

An island parallel Harris Hawks Optimization algorithm

Tansel Dokeroglu^{1*} and Ender Sevinc^{2†}

^{1*}Software Engineering Department, Cankaya University,
Etimesgut, Ankara, Turkey.

^{2*}Computer Engineering Department, Middle East Technical
University, Ankara, Turkey.

*Corresponding author(s). E-mail(s): tdokeroglu@cankaya.edu.tr;

Contributing authors: ender@ceng.metu.edu.tr;

†These authors contributed equally to this work.

Abstract

The HHO method is an impressive optimization algorithm that makes use of unique mathematical approaches. This study proposes an Island Parallel HHO (IP-HHO) version of the algorithm for optimizing continuous multidimensional problems for the first time in the literature. To evaluate the performance of the IP-HHO method, thirteen unimodal and multimodal benchmark problems with four different dimensions (30, 100, 500, and 1000) are evaluated. The implementation of this novel algorithm took into account the investigation, exploitation, and avoidance of local optima issues effectively. Parallel computation, as can be seen, provides a multi-swarm environment for thousands of hawks simultaneously. On all issue cases, we were able to enhance the performance of the sequential version of the HHO algorithm. As the number of processors increases, the suggested IP-HHO method enhances its performance while retaining scalability and improving its computation speed. The IP-HHO method outperforms the other state-of-the-art metaheuristic algorithms on average as the size of the dimensions grows.

Keywords: Optimization, Harris Hawk, multi-swarm, multi-dimensional

1 Introduction

Metaheuristic algorithms are among the best tools to tackle large multi-dimensional continuous optimization problems [1]. Genetic Algorithms (GA) [2], Ant Colony Optimization [3], Particle Swarm Optimization (PSO) [4], Simulated Annealing (SA) [5], Differential Evolution (DE) [6], and Tabu Search (TS) [7] are the classical metaheuristic algorithms that have been the subject of numerous studies, and these algorithms have provided very good results for challenging optimization problems over the last 30 years.

These metaheuristics feature distinct optimization problem-solving strategies. However, according to Wolpert's No Free Lunch (NFL) theory, a single metaheuristic cannot handle all situations optimally [8]. What works well for some problems may not work as well for other problems as it does for other data sets. Therefore, many researchers are still studying on developing new metaheuristic algorithms that work better than the existing ones [9, 10].

The Harris Hawks Optimizer (HHO) is a novel metaheuristic inspired on hawk hunting behavior [11, 12]. The hawks attack the rabbit from separate directions, surprising the victim. These sophisticated birds utilize a variety of designs to catch their prey. The HHO metaheuristic optimization approach mimics the hunting behaviors of Harris hawks. The HHO metaheuristic is substantiated by the performance of cutting-edge metaheuristics on a variety of benchmark challenges. The results demonstrate that the HHO algorithm produces competitive results. In this paper, we propose a novel Island Parallel multi-swarm HHO method (IP-HHO) for optimizing continuous multidimensional problem instances in order to improve on the outcomes of the conventional HHO technique. The suggested IP-HHO method was created with the help of Message Passing Interface (MPI) libraries. On each processor, multi-swarms are produced, and a very broad search region may be examined. Before obtaining the best result, metaheuristic algorithms must do numerous fitness computations repeatedly. This is one of the most significant disadvantages of these algorithms. This issue should be resolved in order to accelerate optimization and perform a more efficient search procedure. Parallel metaheuristic algorithms offer the infrastructure required to compute a large number of fitness values [13, 14]. A better optimization procedure can ideally be supplied with different initial beginning regions of multi-swarms.

Stagnation at a local optimum is one of the key issues preventing metaheuristic algorithms from getting the best outcomes. If your solution is trapped in a local optimum, no matter how many times you produce new solutions that are neighbors of the present ones, it is impossible to develop that solution without deleting that area. Perturbation (on a huge scale, damaging the present solution vector) or starting with fresh points are ways for resolving this challenge and finding better answers. We also discovered a way to prevent stagnation in our study by starting our solutions at parallel nodes from separate positions in the solution space. Another issue with the traditional HHO method is immature convergence. It was feasible to better explore/exploit the issue space using the strategies described in our study.

To adjust the optimal variables of the HHO, a simple parameter tuning approach for the bunnies' energy levels and jump values is also provided. On continuous optimization benchmark issue cases, the proposed technique is compared to sequential versions of state-of-the-art nature-inspired algorithms. The findings demonstrate that the IP-HHO method is scalable and gives competitive solutions when compared to other metaheuristic algorithms. We ran 13 benchmark problem situations (with dimensions of 30, 100, 500, and 1000). In comparison to other contemporary metaheuristics, the findings of 52 function test cases with varying dimensions revealed that the IP-HHO algorithm is the best algorithm in 43 of the test cases and the second or third best algorithm in the remaining issue instances.

In Section 2, related studies on metaheuristic algorithms that solve the multi-dimensional numerical function optimization are summarized. Section 3 introduces the proposed IP-HHO algorithm. Section 4 presents the evaluation of our experimental results. Concluding remarks and future work are provided in the last Section.

2 Related work

In this section of our paper, we examine current work on the metaheuristic algorithm HHO. Louis Lefebvre devised a novel approach for assessing hawk intelligence during hunting activity [17]. Hawks are among the most intelligent of all birds [18]. Along with its swarm, the hawk survives in groups and engages in cooperative foraging activities [19]. The hawks have exceptional chasing and attacking ability. These sophisticated birds work together as predators. Hawks mostly use surprise attacks to grab prey (known as seven kills). The hawks will attack a fleeing victim from separate directions in a concerted attack. They can finish the attack in a few seconds or make fast dives (seven kills). The hawks have various hunting methods and prey escape strategies. The biggest benefit is that hawks may follow the rabbit for an extended period of time, increasing its susceptibility. The prey is unable to recover from its defensive/escape efforts.

In 2019, The HHO algorithm was proposed by Heidari et al. [11].

Hawks' cooperative conduct and pursuing technique served as their primary influence. The HHO optimizer's performance is validated by comparing it to other approaches on a variety of benchmark and technical challenges. In comparison to current metaheuristics, the results demonstrate that the HHO can reach promising solutions. In their study, Dokeroglu et al. highlighted fourteen exceptional metaheuristics that have emerged in the previous twenty years that are not among the classics [1, 9]. The HHO algorithm was chosen as one of the elegant new metaheuristics because to its performance, specialized operators, inter-individual interaction strategies, and stagnation prevention approaches.

Gharehchopogh & Abdollahzadeh introduced a novel strategy for traveling salesmen utilizing HHO with random-key encoding [20]. For neighborhood search, ten distinct operators are employed. To validate the algorithm's performance, 80 issue datasets are evaluated. The findings demonstrate the suggested

algorithm's superior performance. Bairathi & Gopalani have suggested a new HHO [21]. The HHO's performance is compared to that of the PSO, DE, GWO, and Whale Optimization Algorithms (WOA). One of the most efficient optimization methods is said to be the HHO. Sabeena & Abraham present a deep learning approach for image forgery detection by copying that makes use of the HHO algorithm [22]. The results show that the proposed method is the most effective in terms of recognition. Dokeroglu et al. proposed a new HHO algorithm for multi-objective binary classification [23]. In their study, the authors create new exploration and exploitation operators while minimizing the number of features and maximizing the accuracy level of the results. Too et al. created a binary version of the HHO to help with feature selection [24]. To convert variables into binary format, the proposed algorithm employs S and V -shaped transfer functions. This study also proposes a new version of HHO, quadratic binary HHO. A comparison of the binary algorithms DE, GA, Binary Salp Swarm (SSA), and flower pollination is performed. The experimental results show that, depending on the feature size and fitness values, the HHO is superior in classification. Zhang et al. created a new HHO feature selection algorithm [25]. The SSA is incorporated into the HHO metaheuristic. The proposed HHO algorithm performs well in terms of exploration and exploitation. The algorithm converges quickly. Chen et al. created a hybrid HHO algorithm by combining chaos strategy, multi-population strategy, and DE to solve the HHO's local optimization problem [27]. The multi-population strategy has the potential to significantly improve global search. Yildiz et al. incorporate the Nelder-Mead local search into the HHO algorithm [28]. The algorithm is tested on well-known benchmark problem instances. Yildiz et al. propose an HHO algorithm for solving optimization problems in the automotive industry in another study [29]. Song et al. proposed an enhanced HHO algorithm in order to improve its performance in 2021 [30].

Dokeroglu et al. proposed a collection of parallel metaheuristic algorithms (based on Teaching Learning-Based Optimization (TLBO) and Artificial Bee Colony (ABC)) [26]. The proposed parallel Island algorithm outperforms the sequential versions of metaheuristics. Schryen proposes a new parallel optimization framework [33]. Interested readers can find detailed information about parallel metaheuristic techniques, advances, and new trends in the studies by Albas et al. [14, 31] and Crainic et al. [32].

The quality of the results obtained with parallel algorithms will improve as new hardware architectures improve. Because the algorithm we developed is a parallel metaheuristic application in this sense, we thought it would be appropriate to refer to the literature studies that were closest to us. Given the algorithm we created, it is safe to say that the IP-HHO algorithm is the first application of the HHO to the optimization of continuous multidimensional problems.

3 Proposed Island Parallel HHO algorithm

The HHO mimics the hawks’ exploratory and exploitative hunting methods. Because the HHO is a gradient-free optimization algorithm, it is easily applicable to a wide range of optimization problems. Figure 1 presents the main steps of the HHO.

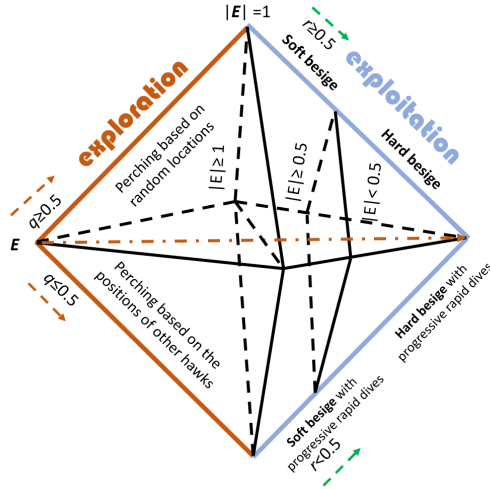


Fig. 1 The representation of the exploration and exploitation steps of HHO algorithm (including perching, soft/hard besiege). q is the probability of selecting a perching strategy, r is a random value between 0 and 1, and E is the escaping energy of the prey.

Hawks track prey during the HHO’s exploration phase, and they are the population’s solutions. The population’s best hawk becomes the best solution. The hawks primarily use two strategies to locate the bait. q is a parameter for perching strategy and the hawks attack in accordance with the location of hawks and the prey (see Equation (1)) when $q < 0.5$, or wait (see Equation (1)) when the value of $q \geq 0.5$.

$$X(t + 1) = \begin{cases} X_{rand}(t) - r_1 |X_{rand}(t) - 2r_2X(t)| & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)) & q < 0.5 \end{cases} \quad (1)$$

the current hawk is $X(t)$, the position of the hawk is $X(t + 1)$, the prey is $X_{rabbit}(t)$. q , r_1 , r_2 , r_3 , and r_4 , are values between (0,1), UB and LB are the bounds, $X_{rand}(t)$ is a random hawk, and X_m is the mean location of the population in the Eq.1. Random locations are generated inside the range (LB , UB). The solutions are generated according to a random location and the locations of the hawks in the population. A random movement is added to the LB and a random coefficient provides diversification and explores diverse regions. Equation determines the population’s average location (2):

$$X_m(t) = \frac{1}{N} \sum_{i=1}^N X_i(t) \quad (2)$$

N is the population size. The HHO algorithm alternates between exploration and exploitation. The energy (E) is modeled as below:

$$E = 2E_0(1 - \frac{t}{T}) \quad (3)$$

E_0 is the initial energy, E is the energy of the rabbit, T is the iterations. The value of E decreases during the iterations. If the escaping energy is more than one, the hawks explore otherwise, they exploit. During exploitation, the hawks hunt their prey by surprise pounces [19]. Hawks apply chase strategies according to the escaping behavior of the prey. r value denotes the probability of the prey to escape, if this value is ($r < 0.5$) the prey can escape. The hawks can surround the prey from different directions. They get closer to the prey and increase the chance of capturing the rabbit. The prey can escape but loses energy during these activities. The hawk can use the besiege process as well. If $E \geq 0.5$, the soft besiege occurs otherwise, the hard besiege happens.

