Journal of Future Sustainability 2 (2022) 133–144

Contents lists available at GrowingScience

Journal of Future Sustainability

homepage: www.GrowingScience.com/jfs

Weevil damage optimization algorithm and its applications

Seyed Muhammad Hossein Mousavi[a*] and S. Younes Mirinezhad[a]

[a]*Independent Researcher, Tehran, Iran*

C H R O N I C L E

Article history:
Received: March 20, 2022
Received in revised format: July 28, 2022
Accepted: October 10, 2022
Available online:
October 10, 2022

Keywords:
Weevil Damage Optimization Algorithm (WDOA)
Swarm-Based Algorithm
Metaheuristics
Optimization Test Functions
Inventory Control

A B S T R A C T

Weevils are a type of insect with elongated snouts coming from superfamily of Curculionoidea with approximately 97,000 species. Most of them consider pest and cause environmental damages but some kinds like wheat weevil, maize weevil, and boll weevils are famous to cause huge damage on crops, especially cereal grains. This research proposes a novel swarm-based metaheuristics algorithm called Weevil Damage Optimization Algorithm (WDOA) which mimics weevils' fly power, snout power, and damage power on crops or agricultural products. The proposed algorithm is tested with 12 benchmark unimodal and multimodal artificial landscapes or optimization test functions. Additionally, the proposed WDOA is employed in five engineering problems to check its robustness for problem solving. Problems are Travelling Salesman Problem (TSP), n-Queens problem, portfolio problem, Optimal Inventory Control (OIC) problem, and Bin Packing Problem (BPP). All tests' functions are compared with widely used benchmark algorithms of Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Harmony Search (HS) algorithm, Imperialist Competitive Algorithm (ICA), Firefly Algorithm (FA), and Differential Evolution (DE) algorithm. Also, all problems are tested with DE, FA, and HS algorithms. The Proposed algorithm showed robustness and speed on all functions and problems by providing precision alongside with reasonable speed.

1. Introduction

Nature holds most of the solutions for our problems and these solutions could be represented in mathematical and statistical forms. Creatures which succeeded to evolve till now and didn't lead to extinctions, worth researching and found their reasons for success. Lots of nature-inspired algorithms have been proposed in the recent decades and it is ongoing to find the best solution for our engineering problems. The process of searching for the optimal solution for a complex mathematical problem is called optimization (Zhan et al., 2022) and has application in multiple scientific areas from industry to agriculture to education. However, it is possible to solve all these problems by traditional and heuristics methods which take less runtime, but employing intelligent metaheuristics methods guarantees more optimal solutions (Zhan et al., 2022). By mimicking and modeling the behavior of creatures around us and nature, it is possible to develop a proper problem solver for almost any optimization task. Metaheuristics (Salhi, & Thompson, 2022) fits into four main categories of evolutionary-based, human behavior-based, swarm-based and physique-based which proposed algorithm is swarm-based. Proposed Weevil Damage Optimization Algorithm (WDOA) mimics fly power, snout power, damage power, and their movement toward the food source of weevils which is the inspiration source of this paper. Weevils are pest insects and from the superfamily of Curculionoidea containing 97,000 species (Abdel-Baky et al., 2022). Some species are more famous for their reproduction and damage power for crops (wheat, oats, rye, barley, rice and corn) such as wheat weevil, maize weevil, and boll weevils (Abdel-Baky et al., 2022) which are at the center of attention in this research. They have long snouts, ability to fly, and normally less than 6 millimeter (mm) in size. Female weevil lays around 36 to 254 eggs inside grains by chewing a

* Corresponding author. Tel.: +9809332892726
E-mail address: Mosavi.a.i.buali@gmail.com (S. M. H. Mousavi)

ISSN 2816-8151 (Online) - ISSN 2816-8143 (Print)

doi: 10.5267/j.ijdns.2022.10.003

hole on grain and closing the cap by gelatinous secretion. Eggs turn into larvae and then pupation occurs (Rehman & Mamoon-ur-Rashid, 2022). They generate around 6,000 eggs per year. The whole process of growing happens inside the grain and they live up to eight months after emergence depending on the environmental condition [3, 4]. This life cycle is the main idea of this paper as it describes in section 3. This research is organized into five main sections of introduction which explains the basics information required for starting the research. Section two pays to prior related research which are benchmark optimization algorithms for comparison purposes. Section three describes the proposed WDOA in detail. Section four covers all the experiments and comparison results with test functions and problems. Finally, section five includes, conclusion, suggestions and future works. Fig. 1 depicts, life cycle of wheat weevil alongside body parts and crop damages.

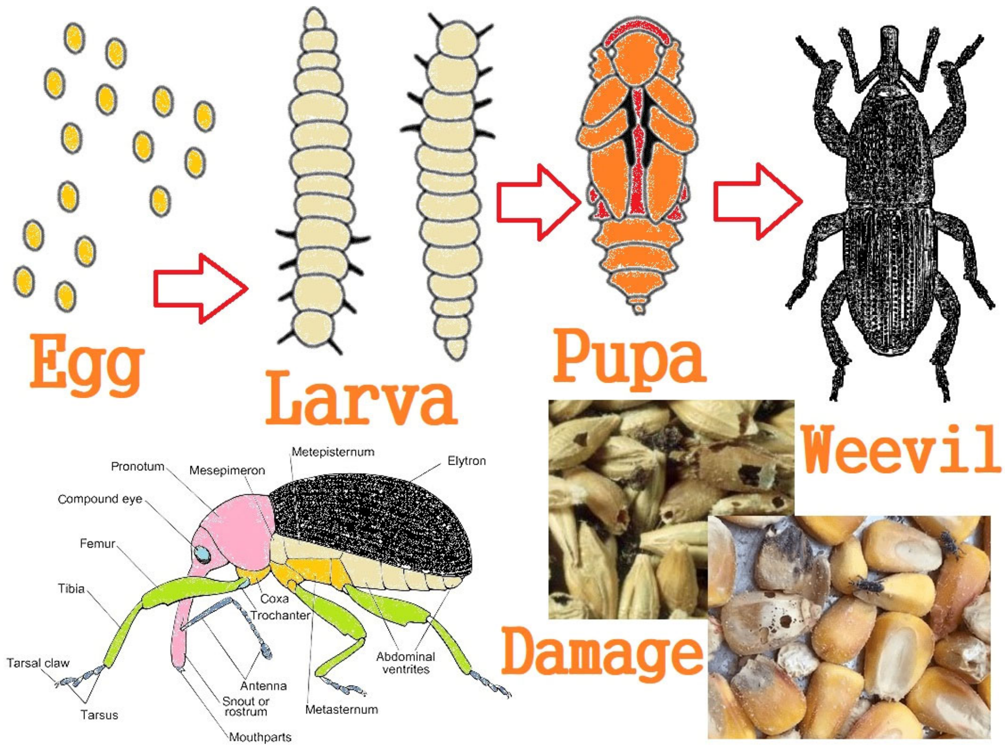

Fig. 1. life cycle of wheat weevil alongside with body parts and crop damages

2. Prior related studies

Here, four main types of metaheuristics including evolutionary-based, swarm-based, human-based, and physiques-based (Abdel-Basset et al., 2018) are covered as it is vital to compare the proposed algorithm with all main categories. One of the most famous and broadly using evolutionary algorithms is Genetic Algorithm (GA) (Mitchell, 1998) and mimics natural selection of Darwin's theory by cross over, selection and mutation operators. Making objective functions for GA is hard and computational expensive. Another famous and applicable evolutionary algorithm is Differential Evolution (DE) (Storn & Price, 1997) algorithm which is made for continuous domain optimization purposes. Parameters of crossover probability and lower/upper scaling factor serves as the main parameter here. DE is a pretty fast algorithm but doesn't converge in the beginning iterations.

Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995) is one of the benchmarks and widely applicable swarm-based algorithms out there and is capable of solving almost any mathematical problem but with different runtime depending on the parameters and problem complexity. PSO mimics the natural movement behavior of fishes and birds swarms in nature toward the most optimal solution or path. The process happens by different parameters of position, velocity, inertia weight, and learning coefficient. PSO easily falls into local minima in high dimension spaces. Another famous swarm-based metaheuristic algorithm is Firefly Algorithm (FA) (Yang, 2010) which is all about moving lower light intensity fireflies toward higher light intensity fireflies to find the best position and solution. Those fireflies with higher light intensity are candidate solutions. Light absorption coefficient and attraction coefficient are two main parameters in this algorithm. The algorithm suffers from trapping in local minima as it is a local search-based algorithm. Also, it is a computationally expensive algorithm.

One of the most famous human-based metaheuristics is Imperialist Competitive Algorithm (ICA) (Atashpaz-Gargari & Lucas, 2007) which is based on the political behavior of empires to get more colonies. Those empires with more colonies are considered to be candidate solutions. Assimilation is the main parameter which comes before the mutation factor of revolution. Also, there is a possibility for colonies to be replaced with central government or empires. This algorithm suffers from computational complexity.

One of the mentionable physique-based metaheuristics algorithms is Harmony Search (HS) (Geem et al., 2001) algorithm. The HS algorithm simulates musicians producing music. The algorithm starts with making some random solutions in the harmony memory and evaluating them by sorting. New harmonies are generated by harmony memory consideration rate factor and pitch adjustment rate which is the mutation. The goal is to find the best harmonies among the population as possible solution candidates. It is a fast algorithm but takes more iterations to convergence.

3. Weevil Damage Optimization Algorithm (WDOA)

There are Wn populations of Weevils which are randomly generated as (W1, W2,…Wn). Weevils search for a better environment for reproduction which here Environmental Situation Index (ESI) is defined as cost function. As long as the termination condition is not satisfied, following actions should be done. First, it is needed to keep the best individual out of the generated population in the previous step. Secondly, Snout Power Rate of φ and Fly Power Rate of ψ for each Weevils and based on ESI should be scattered. Also, Damage Decision Variable (DDV) determines the damage of each weevil which those weevils with higher damage power have higher chance of surviving. Additionally, μ is mutation rate which is Reproduction Environment Rate (RIR) and the more the value the better. Finally, ESI value is calculated for each weevil and after sorting the population, the best individual from the previous generation and new best individuals moves to the next generation. The WDOA would be as (a). Table 1 shows the WDOA pseudo code. Fig. 2 illustrates weevils scattering on Ackley function and comparing performance on Rosenbrock function. In this figure and for Rosenbrock function FA, GA, PSO and WDOA had better performance and DE, ICA, and HS had weaker performance.

$$WDOA = ESI \sum_{i=1}^{n} \sum_{DDV=1}^{n} (Wi[\varphi, \psi]) * RIR\ of\ \mu \tag{1}$$

Table 1
WDOA pseudo code

```
Start
  Generate a random set of Weevils (W1, W2, …, Wn)
  Compute ESI value (Cost function and sort best to worst)
    While termination criterion is not satisfied
      Keep the best individuals
      Calculate Snout power Rate φ and Fly Power Rate ψ for each Weevils based on ESI
      Searching for environment with more food source
        Select Wi with probability by φ
        Select Wj with probability by ψ
        Randomly select a DDV from Wj
        Replace random DDV Wj with Wi
        End of search
      Start Mutation μ
        Select a DDV in Wi with probability of mutation rate (RIR)
        If Wi (DDV) is selected
          Replace Wi (DDV) with a randomly generated DDV
        End if
      End of Mutation
        Recalculate the ESI value of new Weevils
        Sort population (best to worst (cost))
        Replace worst with preview generation's bests
        Sort population (best to worst (cost))
    End of while
End
```

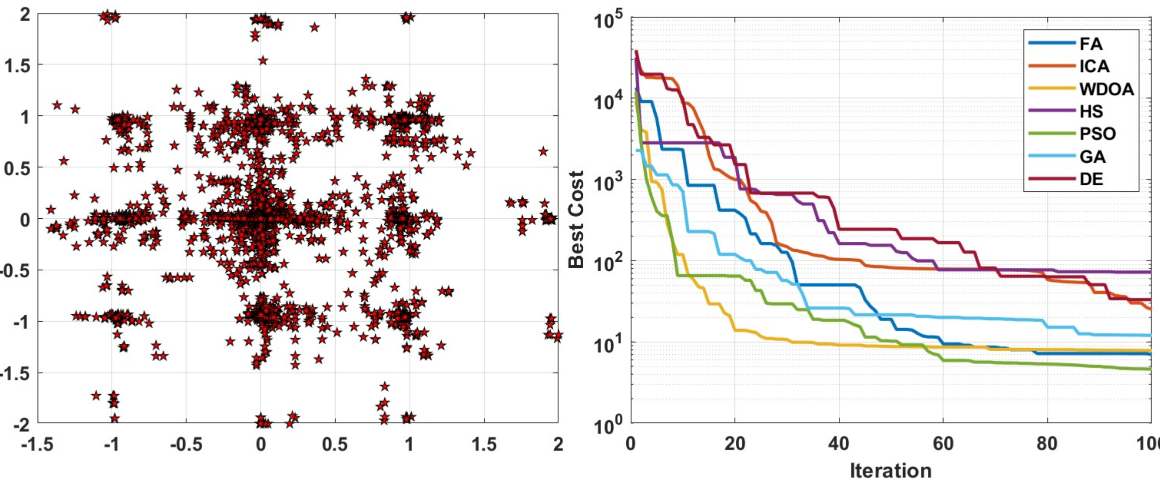

Fig. 2. Left: Weevils scattering plot on Ackley function over 100 iterations and 10 populations, right: Comparison performance of different algorithm on Rosenbrock function over 100 iterations with 30 population.

