − 03
F8	5.7185*E* + 01	1.2982*E* + 02	1.1941*E* + 02	**3.1573*E* + 01**	4.2787*E* + 02	4.5477*E* + 02
F9	**0.0000*E* + 00**	**0.0000*E* + 00**	2.0306*E* + 00	1.1291*E* + 00	3.2952*E* + 00	2.0000*E* − 03
F10	**3.8617*E* − 17**	3.2193*E* − 01	1.2001*E* − 02	2.6994*E* − 03	4.6190*E* − 09	7.8261*E* − 04
F11	**0.0000*E* + 00**	1.2565*E* − 05	4.0065*E* − 04	1.2044*E* + 00	**0.0000*E* + 00**	6.9565*E* − 04
F12	1.3179*E* − 02	1.4769*E* − 02	3.0074*E* − 04	7.8244*E* − 02	**3.7619*E* − 16**	4.0000*E* − 07
F13	1.3006*E* − 01	8.2131*E* − 02	2.9022*E* − 04	3.8692*E* − 01	**2.4286*E* − 15**	6.9565*E* − 06
F14	**8.6957*E* − 05**	4.8347*E* − 02	1.1422*E* − 01	2.1130*E* − 01	9.5048*E* − 05	9.5652*E* − 03
F15	**3.2565*E* − 07**	1.1826*E* − 05	1.2043*E* − 05	1.4665*E* − 04	1.4286*E* − 05	8.6957*E* − 06
F16	**0.0000*E* + 00**	1.3043*E* − 06	1.3043*E* − 06	1.3043*E* − 06	1.4286*E* − 06	6.9565*E* − 05
F17	4.7826*E* − 06	3.7391*E* − 06	4.9130*E* − 06	4.9130*E* − 06	5.3810*E* − 06	**0.0000*E* + 00**
F18	**0.0000*E* + 00**	**0.0000*E* + 00**	**0.0000*E* + 00**	**0.0000*E* + 00**	**0.0000*E* + 00**	8.6957*E* − 04
F19	1.2174*E* − 04	1.6696*E* − 04	1.2087*E* − 04	1.2087*E* − 04	—	**0.0000*E* + 00**
F20	**8.6957*E* − 05**	1.4737*E* − 02	2.3330*E* − 03	9.6522*E* − 05	—	2.1739*E* − 03
F21	**8.6957*E* − 06**	1.3495*E* − 01	1.4295*E* − 01	1.8252*E* − 01	9.5238*E* − 06	2.0143*E* − 01
F22	**4.3478*E* − 06**	9.6575*E* − 02	8.4629*E* − 02	3.1240*E* − 02	4.7619*E* − 06	2.1187*E* − 01
F23	**1.7391*E* − 05**	5.1897*E* − 02	2.5352*E* − 02	**1.7391*E* − 05**	1.9048*E* − 05	1.7243*E* − 01

Table 7: Rank of algorithms for unimodal and multimodal problems using MAE.

Algorithm	MAE	Rank
DOA	**5.8803*E* + 01**	**1**
GSA	7.7042*E* + 01	2
PSO	1.2913*E* + 02	3
WOA	1.3194*E* + 02	4
DE	4.3116*E* + 02	5
FEP	4.5561*E* + 02	6

Table 8: Summary of exploitation/exploration rates capability results of DOA Algorithm.

DOA algorithm	Exploitation (*F*1–*F*7) (%)	Exploration (*F*8–*F*23) (%)
Won	57.14	50
Tied	14.29	31.25
Lost	28.57	18.75

problem and the Gear train design problem contain discrete variables. The constraint handling method used is based on [43], where infeasible solutions (that is, at least one constraint is violated) are compared based on only their constraint violation. The constraint handling methods are formulated as follow:

$$F(\vec{x}) = \begin{cases} f(\vec{x}), & \text{if } g_i(\vec{x}) \le 0, \quad \forall i = 1, 2, \ldots, p, \\ f_{\max} + \sum_{i=1}^{p} \langle g_i(\vec{x}) \rangle, & \text{otherwise,} \end{cases} \tag{8}$$

where $f_{\max}$ is the objective function value of the worst feasible solution in the population. Note that the fitness of a feasible solution is equal to its objective function value. On the other hand, the fitness of an infeasible solution is punished. Typically, it is defined as the value of the worst feasible solution in the current population plus the sum of the values obtained when evaluating each constraint violated.