The prey has enough energy when $r \geq 0.5$ and $E \geq 0.5$. But it might take misleading jumps. Then the hawk encircles it softly. This makes the prey exhausted, and the hawk can perform surprise pounce (see Equation 4).

$$X(t+1) = \Delta X(t) - E |JX_{rabbit}(t) - X(t)| \quad (4)$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \quad (5)$$

$\Delta X(t)$ denotes the difference between the positions of the rabbit at each iteration t . r_5 is a value between 0 and 1, and jump strength of the rabbit is $J = 2(1 - r_5)$. The hawk can hardly surround, attack and pounce the rabbit when $r \geq 0.5$ and $E < 0.5$. This is formulated in Equation (6) and shown in Figure 2):

$$X(t+1) = X_{rabbit}(t) - E |\Delta X(t)| \quad (6)$$

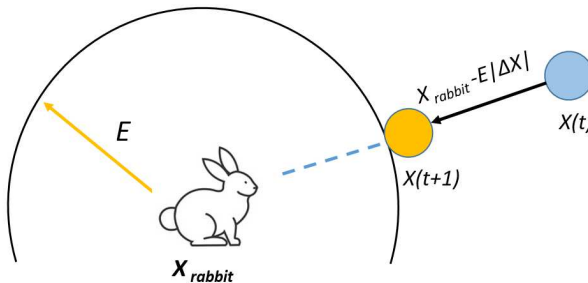


Fig. 2 The representation of vectors for hard besiege

Levy Flights (LF) are used when the rabbit has the energy to escape [19] for modeling these escaping patterns. The LF mimics the zigzag motions of prey during the escape. These movements are accepted to be optimal for searching methods of foragers/predators in non-destructive foraging situations [34, 35]. Monkeys and sharks also use such LF-based patterns [36–39]. A soft besiege can be performed using Equation (7). Each result is compared to the previous one to check the quality (see Equation 8)

$$Y = X_{rabbit}(t) - E |JX_{rabbit}(t) - X(t)| \tag{7}$$

$$Z = Y + S \times LF(D) \tag{8}$$

where D is the dimension, S is the vector of size of D . and LF is calculated by Equation (9) [40]:

$$LF(x) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1 + \beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right)^{\frac{1}{\beta}} \tag{9}$$

where u, v are values between (0,1), β is 1.5. The final position of the hawk can be evaluated by Equation (10) in the soft besiege phase.

$$X(t + 1) = \begin{cases} Y & \text{if } F(Y) < F(X(t)) \\ Z & \text{if } F(Z) < F(X(t)) \end{cases} \tag{10}$$

A soft besiege step of a single hawk is shown in Figure 3. At each step, a better position is selected.

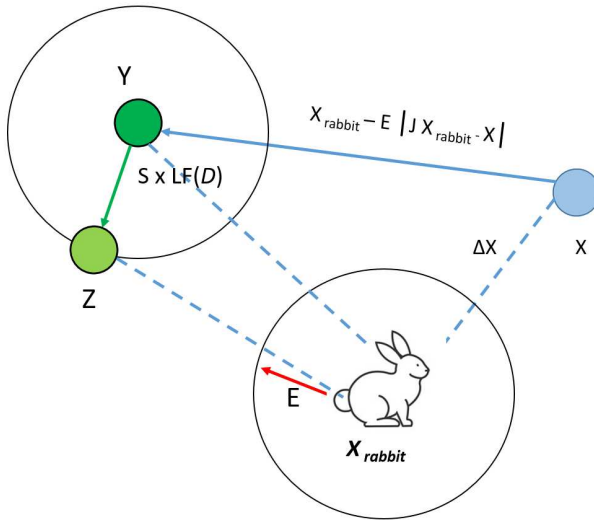


Fig. 3 The representation of vectors for soft besiege with progressive rapid dives

8 An island parallel Harris Hawks Optimization algorithm

If the rabbit does not have enough energy, a hard besiege can happen (see Figure 4). The following rule is applied;

$$X(t+1) = \begin{cases} Y & \text{if } F(Y) < F(X(t)) \\ Z & \text{if } F(Z) < F(X(t)) \end{cases} \quad (11)$$

$$Y = X_{rabbit}(t) - E |JX_{rabbit}(t) - X_m(t)| \quad (12)$$

$$Z = Y + S \times LF(D) \quad (13)$$

The pseudo code of the HHO algorithm is given in Algorithm 1.

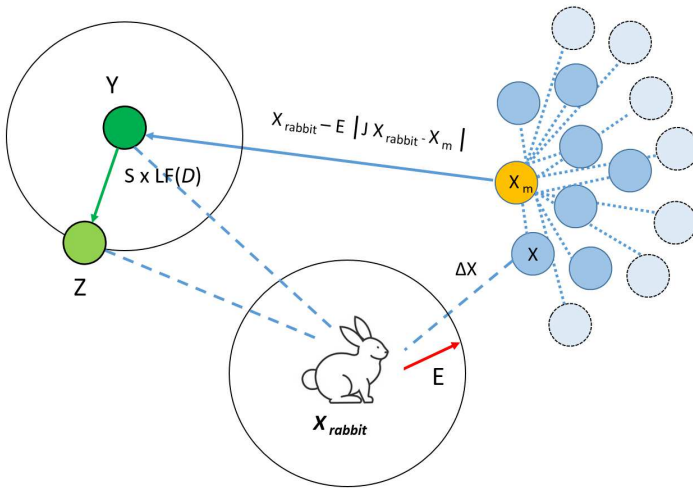


Fig. 4 The example of overall vectors in the case of hard besiege with progressive rapid dives in 2 dimensions

The IP-HHO method operates in a parallel computing environment using several swarms of Harris Hawks. The HHO principles are applied to a variety of processors with a variety of populations, parameter settings, and seedings. This procedure creates a very efficient and scalable environment. Diverse swarms of hawks can investigate and exploit the issue landscape without becoming trapped in the local optima. Between the processors, the IP-HHO algorithm employs a master and slave topology. Figure 5 shows the structure of the IP-HHO algorithm [41, 42]. At slave nodes, the swarm of hawks is located. The master node manages the communication of the slave nodes. Finally, the hawks' results are compiled, and the best one is chosen as the overall best solution of the optimization algorithm. This process significantly reduces the execution time required to arrive at the best solution. Figure 6 gives the flowchart of the IP-HHO algorithm.

Algorithm 1 The Pseudocode of the HHO algorithm [11]

```

1: //Input: The population size  $N$  and the number of iterations  $T$ 
2: //Output: The fitness value of the best hawk
3: Generate an initial population  $X_i(i = 1, 2, \dots, N)$ 
4: while ( $i++ < N$ ) do
5:   Find the fitness values of the hawks in the population
6:   Set  $X_{rabbit}$  as the rabbit
7:   for all hawk ( $X_i$ ) do
8:     Set the initial energy  $E_0$  and jump strength  $J$ 
9:     Update the value of  $E$ 
10:  end for
11:  if ( $|E|$  is more than 1) then                                ▷ Exploration step
12:    Update the location vector
13:  end if
14:  if ( $|E|$  is less than 1) then                                ▷ Exploitation step
15:    if ( $r \geq 0.5$  and  $|E| \geq 0.5$ ) then                        ▷ Perform Soft besiege
16:      Update the location
17:    else if ( $r \geq 0.5$  and  $|E| < 0.5$ ) then                    ▷ Perform Hard besiege
18:      Update the location
19:    else if ( $r < 0.5$  and  $E \geq 0.5$ ) then                        ▷ Soft besiege with
    progressive rapid dives
20:      Update the location
21:    else if ( $r < 0.5$  and  $E < 0.5$ ) then                        ▷ Hard besiege with
    progressive rapid dives
22:      Update the location
23:    end if
24:  end if
25: end while
26: return  $X_{rabbit}$ 

```

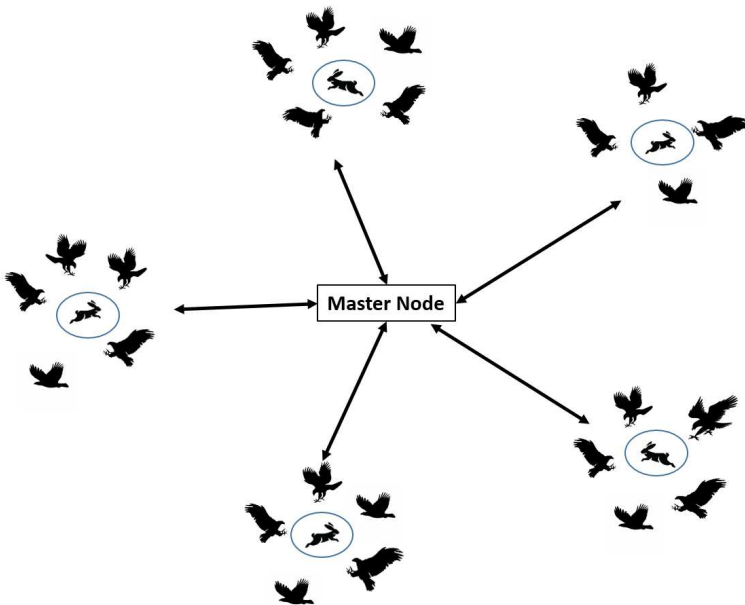


Fig. 5 The multiple swarm Harris hawks and the master node that controls the communication.

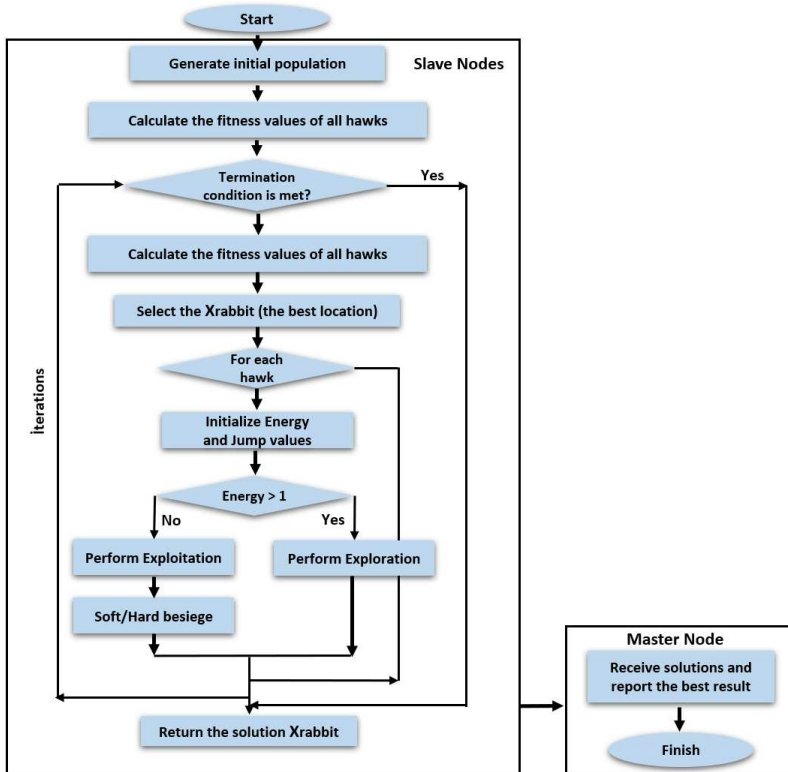


Fig. 6 The flowchart of the IP-HHO algorithm.

4 Performance Evaluation of the Experiments

We explain the details of our benchmark problem instances, the results obtained with the IP-HHO algorithm, the execution times, the comparison with state-of-the-art algorithms, the speedup and scalability of the IP-HHO algorithm in this section of our study.