4. Evaluations, Experiment, and Results

In order to evaluate the robustness and general performance of proposed WDOA, it has to be tested with unimodal and multimodal (constrained and unconstrained) test or performance functions and compared with other metaheuristics algorithms with same parameters. Here, 12 functions of Ackley, Levy, Michalewicz, Bird, Beale, Rastrigin, De Jong, Matyas, Schwefel, Rosenbrock, Egg holder, and Easom are employed for testing purpose. Table 2 represents, mentioned test functions as equations alongside with their global minimum and working range. Figure 3 depicts, mentioned test functions as 3-Dimentional (3-D) model. Figure 4 represents, acquired results of WDOA on test functions over 100 iterations and 30 populations. Table 3 contains algorithm's parameters for using on mentioned 12 test functions. Table 4 shows the cost values achieved by all algorithms in a comparison manner on all test functions. Also, Table 5, holds returned runtime values in second for all algorithms on all test functions in a comparison manner. Experimental system is based on a Core-i7 CPU (4 GHz) PC architecture. It has to be mentioned all experiment on test functions for cost and runtime are based on parameters inside the Table 3. Table 4 represents WDOA superiority over other algorithms for seven test functions of Levy, Bird, Beale, Matyas, Schwefel, Egg holder and Easom (Mousavi et al., 2017; Jamil et al., 2013). However, PSO outperformed other algorithms on Ackley, De Jong, and Rosenbrock test functions. GA gained better cost value for Rastrigin function and FA gained better cost result for Michalewicz function comparing with other algorithms. According to Table 5 results, HS is the fastest algorithms with same parameters and DE is in the second place. PSO algorithm achieved the third fastest algorithms and proposed WDOA is in the fourth place with a bit difference with PSO algorithm. Additionally, GA placed in the fifth place and ICA in the sixth. Finally, FA with a huge runtime difference place at the end of the table.

Table 2
Employed test functions

NAME	EQUATION	GLOBAL MINIMUM	RANGE
ACKLEY – MULTI-MODAL – MANY LOCAL MINIMA	$f1(\mathbf{x}) = -20\mathrm{e}^{\left(-0.02\sqrt{D^{-1}\sum_{i=1}^{D} x_i^2}\right)} - \mathrm{e}^{\left(D^{-1}\sum_{i=1}^{D}\cos(2\pi x_i)\right)} + 20 + \mathrm{e}$	$\mathbf{x}^* = (0,\ldots,0), f(\mathbf{x}^*) = 0$	$-35 \le x_i \le 35$
LEVY – MULTIMODAL – MANY LOCAL MINIMA – CONSTRAINED	$f2(\mathbf{x}) = \sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] + (w_d - 1)^2[1 + \sin^2(2\pi w_d)]$	$\mathbf{x}^* = (1,\ldots,1), f(\mathbf{x}^*) = 0$	$-10 \le x_i \le 10$
MICHALEWICZ – MULTI-MODAL – STEEP RIDGES	$f3(\mathbf{x}) = -\sum_{i=1}^{D}\sin(x_i)\left(\sin\left(\frac{ix_i^2}{\pi}\right)\right)^{2m}$	$\mathbf{x}^* = (2.203191.57049),1.57049)$, $f(\mathbf{x}^*) = -1.8013$ for $n = 2$	$0 \le x_i \le \pi, m = 10$
BIRD, MULTI-MODAL – CONSTRAINED	$f4(x) = \sin(x_1)e^{\left[(1-\cos(x_2))^2\right]} + \cos(x_2)e^{\left[(1-\sin(x_1))^2\right]} + (x_1 - x_2)^2$	$\mathbf{x}^* = (4.7,3.15)$, $f(\mathbf{x}^*) = -106.764537$	$-2pi \le x_i \le 2pi$

BEALE – UNI-MODAL – PLATE-SHAPED	$f5(\mathbf{x}) = (1.5 - x_1 - x_1x_2)^2 + (2.25 - x_1 - x_1x_2^2)^2 + (2.625 - x_1 - x_1x_2^3)^2$	$\mathbf{x}^* = (3,0.5), f(\mathbf{x}^*) = 0$	$-4.5 \le x_i \le 4.5$
RASTRIGIN – MULTI-MODAL - MANY LOCAL MINIMA	$f6(\mathbf{x}) = -20\exp\left(-0.5\sqrt{-0.2(\mathrm{x}^2+\mathrm{y}^2)}\right) - \exp\left(0.5(\cos(2\pi \mathrm{y}))\right)$	$\mathbf{x}^* = (0,\dots,0), f(\mathbf{x}^*) = 0$	$-5 \le x_i \le 5$
DE JONG – UNIMODAL – PLATE-SHAPED	$f7(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$	$\mathbf{x}^* = (0,\dots,0), f(\mathbf{x}^*) = 0$	$-10 \le x_i \le 10$
MATYAS – UNIMODAL – PLATE-SHAPED	$f8(\mathbf{x}) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	$\mathbf{x}^* = (0,\dots,0), f(\mathbf{x}^*) = 0$	$-10 \le x_i \le 10$
SCHWEFEL – MULTI-MODAL – MANY LOCAL MINIMA	$f9(\mathbf{x}) = -\sum_{i=1}^{D}\lvert x_i\rvert + \prod_{i=1}^{D}\lvert x_i\rvert$	$\mathbf{x}^* = (0,\dots,0), f(\mathbf{x}^*) = 0$	$-10 \le x_i \le 10$
ROSENBROCK – UNIMODAL – VALLY SHAPED	$f10(\mathbf{x}) = \sum_{i=1}^{D} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$\mathbf{x}^* = (1,\dots,1), f(\mathbf{x}^*) = 0$	$-30 \le x_i \le 30$
EGG HOLDER – MULTI-MODAL – MANY LOCAL MINIMA	$f11(\mathbf{x}) = \sum_{i=1}^{D-1}\left[-(x_i + 47)\sin\sqrt{\left\lvert x_{i+1} + \frac{x_i}{2} + 47\right\rvert} - x_i\sin\sqrt{\lvert x_i - (x_{i+1} + 47)\rvert}\right]$	$\mathbf{x}^* = (512, 404.2319)$, $f(\mathbf{x}^*)\sim = 959.64$	$-512 \le x_i \le 512$
EASOM – UNI-MODAL – STEEP RIDGES	$f12(\mathbf{x}) = -\cos(x_1)\cos(x_2)e^{[-(x_1-\pi)^2-(x_2-\pi)^2]}$	$\mathbf{x}^* = (\pi,\pi), f(\mathbf{x}^*) = 0$	$-100 \le x_i \le 100$