On the other hand, also the DOA algorithm was tested to find the optimal tuning parameters of a PID controller.

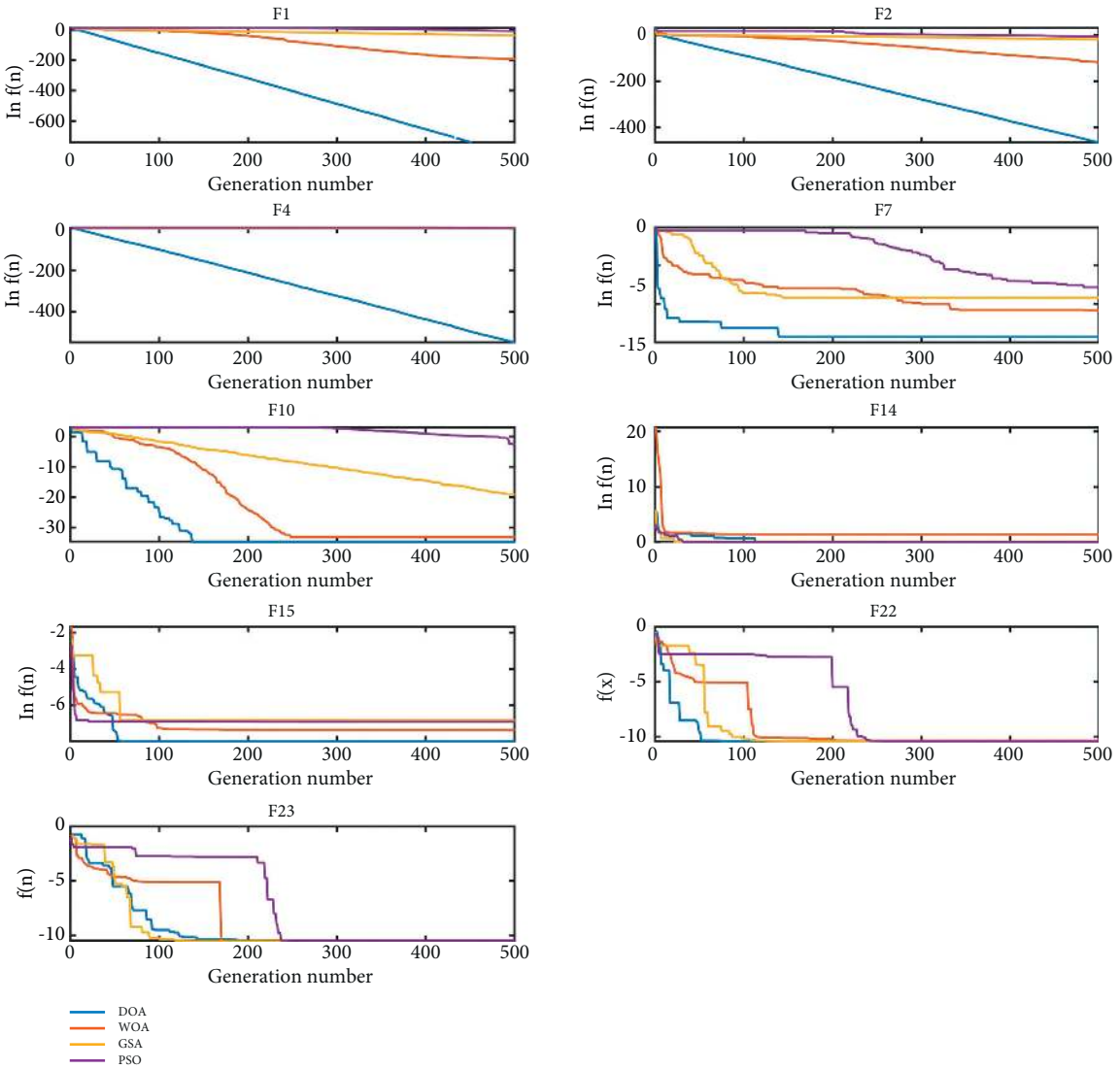

Figure 11: The highest-ranking algorithms taken from the MAE test, DOA, WOA, GSA, and PSO are compared by means of its convergence curves, tested from some of the benchmark problems described in the text.