4.1 Experimental setup and problem instances

The algorithms are implemented using C++ and MPI libraries. The C++ code of the HHO can be found on our website ¹. We use a high-performance AMD Opteron 6212 computer CPU, which runs at 2.6 GHz and has eight cores. Eight threads can be created on each core. The computer works with a 1.5 TB hard drive and 256 GB RAM.

The results of IP-HHO are compared with GA [2, 43], PSO [4, 43], DE [6, 43], Biogeography Based Optimization (BBO) [43], Cuckoo Search (CS) [44], Bat Algorithm (BA) [46], TLBO [45], Flower pollination algorithm (FPA) [47], Firefly Algorithm (FA) [48], Grey Wolf Optimizer (GWO) [49], Island Parallel GWO (IP-GWO), and Moth Flame Optimization (MFO) [50]. The statistical parameters, standard deviation (STD), the average value of the results (AVG) and best score (BEST) are compared in detail. To evaluate the metaheuristics, the Wilcoxon test with a 5% significance level is used. [51, 52]. For all algorithms, the population size and number of iterations are set to 30 individuals and 500 iterations, respectively. Table 1 lists the parameters used by metaheuristic algorithms. These are the best parameters that the metaheuristics use. During our comparison activities, all experiments are repeated 30 times.

A variety of benchmark functions are used to validate the performance of the IP-HHO algorithm [53, 54]. During the experiments, 13 different (most commonly tested) unimodal (UM) and multimodal (MM) functions are tested (for more information, see Tables 2 and 3 for details). These benchmark problems can be used to put the newly proposed algorithms' exploration and exploitation abilities to the test. Tables 4 and 5 provide information about the surface analysis of uni-modal and multi-modal functions, respectively, and the convergence performance of the classical HHO algorithm with 500 iterations. Premature convergence of the classical HHO is observed for functions f_5 , f_6 , f_8 , f_{12} , and f_{13} . The IP-HHO uses a multi-swarm approach to solve this problem and handles local optima.

Table 6 shows the results of the IP-HHO algorithm for 13 different benchmark functions (with 30, 100, 500 and 1000 dimensions and 64 processors).

Figure 7 compares the average performance of the HHO and IP-HHO algorithms with 30, 100, 500, and 1000 dimensions. Figure 8 shows the percentage improvements for the 30, 100, 500, and 1000 dimensions.

Tables 7, 8, 9, and 10 provide detailed information about the performance of the IP-HHO algorithm and the latest metaheuristics (see Figures 9, 10, 11,

¹<https://user.ceng.metu.edu.tr/e1451970>

Table 1 The settings of the parameter for the metaheuristic algorithms used in this study.

Algorithm	Parameter	Value
GA	crossover probability	0.5
	mutation probability	0.01
	selection method	roulette wheel
PSO	topology fully connected inertia factor	0.3
	c_1	1
	c_2	1
DE	the factor of scaling	0.5
	the probability of crossover	0.5
BBO	habitat modification probability	1
	immigration probability bounds per gene	[0, 1]
	step size for numerical integration of probabilities	1
	migration rates for each island	1
CS	discovery rate p_a	0.25
BA	Q_{min} frequency minimum	0
	Q_{max} frequency minimum	2
	A Loudness	0.5
	τ Pulse Rate	0.5
TLBO	the factor of teaching T	1,2
FPA	probability switch p	0.8
FA	α	0.5
	β	0.2
	γ	1
GWO	the constant of convergence a	[2 0]
MFO	convergence constant a	[-1 -2]
	the factor of spiral b	1

and 12 given in logscale for visual representations of the results). The IP-HHO algorithm can achieve the best eight solutions (out of 13 functions) for the 30-dimension problem, as shown in Table 7. For the other benchmark problems with 30 dimensions, the IP-HHO algorithm seems to achieve the second or third best solution.

In Table 8, the IP-HHO algorithm obtains the best results for 11 of the 13 functions with a 100-dimensional search space. In Table 9, the IP-HHO is better than all the other algorithms except the function 8 with 500 dimensions. The IP-HHO is superior when compared to metaheuristics in Table 10. After testing 52 functions with different dimensions, the IP-HHO algorithm is observed to be the best in 43 tests and the second/third best one in 11 metaheuristics. As the dimension increases, it is seen that the IP-HHO algorithm performs better than the others. When the figures of f1 and f4 functions in Table 4 are examined, we see that both functions are located in a similar parameter space with a single local minima. Although there is not much similarity between f12 and f13 in terms of surface shapes, the layout of local minima numbers shows similarity between these two functions.

The IP-HHO algorithm employs numerous exploration and exploitation techniques to efficiently avoid the local optima problem. This algorithm's early convergence is also a disadvantage. The multi-swarm optimization environment takes an efficient approach to calculations and allows for the exploration of a wide range of search areas. Exploration and exploitation patterns can be

Table 2 Details of the uni-modal benchmark functions.

No	Name	Function	Range	f_{min}
1	Sphere	$f_1(x) = \sum_{i=1}^n x_i^2$	[-5.12, 5.12]	0
2	Schwefel's 2.22	$f_2(x) = \sum_{i=0}^n x_i + \prod_{i=0}^n x_i $	[-10, 10]	0
3	Ridge	$f_3(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	[-64, 64]	0
4	Schwefel's 2.21	$f_4(x) = \max_i \{ x_i , 1 \leq i \leq n\}$	[-100, 100]	0
5	Rosenbrock	$f_5(x) = \sum_{i=1}^{n-1} [100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2]$	[-2.048, 2.048]	0
6	Fun6	$f_6(x) = \sum_{i=1}^n ([x_i + 0, 5])^2$	[-100, 100]	0
7	Quartic	$f_7(x) = \sum_{i=0}^n ix_i^4 + \text{random}[0, 1)$	[-1.28, 1.28]	0

explicitly improved by using different levels of energy. Using the escape energy, transitions between these two phases are possible. During optimization, the jump parameter J can be used to balance exploration and exploitation efforts. The probability of obtaining the best solutions increases significantly when many random values are tested in the parallel computation environment.

Table 3 Details of the multi-modal benchmark functions.

No	Name	Func.	Range	f_{min}
8	Schwefel	$f_8(x) = \sum_{i=1}^n (-x_i \sin(\sqrt{ x_i }))$	[-512, 512]	418.982887 · n
9	Rastrigin	$f_9(x) = 10n + \sum_{i=1}^n [x_i^2 - 10\cos(2\pi x_i)]$	[-5.12, 5.12]	0
10	Ackley	$f_{10}(x) = -20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}) - \exp(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)) + 20 + e$	[-32, 32]	0
11	Griewank	$f_{11}(x) = 1 + \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}})$	[-512, 512]	0
12	fl2	$f_{12}(x) = \frac{\Pi}{n} \left\{ 10 \cdot \sin(\Pi \cdot y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \cdot \sin^2(\Pi \cdot y_i + 1)] + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \cdot \sin^2(\Pi \cdot y_i + 1)] \right\}$ $y_i = 1 + \frac{x_i + 1}{4} \cdot u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	[-50, 50]	0
13	fl3	$f_{13}(x) = 0.1 \{ \sin^2(3\Pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \} + \sum_{i=1}^n u(x_i, 5, 100, 4)$	[-50, 50]	0

Table 4 Surface analysis of uni-modal functions and the convergence performance of the IP-HHO algorithm with 500 iterations.

fun.	Surface Diagram and Iterations	
Sphere (f1)		
Schwefel's 2.22 (f2)		
Ridge (f3)		
Schwefel's 2.21 (f4)		
Rosenbrock (f5)		
f6		
Quartic (f7)		

Table 5 Surface analysis of multi-modal functions and the convergence performance of the IP-HHO algorithm with 500 iterations.

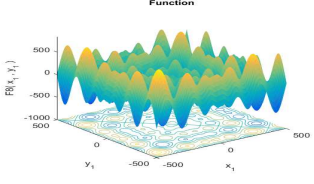
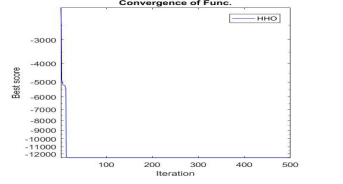
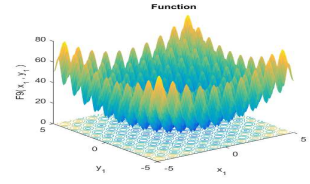
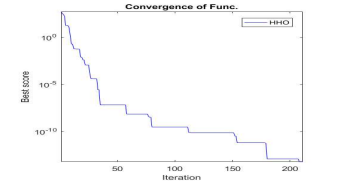
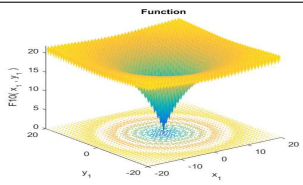
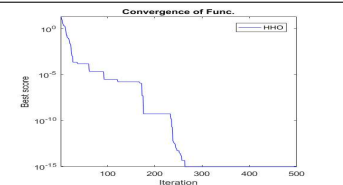
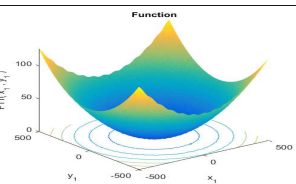
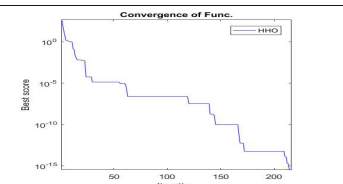
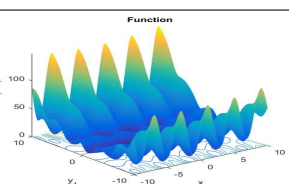
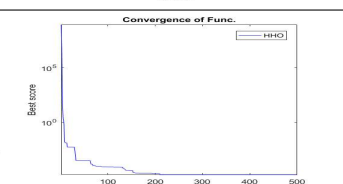
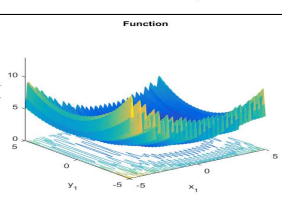
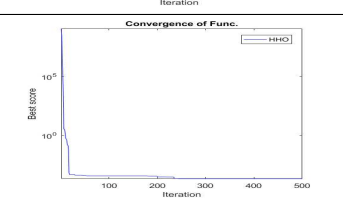
Function	Surface Diagram and Iterations	
Schwefel (f8)		
Rastrigin (f9)		
Ackley (f10)		
Griwank (f11)		
f12		
f13		

Table 6 The results of the IP-HHO with 30, 100, 500, and 1000 dimensions. 64 processors are used during the experiments. AVG is the overall average fitness values of the population, STD is the standard deviation and BEST is the best individual in the population.