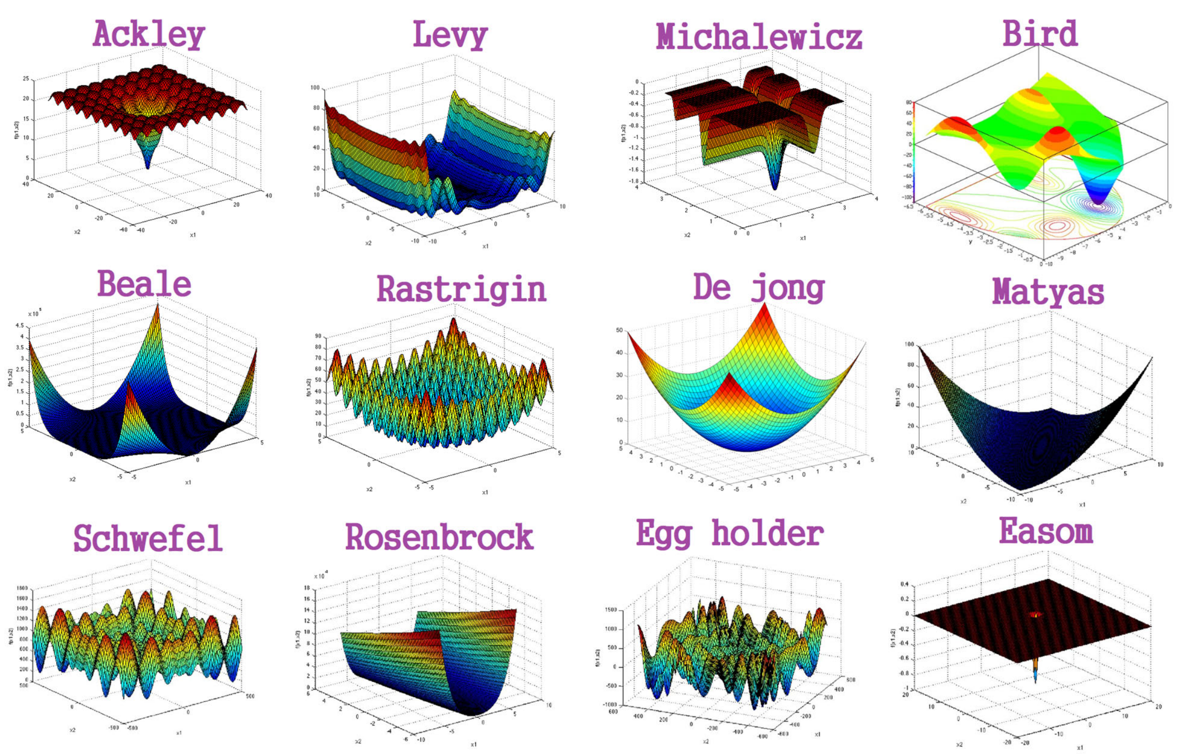

Fig. 3. 3-D model of employed test functions

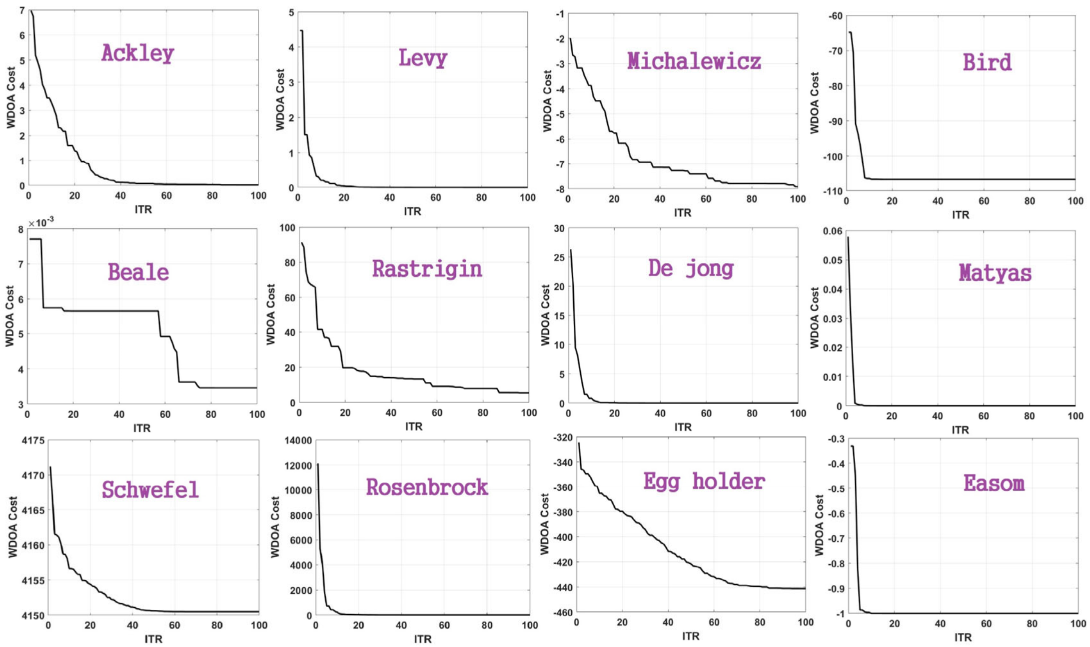

Fig. 4. Acquired results of testing WDOA on performance test functions over 100 iterations with 30 population size.

Table 3
Algorithm's parameters for test functions

PARAMETERS	GA	DE	PSO	FA	ICA	HS	WDOA
DECISION VARIABLES	10	10	10	10	10	10	10
DECISION VARIABLES SIZE	[1,10]	[1,10]	[1,10]	[1,10]	[1,10]	[1,10]	[1,10]
LOWER BOUND OF VARIABLES (LV)	-5	-5	-5	-5	-5	-5	-5
UPPER BOUND OF VARIABLES (UV)	5	5	5	5	5	5	5
ITERATIONS	200	200	200	200	200	200	200
POPULATION SIZE (P)	30	30	30	30	30	30	30
CROSSOVER PERCENTAGE (PC)	0.7	-	-	-	-	-	-
NUMBER OF OFFSPRING'S (PARENTS)	2*(PC*P/2)	-	-	-	-	-	-
MUTATION PERCENTAGE (MP)	0.3	-	-	-	-	-	-
NUMBER OF MUTANTS	MP*P	-	-	-	-	-	-
MUTATION RATE	0.2	-	0.2	0.2	-	-	$\sum_{i=1}^{P}[0,1]$
INERTIA WEIGHT	-	-	1	-	-	-	-
INERTIA WEIGHT DAMPING RATIO	-	-	0.99	-	-	-	-
PERSONAL LEARNING COEFFICIENT	-	-	1.5	-	-	-	-
GLOBAL LEARNING COEFFICIENT	-	-	2	-	-	-	-
LIGHT ABSORPTION COEFFICIENT	-	-	-	1	-	-	-
ATTRACTION COEFFICIENT	-	-	-	2	-	-	-
MUTATION DAMPING RATIO	-	-	-	0.98	-	-	-
LOWER BOUND OF SCALING FACTOR	-	0.2	-	-	-	-	-
UPPER BOUND OF SCALING FACTOR	-	0.8	-	-	-	-	-
CROSSOVER PROBABILITY	-	0.2	-	-	-	-	-
NUMBER OF EMPIRES	-	-	-	-	15	-	-
SELECTION PRESSURE	-	-	-	-	1	-	-
ASSIMILATION COEFFICIENT	-	-	-	-	1.5	-	-
REVOLUTION PROBABILITY	-	-	-	-	0.05	-	-
REVOLUTION RATE	-	-	-	-	0.1	-	-
COLONIES MEAN COST COEFFICIENT	-	-	-	-	0.2	-	-
NUMBER OF NEW HARMONIES	-	-	-	-	-	15	-
HARMONY MEMORY CONSIDERATION RATE	-	-	-	-	-	0.9	-
PITCH ADJUSTMENT RATE	-	-	-	-	-	0.1	-
FRET WIDTH (BANDWIDTH)	-	-	-	-	-	0.2*(UV-LV)	-
FRET WIDTH DAMP RATIO	-	-	-	-	-	0.995	-
DAMAGE RATE (DR)	-	-	-	-	-	-	0.2
NUMBER OF REMAINED WEEVILS (RW)	-	-	-	-	-	-	DR*P
NUMBER OF NEW WEEVILS	-	-	-	-	-	-	P-RW
MUTATION PROBABILITY	-	-	-	-	-	-	0.1
WEEVIL SNOUT POWER RATE	-	-	-	-	-	-	0.8
WEEVIL FLY POWER RATE	-	-	-	-	-	-	0.3*(UV-LV)