6.1. Cantilever beam Design Problem. A Cantilever beam consists of five square hollow blocks, as shown in Figure 12. The objective is to minimize weight. In this problem, there are five optimization variables, one for each cantilever that represents the length of their side and includes one optimization constraint [44]. The cantilever weight optimization is formulated in the following equation:

$$\text{Consider } \vec{x} = [\, x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \,],$$

$$\text{Minimize } f(\vec{x}) = 0.06224\left(x_1 + x_2 + x_3 + x_4 + x_5\right),$$

$$\text{Subject to } g(\vec{x}) = \frac{61}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0,$$

$$\text{variable range } 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100. \tag{9}$$

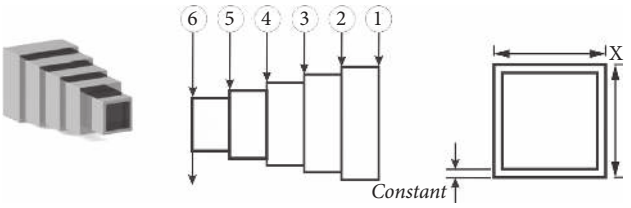

Figure 12: Cantilever beam design problem.

Some of the algorithms that are chosen for comparison are Salp Swarm Algorithm (SSA), Symbiotic Organisms Search (SOS), Method of Moving Asymptotes (MMA), Generalized Convex Approximation (CGA), in its version I and II (*CGA_I* and *CGA_II*, respectively), and Cuckoo Search Algorithm (CSA). The results obtained by DOA and their comparison with the aforementioned state-of-the-art metaheuristics are reported in Table 9, where Table 9 was taken from [44] and updated with DOA's algorithm results.

Table 9: Comparison results for the cantilever design problem, taken from [44] and updated with the DOA's algorithm results.

Algorithm [44]	Optimal values for variables					Optimum weight
	$x1$	$x2$	$x3$	$x4$	$x5$	
DOA	**5.98102**	**4.87800**	**4.46766**	**3.47786**	**2.13461**	**1.30325**
SSA	6.01513	5.30930	4.49501	3.50143	2.15279	1.33996
SOS	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
MMA	6.0100	5.3000	4.49000	3.49000	2.15000	1.34000
GCA_I	6.0100	5.3000	4.49000	3.49000	2.15000	1.34000
GCA_II	6.0100	5.3000	4.49000	3.4900	2.15000	1.34000
CSA	6.0089	5.3049	4.50230	3.50770	2.15040	1.33999

Note that DOA outperforms other techniques when obtaining the lowest weight and shows very competitive results compared to SSA.

6.2. Three-Bar Truss Design Problem. Here, the problem is to design a truss with three bars to minimize its weight. In this test, there are two optimization variables with three optimization constraints, stress, deflection, and buckling. It is formulated as shown in (10). This example is reported in [44] as a highly constrained search space. The overall structure of the three-bar truss is shown in Figure 13.

$$\begin{aligned}
&\text{Consider } \vec{x} = [\, x_1 \;\; x_2 \,] = [\, A_1 \;\; A_2 \,],\\
&\text{Minimize } f(\vec{x}) = \left(2\sqrt{2}\, x_1 + x_2\right) * l,\\
&\text{Subject to } g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0,\\
&\qquad g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0,\\
&\qquad g_3(\vec{x}) = \frac{x_1}{\sqrt{2}x_2 + x_1} P - \sigma \le 0,\\
&\text{variable range } 0 \le x_1, x_2 \le 1,\\
&\text{where } l = 100, P = 2\frac{kN}{cm^2}, \sigma = 2\frac{kN}{cm^2}.
\end{aligned} \tag{10}$$

Table 10 was taken from [44] and updated with DOA's algorithm results. Some of the algorithms that are chosen for comparison are Salp Swarm Algorithm (SSA), Differential Evolution with dynamic stochastic selection (DEDS), Hybridsced Particle Swarm Optimization with Differential Evolution (PSO-DE), Mine Blast Algorithm (MBA), Swarm with intelligent information (Ray and Sain), Tsa Method (Tsa), and Cuckoo Search Algorithm (CSA). The comparison with the aforementioned algorithms shows that the DOA algorithm provides very competitive and very close results compared to SSA and DEDS (the discrepancy is equal to $1E-7$) and outperforming the rest of the algorithms.