Func.	Metric	30	100	500	1000
f1	AVG	1.82E-93	6.17E-90	6.38E-87	3.55E-86
	STD	2.86E-93	8.60E-90	7.93E-87	3.37E-86
	BEST	9.71E-99	3.42E-93	3.62E-92	5.34E-91
f2	AVG	1.51E-48	7.22E-47	5.88E-45	3.22E-42
	STD	1.35E-48	7.14E-47	5.30E-45	4.34E-42
	BEST	5.43E-50	5.53E-49	1.75E-47	7.74E-45
f3	AVG	1.99E-67	5.58E-53	4.29E-31	8.37E-21
	STD	3.31E-67	1.34E-52	9.03E-31	2.07E-20
	BEST	8.60E-74	6.72E-62	3.13E-42	2.77E-35
f4	AVG	4.53E-43	8.27E-42	2.22E-40	8.70E-40
	STD	1.82E-42	1.94E-41	2.79E-40	8.13E-40
	BEST	1.83E-46	8.49E-46	1.39E-43	2.02E-41
f5	AVG	2.65E+01	9.70E+01	4.94E+02	9.89E+02
	STD	1.65E-01	1.56E-01	1.14E-02	3.26E-02
	BEST	2.61E+01	9.63E+01	4.94E+02	9.89E+02
f6	AVG	2.25E-03	1.86E-01	3.64E+00	8.47E+00
	STD	3.52E-04	1.32E-02	1.41E-01	4.20E-01
	BEST	1.53E-03	1.57E-01	3.20E+00	7.21E+00
f7	AVG	1.86E-05	2.80E-05	3.08E-05	3.17E-05
	STD	1.03E-05	1.32E-05	1.74E-05	1.74E-05
	BEST	2.41E-06	4.36E-06	4.30E-06	9.24E-07
f8	AVG	-7.60E+03	-1.89E+04	-5.73E+04	-1.02E+05
	STD	3.38E+02	4.92E+02	2.26E+03	4.62E+03
	BEST	-8.65E+03	-2.02E+04	-6.20E+04	-1.16E+05
f9	AVG	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	STD	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	BEST	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f10	AVG	4.44E-16	4.44E-16	4.44E-16	4.44E-16
	STD	9.86E-32	9.86E-32	9.86E-32	9.86E-32
	BEST	4.44E-16	4.44E-16	4.44E-16	4.44E-16
f11	AVG	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	STD	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	BEST	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f12	AVG	5.01E-04	3.03E-03	7.43E-03	8.63E-03
	STD	8.78E-05	3.02E-04	3.75E-04	5.73E-04
	BEST	3.15E-04	2.36E-03	6.43E-03	7.09E-03
f13	AVG	1.77E-02	1.76E-01	1.32E+00	2.52E+00
	STD	4.34E-03	3.09E-02	1.91E-01	2.37E-01
	BEST	9.20E-03	1.11E-01	9.74E-01	1.78E+00

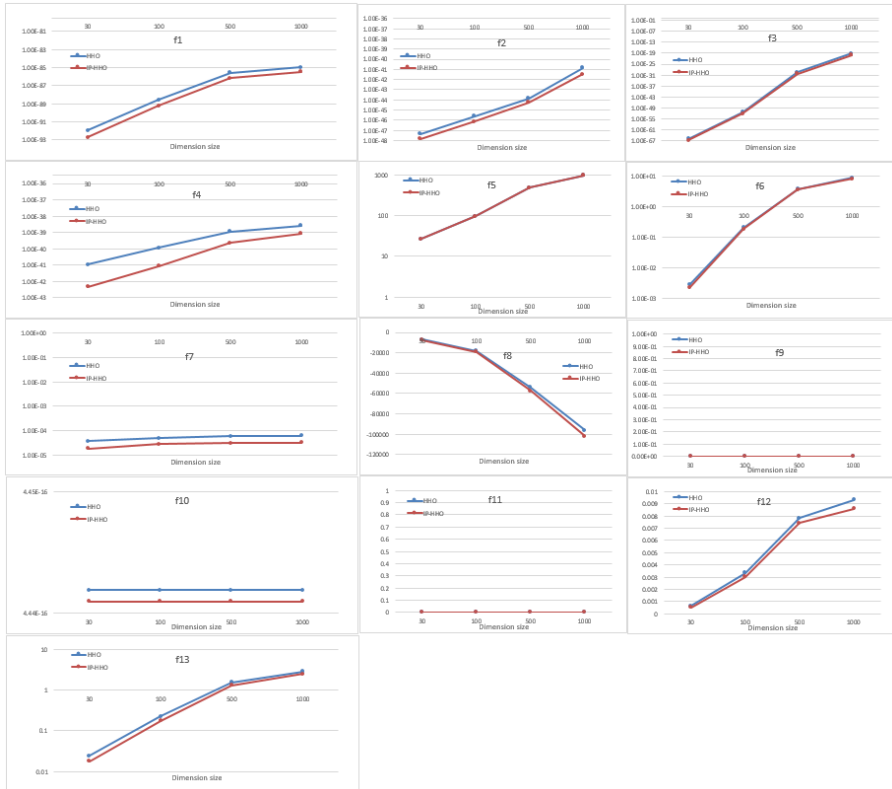


Fig. 7 The performance comparison of HHO and IP-HHO algorithms with 30, 100, 500, and 1000 dimensions. (function plots of f1, f2, f3, f4, f5, f6, f7, and f13 are given in logscale)

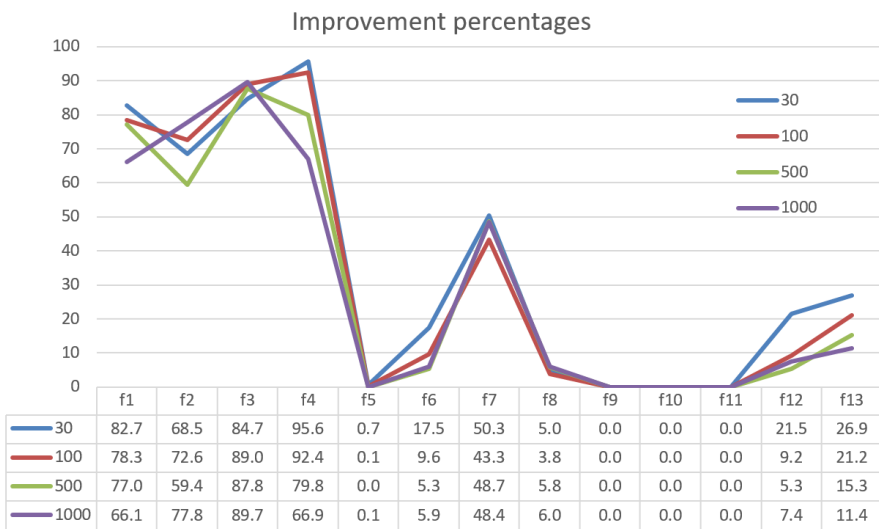


Fig. 8 The improvements of the IP-HHO algorithm against HHO with 30, 100, 500, and 1000 dimensions.

Table 7 Results of benchmark functions (f1-f13) with 30 dimensions. The best results are underlined.

func.	IP-HHO	GA	PSO	BBO	FPA	GWO	IP-GWO	BAT	FA	CS	MFO	TLBO	DE
f1	AVG	1.03E+03	1.83E+04	7.59E+01	2.01E+03	1.18E-27	9.10E-28	6.59E+04	7.11E-03	9.06E-04	1.01E+03	2.17E-89	1.33E-03
	STD	5.79E+02	3.01E+03	2.75E+01	5.60E+02	1.47E-27	1.41E-27	7.51E+03	3.21E-03	4.55E-04	3.05E+03	3.14E-89	5.92E-04
f2	AVG	1.51E-48	2.47E+01	3.58E+02	1.36E+03	9.71E-17	8.92E-17	2.71E+08	4.34E-01	1.49E-01	3.19E+01	2.77E-45	6.83E-03
	STD	1.35E-48	1.35E+00	7.45E-03	5.55E+00	5.60E-17	4.58E-17	1.30E+09	1.84E-01	2.79E-02	2.06E+01	3.11E-45	2.06E-03
f3	AVG	1.99E-67	2.65E+04	4.05E+04	1.21E+04	5.12E-05	1.41E-06	1.38E+05	1.66E+03	2.10E-01	2.43E+04	3.91E-18	3.97E-04
	STD	3.31E-67	3.44E+03	8.21E+03	2.69E+03	2.03E-04	1.90E-04	4.72E+04	6.72E+02	5.69E-02	1.41E+04	8.04E-18	5.37E+03
f4	AVG	4.53E-43	5.17E+01	4.39E+01	3.02E+01	2.38E+01	1.01E-06	8.51E+01	1.11E-01	9.65E-02	7.00E+01	1.68E-36	1.15E+01
	STD	1.82E-42	1.05E+01	3.64E+00	4.39E+00	2.77E+00	1.94E-06	2.95E+00	4.75E-02	1.94E-02	7.06E+00	1.47E-36	2.37E+00
f5	AVG	2.65E+01	1.95E+04	1.96E+07	1.82E+03	3.17E+05	2.30E+01	7.97E+01	2.76E+01	7.35E+03	2.54E+01	1.06E+02	
	STD	1.65E-01	1.31E+04	6.25E+06	9.40E+02	1.75E+05	7.78E-01	4.17E+07	7.39E+01	4.51E-01	2.26E+04	4.26E-01	1.01E+02
f6	AVG	2.25E-03	9.01E+02	1.87E+04	6.71E+01	1.70E+03	8.44E-01	6.69E+04	6.94E-03	3.13E-03	2.68E+03	3.29E-05	1.44E-03
	STD	3.52E-04	2.84E+02	2.92E+03	2.20E+01	3.13E+02	3.18E-01	5.87E+03	3.61E-03	1.30E-03	5.84E+03	8.65E-05	5.38E-04
f7	AVG	1.86E-05	1.91E-01	1.07E+01	2.91E-03	3.41E-01	1.70E-03	4.57E+01	6.62E-02	7.29E-02	4.50E+00	1.16E-03	5.24E-02
	STD	1.03E-05	1.50E-01	3.05E+00	1.83E-03	1.10E-01	1.06E-03	7.82E+00	4.23E-02	2.21E-02	9.21E+00	3.63E-04	1.37E-02
f8	AVG	-7.60E-03	-1.26E+04	-3.86E+03	-1.24E+04	-6.45E+03	3.17E-03	-2.33E+03	-5.85E+03	-5.19E+19	-8.48E+03	-7.76E+03	-6.82E+03
	STD	3.38E+02	4.51E+00	2.49E+02	3.50E+01	3.03E+02	2.10E+02	7.00E-02	2.96E+02	1.16E+03	7.98E+02	1.04E+03	3.94E+02
f9	AVG	0.00E+00	9.04E+00	2.87E+02	0.00E+00	1.82E+02	2.11E+00	1.92E+02	3.82E+01	1.51E+01	1.59E+02	1.40E+01	1.58E+02
	STD	0.00E+00	4.58E+00	1.95E+00	0.00E+00	1.24E+01	3.69E+00	3.52E+00	3.56E+01	1.12E+01	1.25E+00	3.21E+01	5.45E+00
f10	AVG	4.44E-16	1.36E+01	1.75E+01	2.13E+00	7.14E+00	1.03E-13	9.90E-14	1.92E+01	4.38E-02	1.74E+01	6.45E-15	1.21E-02
	STD	9.89E-32	1.51E+00	3.67E-01	3.53E-01	1.08E+00	1.70E-14	1.62E-14	2.43E-01	1.20E-02	7.93E-03	4.95E+00	1.79E-15
f11	AVG	0.00E+00	1.01E+01	1.70E+02	1.46E+00	1.73E+01	3.96E-03	6.01E+02	4.23E-03	4.29E-05	3.10E+01	0.00E+00	3.52E-02
	STD	0.00E+00	2.43E+00	3.17E+01	1.69E-01	3.63E+00	8.57E-03	5.50E+01	1.29E-03	2.00E-05	5.94E+01	0.00E+00	7.20E-02
f12	AVG	5.01E-04	4.77E+00	1.51E+07	6.68E+01	3.05E+02	3.55E-02	3.13E-04	5.57E-05	2.46E+02	7.35E-06	2.25E-03	2.45E-06
	STD	8.78E-05	1.56E+00	9.88E+06	2.62E-01	1.04E+03	2.12E-02	1.54E+08	1.76E-04	4.96E-05	1.21E+07	7.45E-06	1.70E-03
f13	AVG	1.77E-02	1.52E+01	5.73E+01	1.82E+00	9.59E+04	3.96E-01	9.40E+08	2.08E-03	8.19E-03	2.73E+07	7.89E-02	9.12E-03
	STD	1.77E-02	4.52E+00	2.68E+07	3.41E-01	1.46E+05	2.23E-01	1.67E+08	9.62E-04	6.74E-03	1.04E+08	8.78E-02	1.16E-02