Table 4
Cost value comparison of all algorithms on all test functions

TEST FUCNTION	*COST*	GA	DE	PSO	FA	ICA	HS	WDOA
ACKLEY	*Avg*	0.03297	0.0453	**0.0091**	0.0199	0.0566	0.46455	0.0185
-	*Std*	0.0118	0.0185	0.0010	0.0054	0.0140	0.0521	0.0062
LEVY	*Avg*	0.0005	0.0013	0.0002	0.0004	0.0003	0.0069	**0.0001**
-	*Std*	0.0001	0.0008	0.0002	0.0002	0.0002	0.0021	0.0001
MICHALEWICZ	*Avg*	-8.6112	-6.8652	-6.5744	**-9.2512**	-7.9669	-8.0001	-8.8001
-	*Std*	0.499	0.724	0.660	0.571	0.883	0.973	0.485
BIRD	*Avg*	-106.7644	-106.6619	-87.3109	-106.7640	-106.7643	-106.7641	**-106.7645**
-	*Std*	0.0759	0.0997	0.0188	0.0291	0.0627	0.0549	0.0112
BEALE	*Avg*	0.0002	0.0008	0.0002	0.0003	0.0009	0.0035	**0.0001**
-	*Std*	0.0001	0.0004	0.0002	0.0002	0.0003	0.0019	0.0001
RASTRIGIN	*Avg*	**3.0024**	6.9254	8.6132	9.1490	4.8080	4.2500	6.8227
-	*Std*	0.678	0.935	0.999	1.255	2.603	2.200	0.528
DE JONG	*Avg*	2.6049e-06	3.5631e-08	**4.5891e-22**	7.8458e-06	5.0329e-10	0.0032	5.8025e-06
-	*Std*	0.00001	0.00001	0.000001	0.00001	0.00001	0.0006	0.00001
MATYAS	*Avg*	8.2199e-08	7.1474e-05	7.9125e-08	8.3302e-02	8.0021e-02	0.0016	**7.8968e-09**
-	*Std*	0.00002	0.00003	0.00001	0.00002	0.00002	0.0009	0.000001
SCHWEFEL	*Avg*	4317.2451	4560.0023	4290.1118	5120.9228	5012.5582	5960.1200	**4150.4915**
-	*Std*	549	675	421	973	1012	659	334
ROSENBROCK	*Avg*	4.6691	6.7831	**4.0109**	4.2533	5.9201	7.6500	4.1559
-	*Std*	1.630	1.937	0.970	0.886	1.300	2.381	0.771
EGG HOLDER	*Avg*	-440.7710	-412.5280	-425.3198	-439.8715	-440.6699	-400.1285	**-441.2976**
-	*Std*	29	35	72	12	13	66	10
EASOM	*Avg*	-0.7519	-0.8851	-0.9258	-0.3502	-0.5449	-0.6501	**-1.2000**
-	*Std*	0.156	0.254	0.220	0.147	0.363	0.211	0.110

Table 5
Runtime comparison of all algorithms on all test functions (in seconds)

TEST FUCNTION	*TIME*	GA	DE	PSO	FA	ICA	HS	WDOA
ACKLEY	*Avg*	0.571	0.444	0.479	3.529	0.592	**0.331**	0.495
-	*Std*	0.052	0.068	0.041	0.998	0.063	0.029	0.028
LEVY	*Avg*	0.491	0.403	0.471	3.149	0.518	**0.329**	0.472
-	*Std*	0.041	0.036	0.058	1.100	0.072	0.069	0.046
MICHALEWICZ	*Avg*	0.439	0.409	0.421	2.490	0.466	**0.301**	0.430
-	*Std*	0.035	0.047	0.021	0.881	0.052	0.038	0.046
BIRD	*Avg*	0.467	0.399	0.445	3.300	0.499	**0.295**	0.442
-	*Std*	0.024	0.039	0.042	0.669	0.091	0.025	0.036
BEALE	*Avg*	0.441	0.387	0.415	3.259	0.500	**0.349**	0.413
-	*Std*	0.078	0.086	0.066	0.721	0.450	0.036	0.011
RASTRIGIN	*Avg*	0.479	0.398	0.407	3.127	0.508	**0.327**	0.414
-	*Std*	0.041	0.026	0.037	0.611	0.058	0.068	0.032
DE JONG	*Avg*	0.498	0.400	0.420	4.015	0.557	**0.309**	0.428
-	*Std*	0.062	0.043	0.079	1.263	0.093	0.006	0.041
MATYAS	*Avg*	0.426	0.323	0.399	3.690	0.486	**0.300**	0.395
-	*Std*	0.035	0.049	0.054	0.341	0.061	0.034	0.025
SCHWEFEL	*Avg*	0.462	0.369	0.441	3.550	0.502	**0.336**	0.455
-	*Std*	0.055	0.051	0.059	0.493	0.067	0.054	0.049
ROSENBROCK	*Avg*	0.408	0.347	0.400	3.246	0.443	**0.305**	0.404
-	*Std*	0.035	0.027	0.010	0.657	0.026	0.009	0.014
EGG HOLDER	*Avg*	0.489	0.391	0.452	4.682	0.604	**0.303**	0.477
-	*Std*	0.053	0.059	0.047	0.829	0.091	0.026	0.021
EASOM	*Avg*	0.460	0.352	0.420	3.005	0.487	**0.310**	0.422
-	*Std*	0.026	0.011	0.028	0.663	0.055	0.027	0.006