6.2.1. Pressure Vessel Design Problem. The goal of this problem is to minimize the total cost. This includes material, forming, and welding of a cylindrical pressure vessel [38]. Here, there are four optimization variables and four optimization constraints, which are the thickness of the shell (Ts), the thickness of the head (Th), the inner radius (R), and the length of the cylindrical section without considering the head (L), by which the pressure vessel is to be fabricated, as shown in Figure 14. Ts and Th are discrete variables in multiples of 0.0625 in., while R and L are real variables. The mathematical formulation of the optimization problem is described as follows:

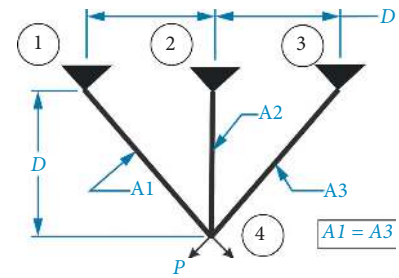

Figure 13: Three-bar truss design problem.

$$\begin{aligned}
&\text{Consider } \vec{x} = [\, x_1 \;\; x_2 \;\; x_3 \;\; x_4 \,] = [\, T_s \;\; T_h \;\; R \;\; L \,]\\
&\text{Minimize } f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2\\
&\qquad\qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3,\\
&\text{Subject to } g_1(\vec{x}) = -x_2 + 0.0193x_3 \le 0\\
&\qquad g_2(\vec{x}) = -x_1 + 0.00954x_3 \le 0\\
&\qquad g_3(\vec{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \le 0\\
&\qquad g_4(\vec{x}) = -x_4 - 240 \le 0\\
&\text{variable range } 0 \le x_1 \le 99,\\
&\qquad 0 \le x_2 \le 99,\\
&\qquad 10 \le x_3 \le 200,\\
&\qquad 10 \le x_4 \le 200,
\end{aligned} \tag{11}$$

Some of the algorithms that are chosen for comparison are Differential Evolution (DE), Genetic Algorithm (GA), Whale Optimization Algorithm (WOA), Particle Swarm

Table 10: Comparison results for the three-bar truss design problem, taken from [44] and updated with the DOA's algorithm results.

Algorithm [44]	Optimal values for variables		Optimum weigth
	$x1$	$x2$	
DOA	0.788675095	0.40824840	263.8958434
SSA	0.78866541	0.40827578	263.8958434
DEDS	0.78867513	0.40824828	263.8958434
PSO-DE	0.78867510	0.40824820	263.8958433
MBA	0.78856500	0.40855970	263.8958522
Ray and sain	0.79500000	0.39500000	264.3000000
Tsa	0.78800000	0.40800000	263.6800000 (infeasible)
CSA	0.78867000	0.40902000	263.9716000

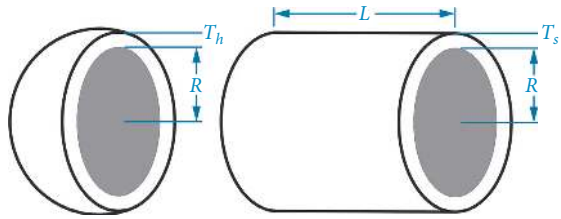

Figure 14: pressure vessel design problem.

Optimization (PSO), among others [38]. Table 11 was taken from [38] and updated with DOA's algorithm results. In this table, the comparison results show that the DOA algorithm is ranked as the first best solution obtained.

6.3. Discrete Engineering Problem-Gear train Design Problem. Here, the objective is to find the optimal number of the teeth of a four gear train while minimizing the gear ratio, as shown in Figure 15, where its four parameters are discrete [21]. In order to handle discrete values, each search agent was rounded to the nearest integer number before the fitness evaluation. The design engineering constraint is defined as the number of teeth on any gear that should only be in the range of [12, 60]. Accordingly, the optimization problem can be formulated as follows:

$$\text{Consider } \vec{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4] = [n_A \;\; n_B \;\; n_C \;\; n_D],$$

$$\text{Minimize } f(\vec{x}) = \left(\frac{1}{6.931} - \frac{x_2 x_3}{x_1 x_4}\right)^2,$$

$$\text{variable range } 12 \le x_1, x_2, x_3, x_4 \le 60, \tag{12}$$

Some of the algorithms that are chosen for comparison are Ant Lion Optimizer (ALO), Cuckoo Search Algorithm (CSA), Mine Blast Algorithm (MBA), Interior Search Algorithm (ISA), Genetic Algorithm (GA), Artificial Bee Colony (ABC), and Augmented Lagrange Multiplier (ALM) [21]. Table 12 was taken from [21] and updated with DOA's algorithm results. It shows that the DOA algorithm gives competitive results for numbers of function evaluations and is suitable to solve discrete constrained problems.