Table 8 Results of benchmark functions (f1-f13) with 100 dimensions. The best results are underlined>

func.	IP-HHO	GA	PSO	BBO	FPA	GWO	IP-GWA	BAT	FA	CS	MFO	TLBO	DE
f1	AVG	5.41E+04	1.06E+04	2.85E+03	1.39E+04	1.59E-12	1.40E-12	2.72E+05	3.05E-01	3.17E-01	6.20E+04	3.62E-81	8.26E+03
	STD	8.60E-90	1.42E+04	4.49E+03	2.71E+03	1.63E-12	1.52E-12	1.42E+04	5.60E-02	5.28E-02	1.25E+04	4.14E-81	1.32E+03
f2	AVG	7.22E-47	2.53E+02	6.06E+23	1.59E+01	1.01E+02	4.31E-08	2.54E-08	6.00E+43	1.45E+01	4.05E+00	2.46E+02	1.21E+02
	STD	7.14E-47	1.18E+01	2.18E+01	3.74E+00	1.46E-08	1.40E-08	1.18E+44	6.73E+00	3.16E-01	4.48E+01	2.75E-41	2.33E+01
f3	AVG	5.58E-53	2.53E+05	4.22E+05	1.70E+05	1.89E+04	4.09E+02	3.54E+02	1.43E+06	4.65E+04	6.88E+00	2.15E+05	5.01E+05
	STD	1.34E-52	5.03E+04	7.08E+04	2.02E+04	5.44E+03	2.77E+02	7.22E+02	6.21E+05	6.92E+03	1.02E+00	4.43E+04	5.87E+04
f4	AVG	8.27E-42	8.19E+01	6.07E+01	7.08E+01	3.37E+01	8.89E-01	9.41E+01	1.91E+01	2.58E-01	9.31E+01	1.60E-33	9.62E+01
	STD	1.94E-41	3.15E+00	3.05E+00	4.73E+00	3.37E+00	9.30E-01	9.24E-01	1.49E+00	3.12E+00	2.80E-02	2.13E+00	6.66E-33
f5	AVG	9.70E+01	2.37E+07	2.42E+08	4.47E+05	4.64E+06	9.79E+01	1.10E+09	8.46E+02	1.33E+02	1.44E+08	9.67E+01	1.99E+07
	STD	1.56E-01	8.43E+06	4.02E+07	2.05E+05	1.98E+06	6.75E-01	6.14E-01	9.47E+07	8.13E+02	7.34E+00	7.77E-01	5.80E+06
f6	AVG	1.86E-01	5.42E+04	1.07E+05	2.85E+03	1.26E+04	1.03E+01	2.69E+00	2.95E-01	2.65E+00	6.68E+04	3.27E+00	8.07E+03
	STD	1.32E-02	1.09E+04	9.70E+03	4.07E+02	2.06E+03	1.05E+00	1.02E+00	1.25E+04	5.34E-02	3.94E-01	1.46E+04	1.64E+03
f7	AVG	2.80E-05	2.73E+01	3.41E+02	1.25E+00	5.84E+00	7.60E-03	5.80E-03	3.01E+02	5.65E-01	1.21E+00	2.56E-02	1.96E+01
	STD	1.32E-05	4.45E+01	8.74E+01	5.18E+00	2.16E+00	2.66E-03	2.42E-03	2.66E+01	1.64E-01	2.65E-01	5.39E-04	5.66E+00
f8	AVG	-1.89E-04	-4.10E+04	-7.33E+03	-3.85E+04	-1.28E+04	-1.42E+04	4.07E+03	-1.81E+04	-2.84E+18	-2.30E+04	-1.71E+04	-1.19E+04
	STD	4.92E+02	1.14E+02	4.75E+02	2.80E+02	4.64E+02	2.62E+03	2.42E-03	9.37E+02	3.23E+03	6.91E+18	1.98E+03	5.80E+02
f9	AVG	0.00E+00	3.39E+02	1.16E+03	9.11E+00	8.47E+02	1.03E+01	9.10E+00	7.97E+02	2.36E+02	8.65E+02	1.02E+01	1.03E+03
	STD	0.00E+00	4.17E+01	5.74E+01	2.73E+00	4.01E+01	9.02E+00	9.44E+00	6.33E+01	2.63E+01	8.01E+01	5.57E+01	4.03E+01
f10	AVG	4.44E-16	1.82E+01	1.91E+01	5.57E+00	8.21E+00	1.20E-07	9.00E-08	1.94E+01	9.81E-01	1.99E+01	1.66E-02	1.22E+01
	STD	9.89E-32	4.35E-01	2.04E-01	4.72E-01	1.14E+00	5.07E-08	4.07E-08	6.50E-02	5.25E-01	5.23E-02	8.58E-02	8.31E-01
f11	AVG	0.00E+00	5.14E+02	9.49E+02	2.24E+01	1.19E+02	4.87E-03	3.87E-03	2.47E+03	1.19E-01	5.60E+02	0.00E+00	7.42E+01
	STD	0.00E+00	1.05E+02	6.00E+01	4.35E+00	2.00E+01	1.07E-02	2.07E-02	1.03E+02	2.34E-02	1.23E+02	0.00E+00	1.40E+01
f12	AVG	3.03E-03	4.55E+06	3.54E+08	3.03E+02	1.55E+05	2.87E-01	2.17E-01	2.64E+09	4.45E+00	2.82E-08	3.03E-02	3.90E+07
	STD	3.02E-04	8.22E+06	8.75E+07	1.48E+01	1.74E+05	6.41E-02	4.12E-02	2.69E+08	1.32E+00	1.45E+08	1.02E-02	1.88E+07
f13	AVG	1.76E-01	5.26E+07	8.56E+07	6.82E+04	2.76E+06	5.56E+00	5.01E+00	4.50E+01	5.84E+00	6.68E+08	4.47E+00	7.19E+07
	STD	3.09E-02	3.76E+07	2.16E+08	3.64E+04	1.80E+06	3.32E-01	2.22E-01	3.93E+08	2.24E+01	3.05E+08	8.34E-01	2.73E+07

Table 9 Results of benchmark functions (f1-f13) with 500 dimensions. The best results are underlined.

func.	IP-HHO	GA	PSO	BBO	FPA	GWO	IP-GWO	BAT	EA	CS	MFO	TLBO	DE
f1	AVG	6.06E+05	6.42E+05	1.60E+05	8.26E+04	1.42E-03	1.32E-03	1.52E+06	6.30E+04	6.80E+00	1.15E+06	2.14E-77	7.43E+05
	STD	7.01E+04	2.96E+04	9.76E+03	1.32E+04	3.99E-04	1.28E-03	3.58E+04	8.47E+03	4.93E-01	3.54E+04	1.94E-77	3.67E+04
f2	AVG	5.88E-45	1.94E+03	6.08E+09	5.95E+02	5.13E-02	1.10E-02	8.34E+09	7.13E+09	4.57E+01	3.00E+08	2.31E-39	3.57E+09
	STD	5.30E-45	7.03E+01	1.70E+01	1.70E+01	1.93E-03	1.09E-02	1.70E+01	3.76E+01	2.05E+00	1.58E+09	1.63E-39	1.70E+10
f3	AVG	4.29E-31	5.79E+06	1.13E+07	2.98E+06	3.34E+05	3.18E+05	3.37E+07	1.19E+06	2.03E+02	4.90E+06	1.63E+07	1.20E+07
	STD	9.03E-31	9.08E+05	1.43E+06	3.87E+05	1.34E+05	7.95E+04	1.41E+07	1.88E+05	2.72E+01	1.02E+06	3.70E+00	1.49E+06
f4	AVG	2.22E-40	9.59E+01	8.18E+01	9.35E+01	4.52E+01	6.51E+01	9.82E+01	5.00E+01	4.06E-01	9.88E+01	4.02E-31	9.92E+01
	STD	2.79E-40	1.20E+00	1.49E+00	9.05E-01	4.28E+00	5.72E+00	3.32E-01	1.73E+00	3.03E-02	4.15E-01	2.67E-31	2.33E-01
f5	AVG	4.94E+02	1.79E+09	1.84E+09	2.07E+08	3.30E+07	4.98E+02	6.94E+09	2.56E+07	1.21E+03	5.01E+09	4.97E+02	4.57E+09
	STD	1.14E-02	4.11E+08	1.11E+08	2.08E+07	8.76E+06	5.23E-01	4.28E+02	2.23E+08	6.14E+06	7.04E+01	3.07E-01	1.25E+09
f6	AVG	3.64E+00	6.27E+05	6.57E+05	1.68E+05	8.01E+04	9.22E+01	1.53E+06	6.30E+04	8.27E+01	1.16E+06	7.82E+01	7.23E+05
	STD	1.41E-01	7.43E+04	3.29E+04	8.23E+03	9.32E+03	2.15E+00	8.78E+01	3.37E+04	8.91E+03	3.48E+04	2.50E+00	3.28E+04
f7	AVG	3.08E-05	9.10E+03	1.43E+04	2.62E+03	2.53E+02	4.67E-02	3.98E-02	2.23E+04	3.71E+02	3.84E+04	1.71E-03	2.39E+04
	STD	1.74E-05	2.20E+03	1.51E+03	3.59E+02	6.28E+01	1.12E-02	3.60E-02	1.15E+03	6.74E+01	1.37E+01	4.80E-04	2.72E+03
f8	AVG	-5.73E+04	-1.31E+05	-1.65E+04	-1.42E+05	-3.00E+04	-5.70E+04	-9.03E+03	-7.27E+04	-2.10E+17	-6.29E+04	-5.02E+04	-2.67E+04
	STD	2.26E+03	2.31E+04	9.99E+02	1.98E+03	1.14E+03	3.12E+03	5.40E+04	2.12E+03	1.15E+04	1.14E+18	1.00E+04	1.38E+03
f9	AVG	0.00E+00	3.29E+03	6.63E+03	7.86E+02	4.96E+03	7.84E+01	6.18E+03	2.80E+03	2.54E+03	6.96E+03	0.00E+00	7.14E+03
	STD	0.00E+00	1.96E+02	1.07E+02	3.42E+01	7.64E+01	1.13E+01	7.20E+01	1.20E+02	1.42E+02	1.48E+02	0.00E+00	1.05E+02
f10	AVG	4.44E-16	1.96E+01	1.97E+01	1.44E+01	8.55E+00	1.93E-03	2.04E+01	1.24E+01	1.07E+00	2.03E+01	7.62E-01	2.06E+01
	STD	9.80E-32	2.04E-01	1.04E-01	2.22E-01	8.66E-01	3.50E-04	1.45E-02	3.25E-02	4.46E-01	6.01E-02	1.48E-01	2.45E-01
f11	AVG	0.00E+00	5.42E+03	3.19E+02	8.10E+01	1.55E-02	1.73E-02	1.38E+04	5.83E+02	2.66E-02	1.03E+04	0.00E+00	6.75E+03
	STD	0.00E+00	7.32E+02	3.19E+02	8.10E+01	8.17E+01	3.50E-02	7.33E-01	2.30E-03	4.43E+02	0.00E+00	2.97E+02	
f12	AVG	3.75E-04	1.11E+09	3.51E+09	1.60E+08	4.50E+06	7.42E-01	1.70E+10	8.67E+05	3.87E-01	1.20E+10	4.61E-01	1.60E+10
	STD	3.32E+00	8.84E+09	4.16E+08	3.16E+07	3.37E+06	4.38E-02	7.00E-01	6.29E+05	2.47E-02	6.82E+08	2.40E-02	2.34E+09
f13	AVG	1.32E+00	1.11E+09	6.82E+09	5.13E+08	3.94E+07	5.06E+01	3.17E+10	6.23E+07	6.00E+01	2.23E+10	4.98E+01	2.42E+10
	STD	1.91E-01	2.00E+09	8.45E+08	6.59E+07	1.87E+07	1.30E+00	4.84E+0	9.68E+08	9.46E+06	1.13E+00	9.97E-03	6.39E+09

Table 10 Results of benchmark functions (f1-f13) with 1000 dimensions. The best results are underlined.