Each metaheuristics optimization algorithm must be able to solve various types of mathematical and statistical problems in different domains such as computer vision (Mousavi et al., 2022), pattern recognition (Mousavi & Ilanloo., 2022), machine learning (Mousavi & Mosavi, 2021), graph theory (Thulasiraman, 2016), and industry applications. Proposed WDOA is tested in 15 areas and problems and passed all tests but here, five problems are selected. Travelling Salesman Problem (TSP) (Reda et al., 2022), n-Queens [15], portfolio (Khan et al., 2022), Optimal Inventory Control [OIC] (Rachi et al., 2022), and Bin Packing Problems (BPP) (Ekici, 2022) are selected for this research. TSP is a graph-based problem in which solutions should find the shortest paths between locations by visiting them just once and returning to the starting point. As it is an NP-hard (Woeginger, 2003) problem, it is better to be solved by optimization methods. DE algorithm from evolutionary-based, FA from swarm-based, HS algorithm from physique-based are selected to compare with proposed WDOA in TSP. Problem is defined with 15 points as x= [87 50 22 19 3 67 86 52 5 21 65 14 88 70 40] and y= [32 56 97 47 27 43 39 89 5 79 56 1 21 18 20] in a 2-D space. Population is 50 individuals which runs over 500 iterations for all algorithms. In this experiment, reaching cost values of 317.9544 shows that the algorithm satisfied the condition and solved the problem. The first model is a basic random model with lots of error which improves during iterations by optimization algorithms.

Figure 5 illustrates acquired results by different algorithms on the TSP which all algorithms could solve the problem before iteration 500 with different run time speed. Here, each blue star is a location and red lines are found paths by algorithms. Table 6, presents cost and run time values after the experiment for all algorithms on TSP. However, HS was the fastest algorithm but FA had fastest convergence. Proposed WDOA placed in second place regarding both cost and runtime parameters. As an additional explanation, it has to be mentioned that, by increasing the number of locations more than 20, HS and DE lose their performance but FA and WDOA, not.

Another graph-based problem is the n-queens problem which normally consists of eight queens but here are 16 queens. Metaheuristic should find a solution that these 16 queens don't attack each other by being in the same row and column. Also, solutions must prevent any diagonal attack. Here, cost value of 0 means that the algorithm is capable of solving the problem. The basic model is a 16 by 16 plane with randomly scattered queens which improves during iterations by different algorithms to achieve 'no hits' situation. Figure 6. Represents iterations and solutions for n-queens problem by all algorithms using 200 individuals as populations over 300 iterations. Here, each white triangle is a queen and each black line is a violation or hit which having no hits means the best solution. Additionally, Table 7, shows the acquired cost and runtime results for n-queens problem by all algorithms. DE algorithm by 1 queen hit and HS algorithm by 2 queens hits failed the problem; however, FA and WDOA could solve the problem with no hits. It has to be mentioned that HS and DE algorithms could solve the problem with less than 12 queens. By increasing the number of queens over 20, population size for WDOA must increase, or it would fail, too; but this scenario is not the same for FA as it could solve harder problems with 200 iterations.

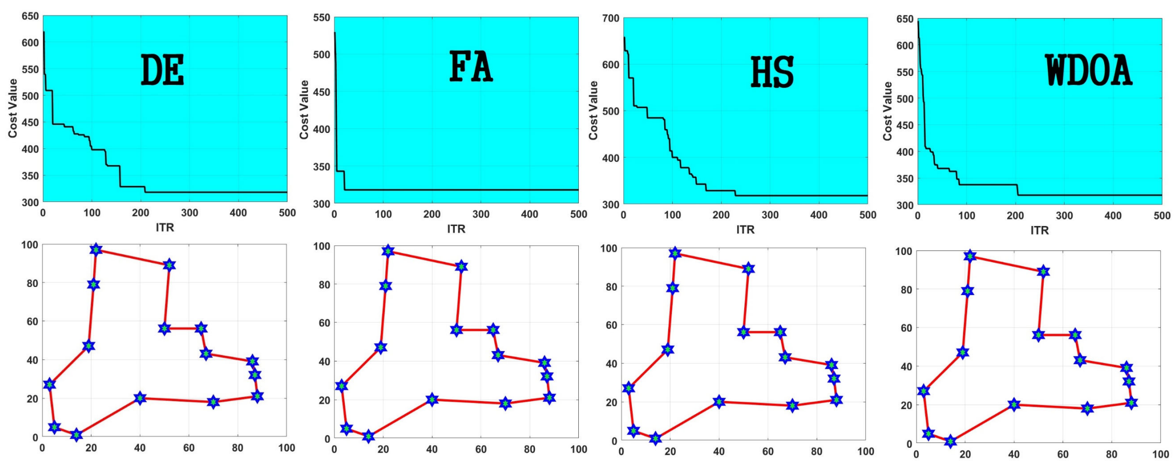

Fig. 5. Acquired results for TSP by all algorithms (top row: iterations and convergences, bottom row: solutions)

Table 6
Cost and runtime values of all algorithms on TSP

		DE	FA	HS	WDOA
Is Problem Solved?		✓	✓	✓	✓
Cost Value	Avg	317.9544	317.9544	317.9544	317.9544
	Std	50	0	61	35
Run Time	Avg	1.550 (s)	3.158 (s)	0.909 (s)	1.987 (s)
	Std	0.158	0.620	0.214	0.361

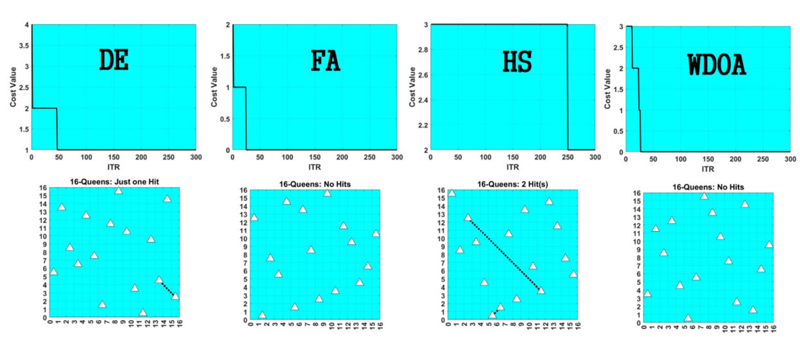

Fig. 6. Acquired results for n-queens problem by all algorithms (top row: iterations and convergences, bottom row: solutions)

Table 7
Cost and runtime values of all algorithms on n-queens problem

		DE	FA	HS	WDOA
Is Problem Solved?		×	✓	×	✓
Cost Value	Avg	1	0	2	0
	Std	0	0	1	0
Run Time	Avg	4.010 (s)	8.522 (s)	0.740 (s)	5.667 (s)
	Std	1.258	4.339	0.124	0.347

Portfolio is a definition in finance which is based on receiving some values (prices) as stock or any other financial assets and converting them into returns by aiming to increase the return and decrease the risk. Finally, those solutions which have highest return and lowest risk, would be considered as a vector called efficient frontier. Normally, portfolio solves by traditional methods such as mean-variance, mean semivariance, and mean absolute deviation but not always guaranteed the best solution. Based on experiments on this paper, however optimization methods for portfolio problems take longer but always guarantee the best solution by different algorithms. Here, the data input is a 50*5 matrix consisting of 20 stock values for 5 companies and optimization algorithms try to suggest the best company based on return and risk which they calculate. Figure 7 represents cost values and solutions of all algorithms on the data over 100 iterations. Clearly all algorithms managed to solve the problem. Table 8 shows acquired values for cost, runtime and solutions of all algorithms. All algorithms suggest the fifth company with the same and the best risk and return value.