6.4. Tuning of a Proportional-Integral-Derivative (PID) Controller: Sloshing Dynamics Problem. Sloshing dynamics is a well-studied phenomenon in fluid dynamics. It is related to the movement of a liquid inside another object, altering the system dynamics [45]. Sloshing is an important effect on ships, spacecraft, aircraft, and trucks carrying liquids, as it causes instability and accidents. Sloshing dynamics can be depicted as a Ball and Hoop System (BHS). This effect illustrates the dynamics of a steel ball that is free to roll on the inner surface of a rotating circular hoop. The ball exhibits an oscillatory motion caused by the continuously rotated hoop through a motor. The ball will tend to move in the direction of the hoop rotation and will fall back, at some point, when gravity overcomes the frictional forces. The BHS behavior can be described by seven variables: hoop radius (R), hoop angle (θ), input torque to the hoop (T(t)), ball position on the hoop (y), ball radius (r), ball mass (m), and ball angles with vertical (slosh angle) (ψ). A schematic representation is shown in Figure 16. The transfer function of the BHS system, taken from [46,47], is formulated in equation (13), where θ is the input and y is the output of the BHS system.

$$G_{\text{BHS}}(s) = \frac{y(s)}{\theta(s)} = \frac{1}{s^4 + 6s^3 + 11s^2 + 6s}. \tag{13}$$

Pareek et al. studied the optimal tuning of a Proportional-Integral-Derivative (PID) controller using metaheuristic algorithms [47], specifically by using Bacteria Foraging Optimization (BFO), Particle Swarm Optimization (PSO), and Artificial Bee Colony Algorithm (ABC). In this study, we updated Tables 13 and 14, taken from [47], with the DOA algorithm's results.

The transient response parameters of the Proportional-Integral-Derivative (PID) controller are Rise time, Settling time, Peak time, and Peak overshoot [48]. The PID controller

Table 11: Comparison results for pressure vessel design problem, taken from [38] and updated with the DOA's algorithm results.

Algorithm [38]	Optimal values for variables				Optimum weight
	T_s	T_h	R	L	
DOA	**0.812500**	**0.437500**	**42.09845**	**176.6366**	**6059.7143**
ACO (Kaveh and Telataheri)	0.812500	0.437500	42.103624	176.572656	6059.0888 (infeasible)
DE (Huang et al.)	0.812500	0.437500	42.098411	176.637690	6059.7340
WOA	0.812500	0.437500	42.0982699	176.638998	6059.7410
ES (Montes and Coello)	0.812500	0.437500	42.098087	176.640518	6059.7456
GA (Coello and Montes)	0.812500	0.437500	42.097398	176.654050	6059.9463
PSO (He and Wang)	0.812500	0.437500	42.091266	176.746500	6061.0777
GA (Coello)	0.812500	0.434500	40.323900	200.000000	6288.7445
GA (deb and gene)	0.812500	0.500000	48.329000	112.679000	6410.3811
Improved HS	1.125000	0.625000	58.290150	43.6926800	7197.730
Lagrangian multiplier (Kannan)	1.125000	0.625000	58.291000	43.6900000	7198.0428
Branch-bound (Sandgren)	1.125000	0.625000	47.700000	117.701000	8129.1036
GSA	1.125000	0.625000	55.9886598	84.4542025	8538.8359

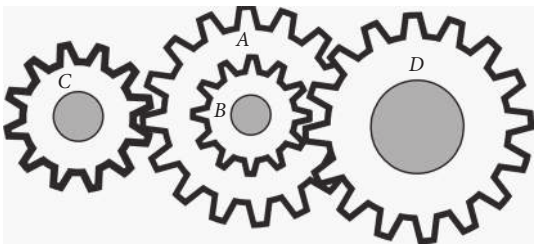

Figure 15: Gear train design problem.

Table 12: Comparison results of the gear train design problem, taken from [21] and updated with the DOA's algorithm results.

Algorithm [21].	Optimal values for variables				$f_{\min}$	Max. eval
	n_A	n_B	n_C	n_D		
DOA	43	16	19	49	$2.7009e-12$	130
ALO	49	19	16	43	$2.7009e-12$	120
CSA	43	16	19	49	$2.7009e-12$	5000
MBA	43	16	19	49	$2.7009e-12$	10000
ABC	19	16	44	49	$2.78e-11$	40000
GA	33	14	17	50	$1.362e-9$	N/A
ALM	33	15	13	41	$2.1469e-8$	N/A

Ant Lion Optimizer (ALO), Cuckoo Search Algorithm (CSA), Mine Blast Algorithm (MBA), Artificial Bee Colony (ABC), Genetic Algorithm (GA), and Augmented Lagrange Multiplier (ALM).