func.	IP-HHO	GA	PSO	BBO	FPA	GWO	IP-GWO	BAT	EA	CS	MFO	TLBO	DE
f1	AVG	<u>3.55E-86</u>	1.36E+06	6.51E+05	1.70E+05	2.42E-01	2.12E-01	3.12E+06	3.20E+05	1.65E+01	2.73E+06	2.73E-76	2.16E+06
	STD	3.37E-86	6.33E+04	2.37E+04	2.99E+04	4.72E-02	2.10E-01	4.61E+04	2.11E+04	1.27E+00	4.70E+04	7.67E-76	3.39E+05
f2	AVG	<u>3.22E-42</u>	4.29E+03	1.96E+03	8.34E+02	7.11E-01	6.88E-01	1.79E+10	1.79E+10	1.02E+02	1.79E+10	1.79E+10	1.79E+10
	STD	4.34E-42	8.86E+01	1.79E+10	2.18E+10	4.96E-01	6.80E-01	1.79E+10	1.79E+10	3.49E+00	1.79E+10	1.79E+10	1.79E+10
f3	AVG	<u>8.37E-21</u>	2.99E+07	3.72E+07	1.95E+06	1.95E+06	1.22E+06	1.35E+08	4.95E+06	8.67E+02	1.94E+07	8.61E-01	5.03E-07
	STD	2.07E-20	3.93E+06	1.16E+07	1.48E+06	2.43E+05	1.18E+06	4.76E+07	7.19E+05	1.10E+02	3.69E+06	1.33E+00	4.14E+06
f4	AVG	<u>8.70E-40</u>	9.79E+01	8.92E+01	5.03E+01	7.94E+01	7.62E+01	9.89E+01	6.06E+01	4.44E-01	9.96E+01	1.01E-30	9.95E+01
	STD	8.13E-40	7.16E-01	2.39E+00	7.62E-01	2.77E+00	7.51E+01	2.22E-01	2.69E+00	2.24E-02	1.49E-01	5.25E-31	1.43E-01
f5	AVG	<u>9.89E+02</u>	4.73E+09	3.72E+09	1.29E+09	7.27E+07	1.05E+03	1.45E+10	2.47E+08	2.68E+03	1.25E+10	9.97E+02	1.49E+10
	STD	3.26E-02	9.63E+08	2.76E+08	6.36E+07	1.84E+07	1.02E+03	3.20E+08	3.24E+07	1.27E+02	3.15E+08	2.01E-01	3.06E+08
f6	AVG	<u>8.47E+00</u>	1.52E+06	1.38E+06	6.31E+05	1.60E+05	1.95E+02	3.11E+06	3.18E+05	2.07E+02	2.73E+06	1.93E+02	2.04E+06
	STD	4.20E-01	1.88E+05	6.05E+04	1.82E+04	1.86E+04	1.05E+02	6.29E+04	2.47E+04	4.12E+00	4.56E+04	2.35E+00	2.46E+05
f7	AVG	<u>3.17E-05</u>	4.45E+04	6.26E+04	3.84E+04	1.09E+03	1.47E-01	1.25E+05	4.44E+03	4.10E+02	1.96E+05	1.83E-03	2.27E+05
	STD	1.74E-05	8.40E+03	4.16E+03	2.91E+03	3.49E+02	3.28E-02	1.28E-01	3.93E+03	4.00E+02	6.19E+03	5.79E-04	3.52E+04
f8	AVG	<u>4.62E+03</u>	-1.94E+05	-2.30E+04	-2.29E+05	-4.25E+04	-8.51E+04	-1.48E+04	-1.08E+05	-9.34E+14	-9.00E+04	-6.44E+04	-3.72E+04
	STD	-1.02E+05	9.74E+03	1.70E+03	3.76E+03	1.47E+03	1.91E+04	1.51E+04	3.14E+03	1.69E+04	2.12E+15	7.20E+03	1.23E+03
f9	AVG	<u>0.00E+00</u>	8.02E+03	1.35E+04	2.86E+03	1.01E+04	2.06E+02	1.92E+02	1.40E+04	7.17E+03	6.05E+03	1.56E+04	1.50E+04
	STD	0.00E+00	3.01E+02	1.83E+02	9.03E+01	1.57E+02	4.81E+01	1.90E+02	1.85E+02	1.88E+02	1.41E+02	1.94E+02	0.00E+00
f10	AVG	<u>4.44E-16</u>	1.95E+01	1.98E+01	1.67E+01	8.62E+00	1.88E-02	1.63E-02	2.07E+01	1.55E+01	2.04E+01	5.09E-01	2.07E+01
	STD	9.80E-32	2.55E-01	1.24E-01	8.63E-02	9.10E-01	2.74E-03	1.61E-02	2.23E-02	2.42E-01	2.16E-01	1.94E+00	1.06E-01
f11	AVG	<u>0.00E+00</u>	1.26E+04	1.23E+04	5.75E+03	1.52E+03	6.58E-02	2.83E+04	2.87E+03	3.92E-02	2.47E+04	1.07E-16	1.85E+04
	STD	0.00E+00	1.63E+03	5.18E+02	1.78E+02	2.66E+02	8.82E-02	6.18E-02	4.21E+02	1.78E+02	4.51E+02	2.03E-17	2.22E+03
f12	AVG	<u>8.63E-03</u>	1.14E+10	7.73E+09	1.56E+09	8.11E+06	1.10E+00	3.63E+10	6.76E+07	6.53E-01	3.04E+10	6.94E-01	3.72E+10
	STD	5.73E-04	1.27E+09	6.72E+08	1.46E+08	3.46E+06	1.82E-01	1.11E+09	1.80E+07	2.45E-02	9.72E+08	1.90E-02	7.67E+08
f13	AVG	<u>2.52E+00</u>	1.91E+10	1.58E+10	4.17E+09	8.96E-07	1.21E+02	6.61E+02	4.42E+08	1.32E+02	5.62E+10	9.98E+01	6.66E+10
	STD	2.37E-01	4.21E+09	1.56E+09	2.54E+09	3.65E+07	1.11E+01	1.08E+02	1.40E+09	7.91E+07	1.48E+00	1.31E-02	2.26E+09

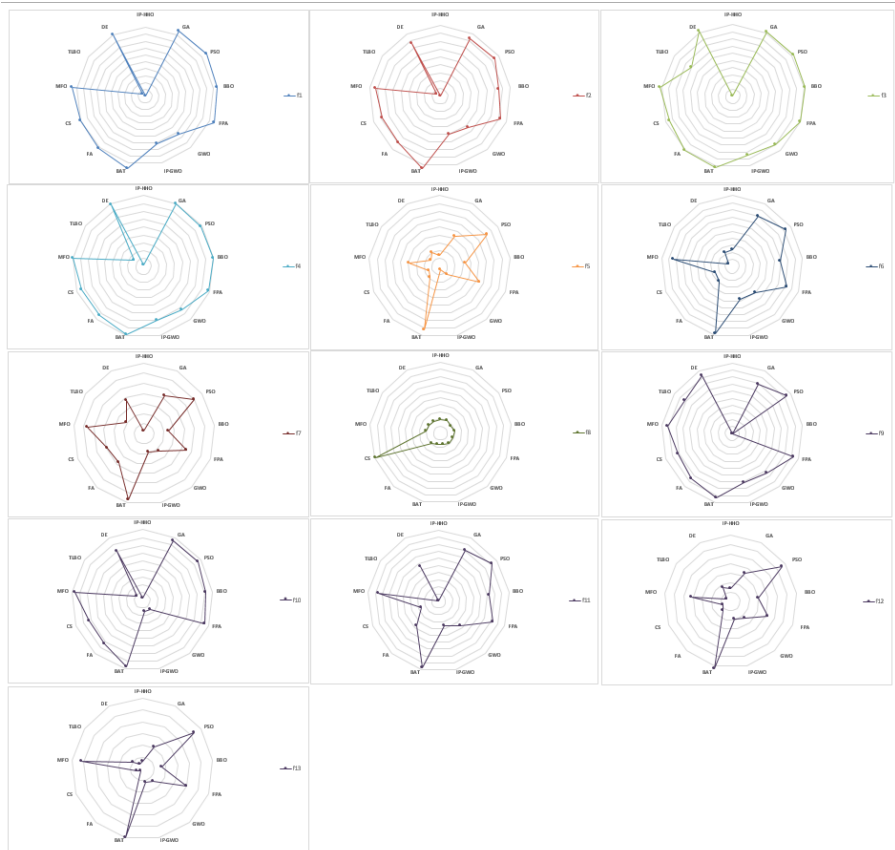


Fig. 9 Logscale visualization of results in Table 7 for 30 dimensions (The results closer to the center of the plot have better performance).

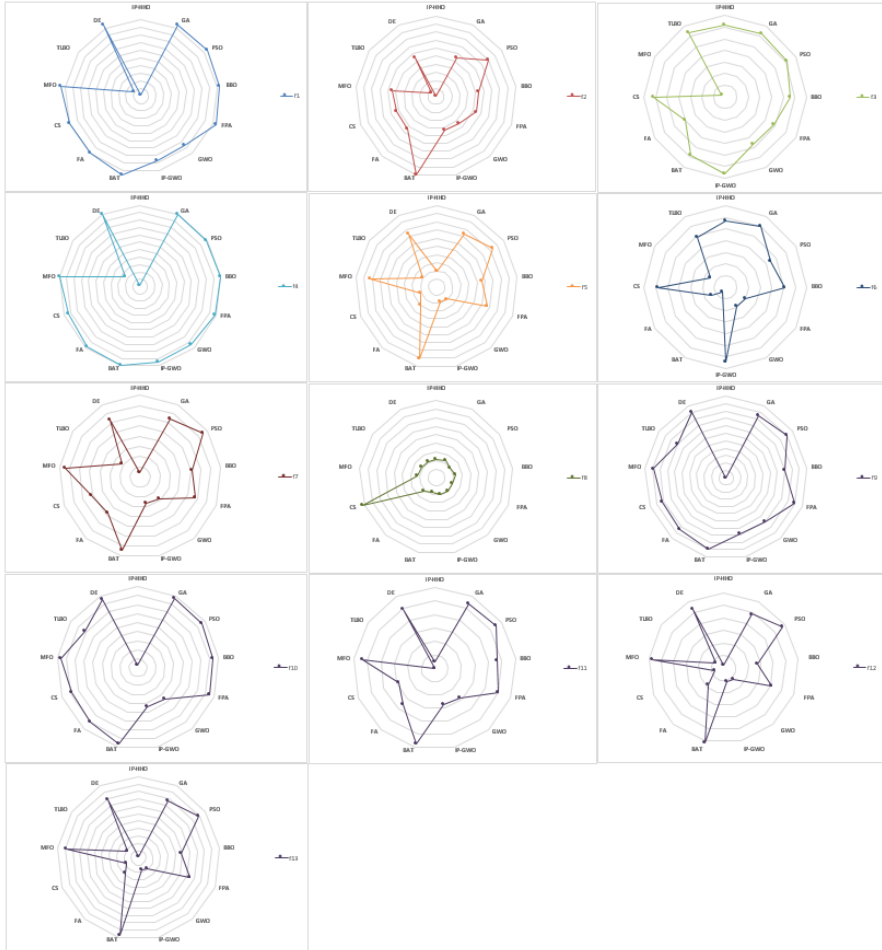


Fig. 10 Logscale visualization of results in Table 8 for 100 dimensions (The results closer to the center of the plot have better performance).

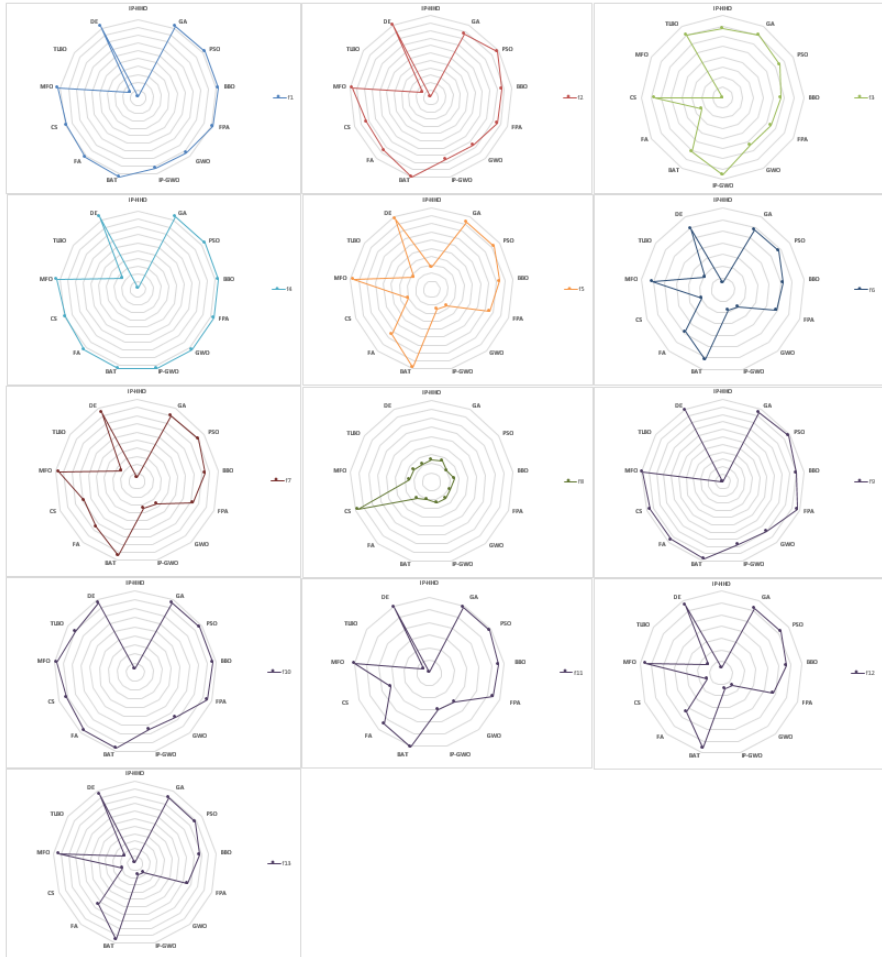


Fig. 11 Logscale visualization of results in Table 9 for 500 dimensions (The results closer to the center of the plot have better performance).