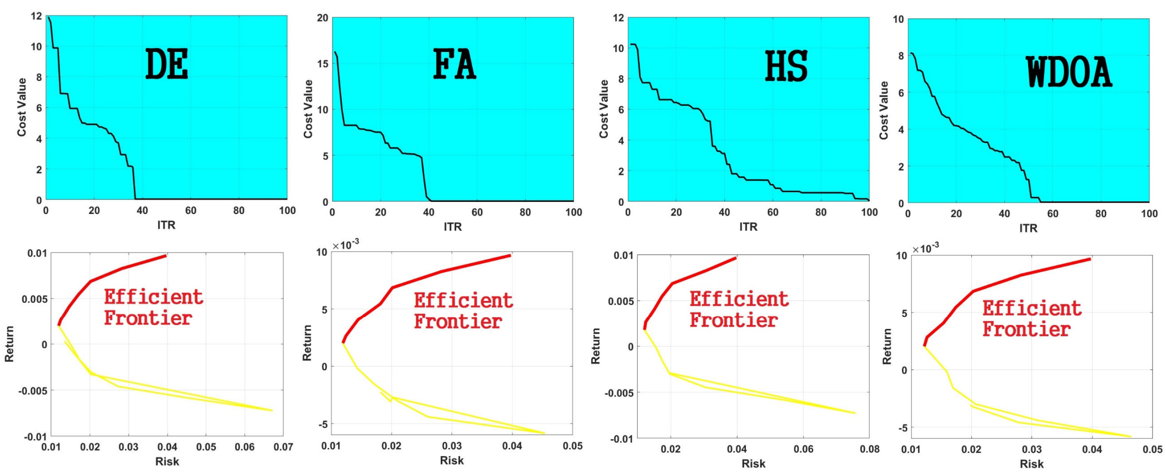

Fig. 7. Acquired results for Portfolio problem by all algorithms (top row: iterations and convergences, bottom row: solutions)

Table 8
Cost, runtime, and solution values of all algorithms on portfolio problem

		DE	FA	HS	WDOA
Is Problem Solved?		✓	✓	✓	✓
Cost Value	Avg	0.0397	0.0397	0.0397	0.0397
	Std	0.0001	0.0001	0.0001	0.0001
Run Time	Avg	35.323 (s)	45.500 (s)	30.520 (s)	32.478 (s)
	Std	5	7	2	4
Solution	Stocks	0 0 0 0 1	0 0 0 0 1	0 0 0 0 1	0 0 0 0 1
	Risk	0.0398	0.0398	0.0398	0.0398
	Return	0.0097	0.0097	0.0097	0.0097

Optimal Inventory Control (OIC) is another optimization problem in management and business. OIC simply refers to having the required quantity of product in inventory at all times for the center or facility in order to prevent typical inventory problems. As an experiment, we examine two products or items in a 5-time unit with an initial inventory level of zero and a maximum capacity of 400. There is demand of [59 34 84 69 28] for the first product and demand of [84 34 31 46 78] for the second product. Also, cost or price for both products are as [188 138 176 104 153] and [149 129 117 181 196]. Additionally, maintenance cost for both products is as [3 6 10 2 4] and [8 10 4 8 5]. The system must make a balance between order amount and inventory amount without passing a maximum capacity of 400 at the end of the run. Figure 8 depicts acquired cost and solutions for defined OIC problems by all algorithms over 200 iterations and 400 populations. Table 9 holds runtime, cost and solutions for the same OIC problem by all algorithms. Best performance belongs to FA algorithm with 1680 maintenance cost and worst result belongs to DE with 2464 maintenance cost. Proposed WDOA with 1714 is in second place and HS placed in the third place with maintenance cost of 2229. Additionally, HS was the fastest algorithm and FA the slowest.

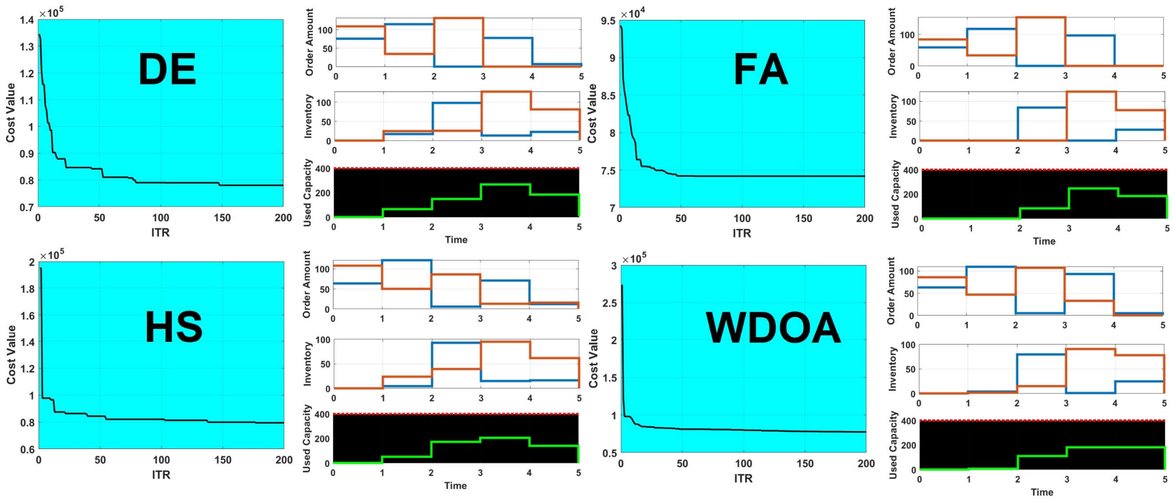

Fig. 8. Acquired results for OIC problem by all algorithms.