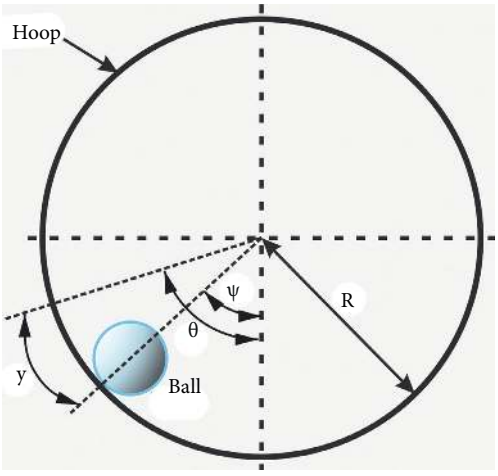

Figure 16: Schema of the ball and hoop system.

Table 13: Comparison results of optimized PID parameters, taken from [47] and updated with the DOA's algorithm results.

Algorithm [47]	Parameter		
	Kp	Ki	Kd
DOA	3.6677	0.01	4.9852
BFO	3.6616	0.3112	3.7334
PSO	4.0993	0.0325	2.9844
ABC	8.5164	0.0043	9.3419

Table 14: Comparison results of transient response parameters, taken from [47] and updated with the DOA's algorithm results.

Algorithm [47]	Transient parameters			
	Rise time (sec)	Settling time (sec)	Peak time (sec)	Peak overshoot (%)
DOA	**2.0044**	**3.2900**	10.6541	**0.3452**
BFO	2.0419	23.4616	5.2414	16
PSO	2.0135	72809	4.5558	14
ABC	1.0013	7.4979	2.4653	29

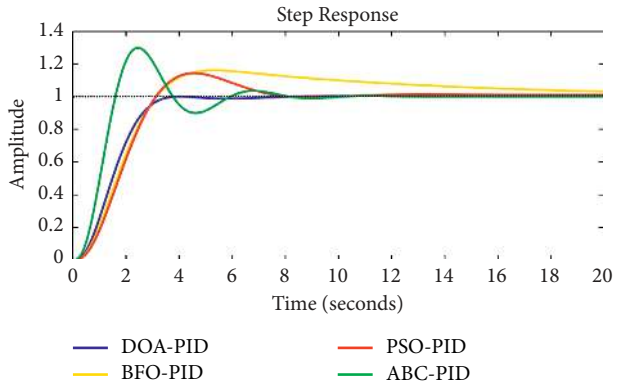

Figure 17: Comparative results of step response for the PID controller.

is designed to minimize the overshoot and settling time so that the liquid can remain as stable as possible under any perturbance and if it moves, it can rapidly go back to the steady state. It is to be noticed that the DOA outperforms the aforementioned algorithms, obtaining the lowest rising and settling time value, as well as the peak overshoot, see Table 14. In addition, in Figure 17, we can see the PID controller step response for the four algorithms. Note that the DOA-PID (blue line) is more stable with very fine control without exceeding the setpoint (dotted line).

7. Conclusions

This study presented a novel population-based optimization algorithm based on three different hunting strategies of the *Canis lupus* dingo. These strategies, attacking by persecution, grouping tactics, and scavenging behaviors, were carefully designed to guarantee the exploration and exploitation of the search state and evaluated by 23 mathematical benchmark functions. The DOA showed an exploitation and exploration accumulated rate of 71.43% and 81.25%, respectively, that outperforms or ties the other algorithms. DOA uses two parameters, P and Q, to indicate the probability of the algorithm to choose between the hunting or scavenger strategy. It is to be noticed that regardless of P and Q values, the algorithm converges to the solution due to the incorporation of the survival strategy. The DOA performance was compared with five well-known state-of-the-art metaheuristic methods available in the literature: Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO), Gravity Search Algorithm (GSA), Differential Evolution (DE), and Fast Evolution Programming (FEP). The statistical analysis mean absolute error (MAE) was conducted to measure the performance of the DOA and the previously mentioned algorithms. DOA was found to be highly competitive in the majority of the test functions. The capabilities of DOA were also tested with classical engineering problems (design of a cantilever beam, design of a three-bar truss and design of a pressure vessel). The results obtained by DOA in most cases overcomes several well-known metaheuristics Additionally, the DOA demonstrated its capability to find the optimal tuning parameters of a PID controller, which is the rise time, the settling time, the peak time, and the peak overshoot, which were efficiently optimized for the sloshing dynamics problem. In order to expand the algorithm scope, it was also tested with an engineering discrete problem (design of a gear train), showing competitive results. Finally, this paper opens up several research directions for future studies. They include the incorporation of self-adaptive parameters, a method to handle multiobjective optimization problems with large problem instances using parallelization strategies, e.g., GPU computing and multicore resources.