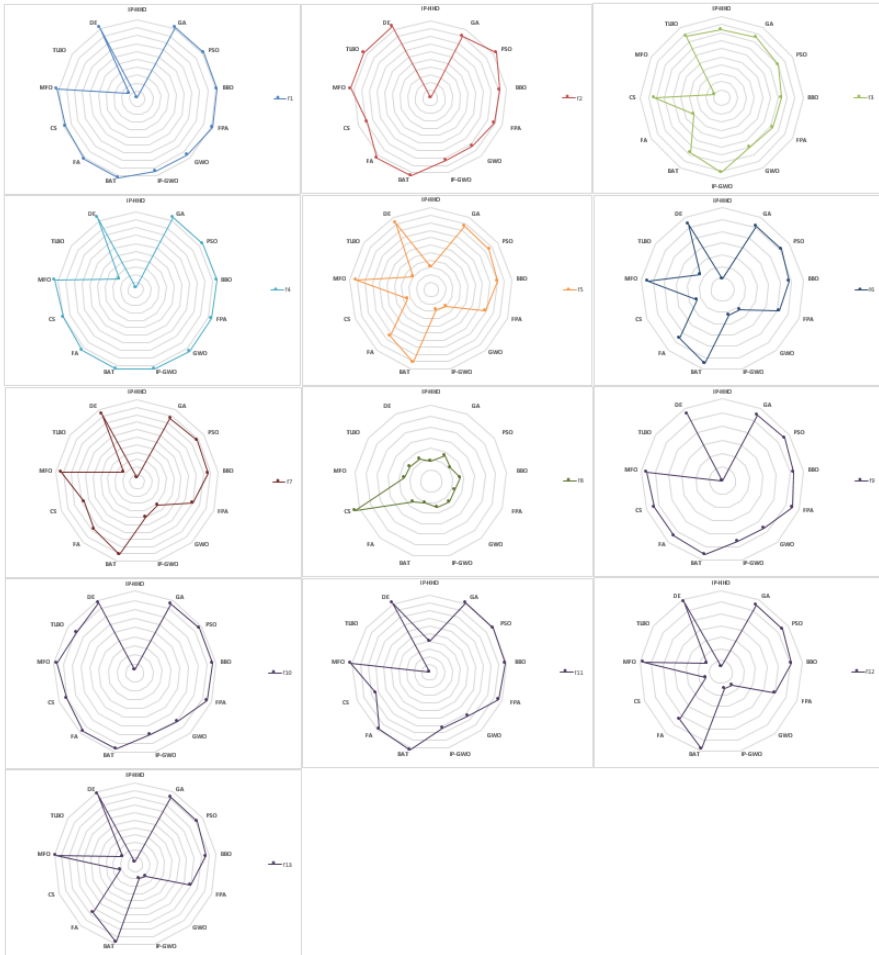


Fig. 12 Logscale visualization of results in Table 10 for 1000 dimensions (The results closer to the center of the plot have better performance).

Table 14 The Wilcoxon rank-sum test p -values with 5% significance for 1000 dimensions (results larger than 0.05 are underlined).

func.	GA	PSO	BBO	FPA	GWO	IP-GWO	BAT	FA	CS	MFO	TLBO	DE
f1	2.74E-10	2.54E-10	2.52E-11	2.74E-10	2.74E-10	2.72E-10	2.74E-10	2.74E-10	2.74E-10	2.74E-10	2.74E-10	2.74E-10
f2	2.54E-10	1.21E-12	2.72E-11	2.74E-10	2.74E-10	2.72E-10	2.54E-10	2.54E-10	2.74E-10	2.41E-10	1.21E-12	1.21E-12
f3	1.86E-11	1.86E-11	1.86E-11	2.02E-11	2.02E-11	2.00E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11
f4	1.93E-10	1.93E-10	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09
f5	2.70E-11	1.52E-11	1.52E-11	2.02E-11	2.02E-11	2.00E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11	2.02E-11
f6	2.21E-10	2.63E-10	2.63E-10	3.00E-09	3.00E-09	2.98E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09
f7	2.64E-10	2.52E-10	2.52E-10	2.41E-10	2.41E-10	2.40E-10	2.41E-10	2.41E-10	2.41E-10	2.41E-10	2.41E-10	2.41E-10
f8	2.21E-10	2.63E-10	2.52E-11	3.00E-09	3.00E-09	2.98E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09	3.00E-09
f9	1.01E-12	1.06E-12	9.57E-13	1.21E-12	1.21E-12	1.18E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	NaN	1.21E-12
f10	1.01E-10	1.01E-10	9.57E-11	1.42E-10	1.42E-10	1.40E-10	1.42E-10	1.42E-10	1.42E-10	1.42E-10	8.72E-14	1.42E-10
f11	1.06E-12	1.01E-12	9.57E-13	1.21E-12	1.21E-12	1.18E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.17E-13	1.21E-12
f12	3.21E-09	2.11E-09	2.23E-09	3.02E-09	3.02E-09	2.98E-09	3.02E-09	3.02E-09	3.02E-09	3.02E-09	3.02E-09	3.02E-09
f13	1.44E-09	2.21E-09	2.72E-09	3.02E-10	3.02E-10	2.98E-10	3.02E-10	3.02E-10	3.02E-10	3.02E-10	3.02E-10	3.02E-10

4.3 Execution times of the IP-HHO algorithm

In Table 15, the execution times of the algorithms are presented. The results of the IP-HHO algorithm are obtained with 64 processors. Considering the algorithm's computational complexity, the leading factors of the IP-HHO are the number of trials and the computation time of the functions. It is seen that the function $f3$ needs more time during the calculations compared to the others [11]. Since it is an $O(n^2)$ complexity function, this difference appears clearly when the dimension number is 1000. The same situation is observed in other metaheuristic algorithms. The IP-HHO algorithm shows a fast performance in obtaining the solutions with high dimensional functions.

4.4 The performance of the IP-HHO algorithm with increasing number of processors

In this section of our experiments, we report on the performance of the IP-HHO algorithm as the number of processors increases. First, we begin with two processors and demonstrate how the algorithm's performance changes with 4, 8, 16, 32, and 64 processors for the f1, f5, and f9 functions. Figures 13, 14, and 15 show the results of the functions as the number of processors increases. As can be seen, as the number of processors increases from 2 to 64, the results appear to improve. Furthermore, the proposed IP-HHO algorithm employs a master-slave communication style, with slave nodes sending the best solutions to the master node when the results of their local optimization are available. This procedure also significantly reduces the need for communication.

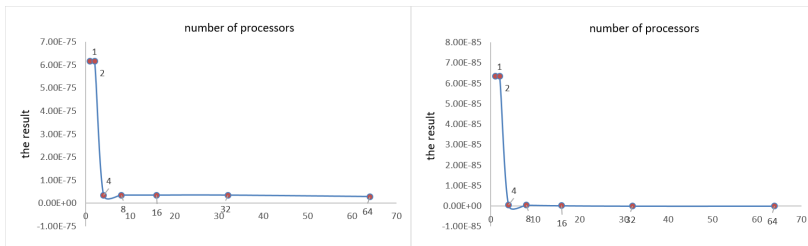
**Fig. 13** The performance of the IP-HHO algorithm with increasing number of processors for function f1.

Table 15 Comparison of average running times with 1000 dimensions. The execution times of the IP-HHO and IP-GWO are obtained with 64 processors.

func.	IP-HHO	GA	PSO	BBO	FPA	GWO	IP-GWO	BAT	FA	CS	MFO	TLBO	DE
f1	5.33E+01	8.27E+01	8.29E+01	1.17E+02	2.13E+00	4.47E+00	6.76E+00	1.60E+00	5.62E+00	5.47E+00	3.23E+00	2.21E+00	2.38E+00
f2	7.91E+01	8.41E+01	8.28E+01	1.16E+02	2.09E+00	4.37E+00	6.71E+00	1.61E+00	2.57E+00	5.50E+00	3.25E+00	1.99E+00	2.28E+00
f3	1.35E+03	1.32E+02	1.30E+02	1.65E+02	5.10E+01	5.20E+01	7.51E+01	5.23E+01	3.70E+01	1.02E+02	5.11E+01	9.76E+01	5.04E+01
f4	7.18E+01	8.14E+01	8.24E+01	1.18E+02	1.90E+00	4.27E+00	6.72E+00	1.44E+00	5.43E+00	5.14E+00	3.14E+00	1.87E+00	2.21E+00
f5	1.01E+02	8.16E+01	8.33E+01	1.17E+02	2.04E+00	4.46E+00	5.52E+00	1.65E+00	5.61E+00	5.49E+00	3.31E+00	2.23E+00	2.38E+00
f6	9.02E+01	8.08E+01	8.26E+01	1.17E+02	1.88E+00	4.29E+00	5.23E+00	1.47E+00	5.51E+00	5.17E+00	3.13E+00	1.89E+00	2.19E+00
f7	1.22E+02	8.26E+01	8.52E+01	1.18E+02	4.79E+00	7.08E+00	9.05E+00	4.22E+00	6.89E+00	1.08E+01	5.83E+00	7.23E+00	4.95E+00
f8	1.18E+02	8.47E+01	8.36E+01	1.18E+02	3.18E+00	5.21E+00	6.24E+00	2.45E+00	6.04E+00	7.69E+00	4.05E+00	3.84E+00	3.23E+00
f9	1.01E+02	8.09E+01	8.33E+01	1.15E+02	2.84E+00	4.72E+00	5.36E+00	2.33E+00	5.89E+00	6.90E+00	3.94E+00	2.70E+00	3.20E+00
f10	9.96E+01	8.24E+01	8.36E+01	1.17E+02	2.96E+00	4.80E+00	5.98E+00	2.46E+00	5.98E+00	6.56E+00	4.04E+00	2.84E+00	3.41E+00
f11	1.22E+02	8.23E+01	8.38E+01	1.18E+02	3.16E+00	4.95E+00	5.78E+00	2.61E+00	6.03E+00	6.43E+00	4.22E+00	3.03E+00	3.38E+00
f12	1.31E+02	8.64E+01	8.85E+01	1.23E+02	9.09E+00	1.06E+01	1.72E+01	8.66E+00	9.17E+00	1.90E+01	9.67E+00	1.53E+01	9.14E+00
f13	1.38E+02	8.64E+01	8.90E+01	1.23E+02	9.28E+00	1.05E+01	1.54E+01	8.74E+00	9.24E+00	1.83E+01	9.66E+00	1.46E+01	9.34E+00

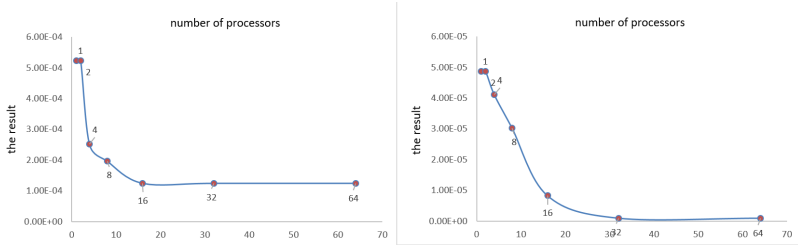


Fig. 14 The performance of the IP-HHO algorithm with increasing number of processors for function f7.