Data Availability

The source code used to support the findings of this study have been deposited in the Mathworks repository (https://www.mathworks.com/matlabcentral/fileexchange/98124-dingo-optimization-algorithm-doa).

Conflicts of Interest

The authors declared no potential conflicts of interest with respect to the research, authorship, funding, and/or publication of this article.

Acknowledgments

This project was supported by Instituto Politécnico Nacional (IPN) through Grant SIP-no. 20200068 and SIP-no. 20211364. The third author acknowledges support from CONACYT to pursue his graduate studies of Master in Advanced Technology at IPN-CICATA Altamira.

References

[1] E. K. P. Chong and S. H. Żak, *Wiley-Interscience Series in Discrete Mathematics and Optimization*, John Wiley & Sons, Ltd., Hoboken, NJ, USA, 2011.

[2] J. H. Lee, J. Shin, and M. J. Realff, "Machine learning: overview of the recent progresses and implications for the process systems engineering field," *Computers & Chemical Engineering*, vol. 114, pp. 111–121, 2018.

[3] J. K. Mandal, P. Dutta, and S. Mukhopadhyay, "Advances in intelligent computing," *Studies in Computational Intelligence*, vol. 687, 2019.

[4] P. Hansen, N. Mladenović, J. Brimberg, and J. A. M. Pérez, *Variable Neighborhood Search*, Springer, Berlin, Germany, 2019.

[5] M. Bereta, "Regularization of boosted decision stumps using tabu search," *Applied Soft Computing*, vol. 79, pp. 424–438, 2019.

[6] D. Delahaye, S. Chaimatanan, and M. Mongeau, *Simulated Annealing: From Basics to Applications*, Springer, Berlin, Germany, 2019.

[7] H. R. Lourenço, O. C. Martin, and T. Stützle, *Iterated Local Search: Framework and Applications*, Springer, Berlin, Germany, 2019.

[8] R. Alipour-Sarabi, Z. Nasiri-Gheidari, F. Tootoonchian, and H. Oraee, "Improved winding proposal for wound rotor resolver using genetic algorithm and winding function approach," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1325–1334, 2019.

[9] K. Nag and N. R. Pal, *Genetic Programming for Classification and Feature Selection*, Springer, Berlin, Germany, 2019.

[10] H. Qu, X.-Y. Ai, and L. Wang, "Optimizing an integrated inventory-routing system for multi-item joint replenishment and coordinated outbound delivery using differential evolution algorithm," *Applied Soft Computing*, vol. 86, Article ID 105863, 2020.

[11] W. Wang, Z. Xiong, D. Niyato, P. Wang, and Z. Han, "A hierarchical game with strategy evolution for mobile sponsored content and service markets," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 472–488, 2019.

[12] M. López-Ibáñez, T. Stützle, and M. Dorigo, *Ant Colony Optimization: A Component-Wise Overview*, Springer, Berlin, Germany, 2018.

[13] M. Shehab, A. T. Khader, and M. A. Al-Betar, "A survey on applications and variants of the cuckoo search algorithm," *Applied Soft Computing*, vol. 61, pp. 1041–1059, 2017.

[14] S. Mirjalili, J. Song Dong, A. Lewis, and A. S. Sadiq, *Particle Swarm Optimization: Theory, Literature Review, and Application in Airfoil Design*, Springer, Berlin, Germany, 2020.

[15] M. N. Ab Wahab, S. Nefti-Meziani, and A. Atyabi, "A comprehensive review of swarm optimization algorithms," *PLoS One*, vol. 10, no. 5, p. e0122827, 2015.

[16] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[17] J. Del Ser, E. Osaba, D. Molina et al., "Bio-inspired computation: where we stand and what's next," *Swarm and Evolutionary Computation*, vol. 48, pp. 220–250, 2019.