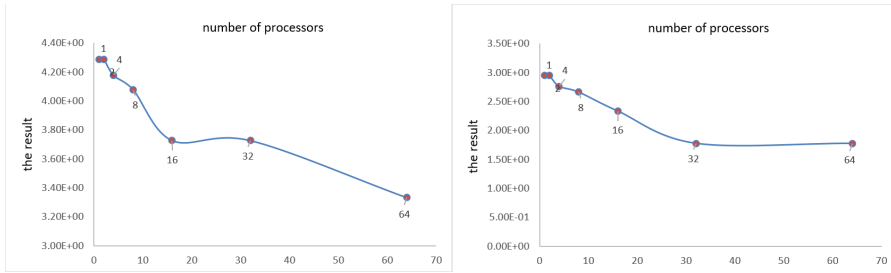


Fig. 15 The performance of the IP-HHO algorithm with increasing number of processors for function f13.

5 Conclusions and Future Work

For the first time in the literature, this work proposes an Island Parallel HHO (IP-HHO) technique for optimizing continuous multidimensional problems. The HHO method is a revolutionary optimization technique that makes use of novel mathematical notions. Thirteen unimodal and multimodal benchmark problems with four different dimensions (30, 100, 500, and 1000) were assessed to determine the IP-HHO method's performance. This unique algorithm was designed with the discovery, exploitation, and avoidance of local optima in mind. Parallel computation may be demonstrated to generate a multi-swarm environment in which thousands of hawks can optimize problems in a variety of ways. We were able to improve the performance of the sequential version of the HHO approach in all of the scenarios studied. The proposed IP-HHO approach enhances performance while preserving scalability and increasing calculation speed as the number of processors increases. As the number of dimensions increases, it is discovered that the IP-HHO approach consistently outperforms other state-of-the-art metaheuristic algorithms. We compared the performance of the IP-HHO algorithm with the parallel IP-GWO algorithm. In the results we obtained, we saw that IP-HHO gave better results than IP-GWO. Our IP-HHO algorithm has improved the solutions of f1, f2, f3, f4, f7, f12, and f14 more than the other benchmark functions in our testset.

We intend to develop hybrid versions of the HHO algorithm in the future that combine WOA and ABC optimization techniques and then apply them to a real-world case study. Hyperheuristic algorithms can also produce incredibly promising results because they mix several search approaches from diverse heuristics. As a result, witnessing the outcomes of modern metaheuristics when they are combined into a single and common population would be fascinating. Parallelized algorithms are another effective strategy for improving the quality of the outcomes.

Data Availability Statement:

The test functions and datasets that are used in this study are publicly available on website : (<https://www.sfu.ca/ssurjano/optimization.html>).

Compliance with Ethical Standards:

This study is not funded by any institution.

Conflict of Interest: There is no conflict of interest between authors.

This article does not contain any studies with human participants performed by any of the authors.

This article does not contain any studies with animals performed by any of the authors.

This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent: There is no individual participant included in the study.

References

- [1] Boussaid, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information sciences*, 237, 82-117.
- [2] Holland, J. H. (1992). Genetic algorithms. *Scientific American*, 267(1), 66-73.
- [3] Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE computational intelligence magazine*, 1(4), 28-39.
- [4] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948)*. IEEE.
- [5] Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: Theory and applications (pp. 7-15)*. Springer, Dordrecht.
- [6] Price, K. V. (2013). Differential evolution. In *Handbook of optimization (pp. 187-214)*. Springer, Berlin, Heidelberg.
- [7] Glover, F., & Laguna, M. (1998). Tabu search. In *Handbook of combinatorial optimization (pp. 2093-2229)*. Springer, Boston, MA.
- [8] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1), 67-82.
- [9] Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering*, 137, 106040.
- [10] Piotrowski, A. P., Napiorkowski, J. J., & Kiczko, A. (2012). Differential evolution algorithm with separated groups for multi-dimensional optimization problems. *European Journal of Operational Research*, 216(1), 33-46.
- [11] Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future generation computer systems*, 97, 849-872.
- [12] Alabool, H. M., Alarabiat, D., Abualigah, L., & Heidari, A. A. (2021). Harris hawks optimization: a comprehensive review of recent variants and applications. *Neural Computing and Applications*, 1-42.
- [13] Alba, E. (2005). *Parallel metaheuristics: a new class of algorithms (Vol. 47)*. John Wiley & Sons.

- [14] Alba, E., Luque, G., & Nesmachnow, S. (2013). Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research*, 20(1), 1-48.
- [15] Neumann, F., & Witt, C. (2010). Combinatorial optimization and computational complexity. In *Bioinspired Computation in Combinatorial Optimization* (pp. 9-19). Springer, Berlin, Heidelberg.
- [16] Leiserson, C. E., Rivest, R. L., Cormen, T. H., & Stein, C. (2001). *Introduction to algorithms* (Vol. 6). Cambridge, MA: MIT press.
- [17] Lefebvre, L., Whittle, P., Lascaris, E., & Finkelstein, A. (1997). Feeding innovations and forebrain size in birds. *Animal Behaviour*, 53(3), 549-560.
- [18] Sol, D., Duncan, R. P., Blackburn, T. M., Cassey, P., & Lefebvre, L. (2005). Big brains, enhanced cognition, and response of birds to novel environments. *Proceedings of the National Academy of Sciences*, 102(15), 5460-5465.
- [19] Bednarz, J. C. (1988). Cooperative hunting Harris' hawks (*Parabuteo unicinctus*). *Science*, 239(4847), 1525-1527.
- [20] Gharehchopogh, F. S., & Abdollahzadeh, B. (2021). An efficient harris hawk optimization algorithm for solving the travelling salesman problem. *Cluster Computing*, 1-25.
- [21] Bairathi, D., & Gopalani, D. (2018, December). A novel swarm intelligence based optimization method: Harris hawk optimization. In *International Conference on Intelligent Systems Design and Applications* (pp. 832-842). Springer, Cham.
- [22] Sabeena, M., & Abraham, L. (2021). Digital image forensic using deep flower pollination with adaptive Harris hawk optimization. *Multimedia Tools and Applications*, 1-23.
- [23] Dokeroglu, T., Deniz, A., & Kiziloz, H. E. (2021). A robust multiobjective Harris' Hawks Optimization algorithm for the binary classification problem. *Knowledge-Based Systems*, 107219.
- [24] Too, J., Abdullah, A. R., & Mohd Saad, N. (2019). A new quadratic binary harris hawk optimization for feature selection. *Electronics*, 8(10), 1130.
- [25] Zhang, Y., Liu, R., Wang, X., Chen, H., & Li, C. (2020). Boosted binary Harris hawks optimizer and feature selection. *Engineering with Computers*, 1-30.

- [26] Dokeroglu, T., Pehlivan, S., & Avenoglu, B. (2020). Robust parallel hybrid artificial bee colony algorithms for the multi-dimensional numerical optimization. *The Journal of Supercomputing*, 1-21.
- [27] Chen, H., Heidari, A. A., Chen, H., Wang, M., Pan, Z., & Gandomi, A. H. (2020). Multi-population differential evolution-assisted Harris hawks optimization: Framework and case studies. *Future Generation Computer Systems*, 111, 175-198.
- [28] Yildiz, A. R., Yildiz, B. S., Sait, S. M., Bureerat, S., & Pholdee, N. (2019). A new hybrid Harris hawks-Nelder-Mead optimization algorithm for solving design and manufacturing problems. *Materials Testing*, 61(8), 735-743.
- [29] Yildiz, B. S., & Yldz, A. R. (2019). The Harris hawks optimization algorithm, salp swarm algorithm, grasshopper optimization algorithm and dragonfly algorithm for structural design optimization of vehicle components. *Materials Testing*, 61(8), 744-748.
- [30] Song, S., Wang, P., Heidari, A. A., Wang, M., Zhao, X., Chen, H., ... & Xu, S. (2021). Dimension decided Harris hawks optimization with Gaussian mutation: Balance analysis and diversity patterns. *Knowledge-Based Systems*, 215, 106425.
- [31] Alba, E., & Luque, G. (2006). Evaluation of parallel metaheuristics. *Lecture Notes in Computer Science*, 4193, 9-14.
- [32] Crainic, T. G., & Toulouse, M. (2003). Parallel strategies for metaheuristics. In *Handbook of metaheuristics* (pp. 475-513). Springer, Boston, MA.
- [33] Schryen, G. (2020). Parallel computational optimization in operations research: A new integrative framework, literature review and research directions. *European Journal of Operational Research*, 287(1), 1-18.
- [34] Humphries, N. E., Queiroz, N., Dyer, J. R., Pade, N. G., Musyl, M. K., Schaefer, K. M., ... & Sims, D. W. (2010). Environmental context explains Levy and Brownian movement patterns of marine predators. *Nature*, 465(7301), 1066-1069.
- [35] Viswanathan, G. M., Afanasyev, V., Buldyrev, S. V., Murphy, E. J., Prince, P. A., & Stanley, H. E. (1996). Levy flight search patterns of wandering albatrosses. *Nature*, 381(6581), 413-415.
- [36] Sims, D. W., Southall, E. J., Humphries, N. E., Hays, G. C., Bradshaw, C. J., Pitchford, J. W., ... & Metcalfe, J. D. (2008). Scaling laws of marine predator search behaviour. *Nature*, 451(7182), 1098-1102.

- [37] Gautestad, A. O., & Mysterud, I. (2006). Complex animal distribution and abundance from memory-dependent kinetics. *ecological complexity*, 3(1), 44-55.
- [38] Shlesinger, M. F. (1989). Levy flights: Variations on a theme. *Physica D: Nonlinear Phenomena*, 38(1-3), 304-309.
- [39] Viswanathan, G. M., Afanasyev, V., Buldyrev, S. V., Havlin, S., Da Luz, M. G. E., Raposo, E. P., & Stanley, H. E. (2000). Lvy flights in random searches. *Physica A: Statistical Mechanics and its Applications*, 282(1-2), 1-12.
- [40] Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver press.
- [41] Tosun, U., Dokeroglu, T., & Cosar, A. (2013). A robust island parallel genetic algorithm for the quadratic assignment problem. *International Journal of Production Research*, 51(14), 4117-4133.
- [42] Dokeroglu, T., & Cosar, A. (2014). Optimization of one-dimensional bin packing problem with island parallel grouping genetic algorithms. *Computers & Industrial Engineering*, 75, 176-186.
- [43] Simon, D. (2008). Biogeography-based optimization. *IEEE transactions on evolutionary computation*, 12(6), 702-713.
- [44] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with computers*, 29(1), 17-35.
- [45] Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2012). Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information sciences*, 183(1), 1-15.
- [46] Yang, X. S., & Gandomi, A. H. (2012). Bat algorithm: a novel approach for global engineering optimization. *Engineering computations*.
- [47] Yang, X. S., Karamanoglu, M., & He, X. (2014). Flower pollination algorithm: a novel approach for multiobjective optimization. *Engineering optimization*, 46(9), 1222-1237.
- [48] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2011). Mixed variable structural optimization using firefly algorithm. *Computers & Structures*, 89(23-24), 2325-2336.
- [49] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, 46-61.

- [50] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249.
- [51] Derrac, J., Garca, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1), 3-18.
- [52] Blum, C., Puchinger, J., Raidl, G. R., & Roli, A. (2011). Hybrid metaheuristics in combinatorial optimization: A survey. *Applied Soft Computing*, 11(6), 4135-4151.
- [53] Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary computation*, 3(2), 82-102.
- [54] Digalakis, J. G., & Margaritis, K. G. (2001). On benchmarking functions for genetic algorithms. *International journal of computer mathematics*, 77(4), 481-506.