[18] M. D. Li, H. Zhao, X. W. Weng, and T. Han, "A novel nature-inspired algorithm for optimization: Virus colony search," *Advances in Engineering Software*, vol. 92, pp. 65–88, 2016.

[19] M. Sulaiman, A. Salhi, A. Khan, S. Muhammad, and W. Khan, "On the theoretical analysis of the plant propagation algorithms," *Mathematical Problems in Engineering*, vol. 2018, Article ID 6357935, 8 pages, 2018.

[20] H. Shareef, A. A. Ibrahim, and A. H. Mutlag, "Lightning search algorithm," *Applied Soft Computing*, vol. 36, pp. 315–333, 2015.

[21] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015.

[22] M. Yazdani and F. Jolai, "Lion optimization algorithm (loa): a nature-inspired metaheuristic algorithm," *Journal of Computational Design and Engineering*, vol. 3, no. 1, pp. 24–36, 2016.

[23] G. Dhiman and V. Kumar, "Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications," *Advances in Engineering Software*, vol. 114, pp. 48–70, 2017.

[24] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[25] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing & Applications*, vol. 27, no. 4, pp. 1053–1073, 2015.

[26] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. D. S. Coelho, "Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization," *Expert Systems with Applications*, vol. 47, pp. 106–119, 2016.

[27] A. Kaveh, *Advances in Metaheuristic Algorithms for Optimal Design of Structuresdoi*.

[28] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020.

[29] G.-G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, no. 2, pp. 151–164, 2018.

[30] J. Tu, H. Chen, M. Wang, and A. H. Gandomi, "The colony predation algorithm," *Journal of Bionics Engineering*, vol. 18, no. 3, pp. 674–710, 2021.

[31] A. F. Peña-Delgado, H. Peraza-Vázquez, J. H. Almazán-Covarrubias et al., "A novel bio-inspired algorithm applied to selective harmonic elimination in a three-phase eleven-level inverter," *Mathematical Problems in Engineering*, vol. 2020, Article ID 8856040, 10 pages, 2020.

[32] Y. Meraihi, A. B. Gabis, S. Mirjalili, and A. Ramdane-Cherif, "Grasshopper optimization algorithm: theory, variants, and applications," *IEEE Access*, vol. 9, pp. 50001–50024, 2021.

[33] H. Nguyen and X.-N. Bui, "A novel hunger games search optimization-based artificial neural network for predicting

ground vibration intensity induced by mine blasting," *Natural Resources Research*, vol. 30, no. 5, pp. 3865–3880, 2021.
[34] B. L. Allen and L. K.-P. Leung, "Assessing predation risk to threatened fauna from their prevalence in predator scats: dingoes and rodents in arid Australia," *PLoS One*, vol. 7, no. 5, p. e36426, 2012.
[35] S. M. Jackson, C. P. Groves, P. J. S. Fleming et al., "The wayward dog: is the australian native dog or dingo a distinct species?" *Zootaxa*, vol. 4317, no. 2, p. 201, 2017.
[36] M. S. Crowther, M. Fillios, N. Colman, and M. Letnic, "An updated description of the Australian dingo (C anis dingo Meyer, 1793)," *Journal of Zoology*, vol. 293, no. 3, pp. 192–203, 2014.
[37] P. N. Suganthan, N. Hansen, J. J. Liang et al., "Problem definitions and evaluation criteria for the cec 2005. special session on real-parameter optimization," *Natural Computing*, pp. 341–357, 2005.
[38] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
[39] J. Kennedy, *Encyclopedia of Machine Learning and Data Mining*, Springer, Berlin, Germany, 2017.
[40] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "Gsa: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
[41] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[42] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
[43] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.
[44] 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," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.
[45] R. Ibrahim, *Liquid Sloshing Dynamics: Theory and Applications*, Cambridge University Press, Cambridge, UK, 2005.
[46] N. Jain, G. Parmar, R. Gupta, and I. Khanam, "Performance evaluation of gwo/pid approach in control of ball hoop system with different objective functions and perturbation," *Cogent Engineering*, vol. 5, no. 1, p. 1465328, 2018.
[47] S. Pareek, M. Kishnani, and R. Gupta, "Optimal tuning of pid controller using meta heuristic algorithms," in *Proceedings of the 2014 International Conference on Advances in Engineering Technology Research*, pp. 1–5, Singapore, March 2014.
[48] G. C. Goodwin, S. Graebe, and M. E. Salgado, *Control System Design*, Pearson, London, UK, 1st edition, 2020.