Table 9
Cost, runtime, and solution values of all algorithms on OIC problem

		DE	FA	HS	WDOA
Is Problem Solved?		✓	✓	✓	✓
Cost Value	Avg	78005	74181	79466	77424
	Std	251	187	365	221
Run Time	Avg	7.363 (s)	34.112 (s)	4.231 (s)	8.111 (s)
	Std	1.240	3.652	0.527	0.699
Solution	Orders or Products Costs	75541	72501	77237	75710
	Inventory or Maintenance Costs	2464	1680	2229	1714
	Used Capacity	[67 150 268 185 8]	[0 84 248 184 0]	[53 173 205 141 1]	[8 110 183 181 2]

Bin Packing Problem (BPP) is a NP-hard optimization problem which has applications in loading trucks with constant capacity, creating backups for media and more. As it is NP-hard, one of the best ways to solve it is by nature-inspired methods. The problem is based on the number of items which should be fitted inside the number of bins or containers. Items have different sizes but bins have a constant size. The main objective is to use less bins for all items. Our experiment is using bin capacity of 35 on 25 items as follows: [6 3 4 6 8 7 4 7 7 5 5 6 7 7 6 4 8 7 8 8 2 1 4 9 6] for input. Figure 9 illustrates iterations and convergence of all algorithms on BPP over 100 iterations with 10 individuals as populations. Additionally, Table 10 contains cost, runtime and solutions for all algorithms on BPP with the same problem data. Regarding run time FA is the slowest and HAS the fastest. However, FA and WDOA achieved less cost and bins to complete the task with just five bins but DE and HS took six bins.

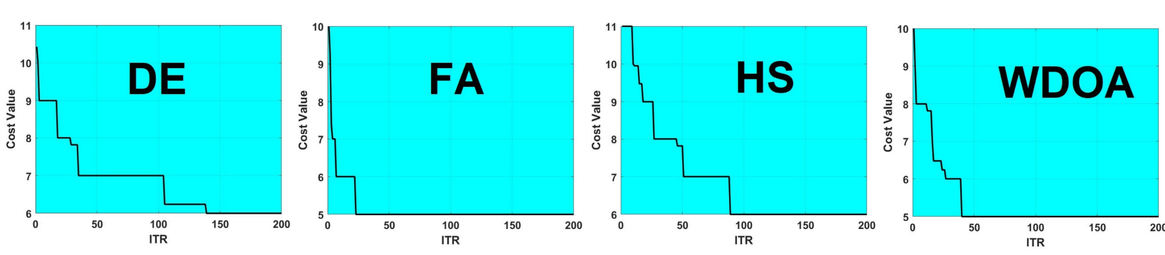

Fig. 9. Iterations and convergence for BPP of all algorithms

Table 10
Cost, runtime, and solution values of all algorithms on BPP problem

		DE	FA	HS	WDOA
Is Problem Solved?		✓	✓	✓	✓
Cost Value	Avg	6	5	6	5
	Std	1	0	1	1
Run Time	Avg	0.656 (s)	1.937 (s)	0.578 (s)	0.980 (s)
	Std	0.057	0.251	0.091	0.133
Solution	Bins' Content Bin size = 35 Items = 25	[3,8,4,6] [6,7,8,5] [7,4,6,7,9,2] 6 [4,8,5,7,4] [8,6,7,7,1]	[8,5,4,7] [1,8,6,6,4,6,3] [7,7,4,7] [4,2,7,5,8,7] [8,6,9,6]	[7,7] [7,8,8,7,2] [9,8,4,5] [1,6,4] [6,3,4,8,5,6] [6,7,4,6,7]	[9,4,8,7,5] [6,8,6,6] [6,2,7,5] [7,3,7,7,8] [8,1,7,6,4,4,4]

5. Conclusion, Suggestions and Future Works

Formulating natural swarm-based behavior of weevils in order to solve NP-hard and other optimization problems was a success. The WDOA showed high accuracy along-side with keeping low runtime on test discrete and continuous (unimodal and multimodal) test functions compared with other algorithms. The WDOA easily could compete with the famous metaheuristics algorithm and showed decent performance in almost 15 problems of which just five of them are reported in this research. Fast convergence was another distinguishing factor for WDOA compared with other similar algorithms. It is suggested to test the algorithm with more optimization test functions and evaluating the performance of the algorithm. Employing WDOA on hub location allocation, minimum spanning tree (Thulasiraman, 2016), image quantization, image segmentation (Castleman, 1996), and economic dispatching (Yuan & Yang, 2019) problems are of future works. Overall, it can be concluded that proposed WDOA has robust accuracy in level of PSO and FA algorithms with decent speed but not as fast as the fastest algorithm such as HS and DE. Fast convergence in the beginning iterations and applicability in different domains from graph-based problems to management problems are two other advances of WDOA.

References

Abdel-Baky, N. F., Aldeghairi, M. A., Motawei, M. I., Al-Shuraym, L. A., Al-Nujiban, A. A., Alharbi, M., & Rehan, M. (2022). Genetic Diversity of Palm Weevils, Rhynchophorus Species (Coleoptera: Curculionidae) by Mitochondrial COI Gene Sequences Declares a New Species, R. bilineatus in Qassim, Saudi Arabia. Arabian *Journal for Science and Engineering*, 1-18.

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. Computational intelligence for multimedia big data on the cloud with engineering applications, 185-231.

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. *2007 IEEE Congress on Evolutionary Computation*, 4661-4667.

Castleman, K. R. (1996). *Digital image processing*. Prentice Hall Press.

Ekici, A. (2022). Variable-sized bin packing problem with conflicts and item fragmentation. *Computers & Industrial Engineering, 163*, 107844.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony search. *Simulation, 76*(2), 60-68.

Jamil, M., Yang, X. S., & Zepernick, H. J. (2013). Test functions for global optimization: a comprehensive survey. *Swarm intelligence and Bio-inspired Computation*, 193-222.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95-International Conference on Neural Networks, 4*, 1942-1948.

Khan, A. T., Cao, X., Brajevic, I., Stanimirovic, P. S., Katsikis, V. N., & Li, S. (2022). Non-linear Activated Beetle Antennae Search: A novel technique for non-convex tax-aware portfolio optimization problem. *Expert Systems with Applications, 197*, 116631.

Mitchell, M. (1998). An introduction to genetic algorithms. MIT press.

Mousavi, S. M. H., & Ilanloo, A. (2022, March). Seven Staged Identity Recognition System Using Kinect V. 2 Sensor. *In 2022 9th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)* (pp. 1-7). IEEE.

Mousavi, S. M. H., & Mosavi, S. M. H. (2021, October). Automatic Infrared-Based Volume and Mass Estimation System for Agricultural Products: Along with Major Geometrical Properties. In 2021 11th International Conference on Computer Engineering and Knowledge (ICCKE) (pp. 140-149). IEEE.

Mousavi, S. M. H., & Mosavi, S. M. H. (2022, March). A New Edge and Pixel-Based Image Quality Assessment Metric for Colour and Depth Images. *In 2022 9th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS) (pp. 1-11). IEEE.*

Mousavi, S. M. H., MiriNezhad, S. Y., & Dezfoulian, M. H. (2017, October). Galaxy gravity optimization (GGO) an algorithm for optimization, inspired by comets life cycle. *In 2017 Artificial Intelligence and Signal Processing Conference (AISP)* (pp. 306-315). IEEE.

Rachih, H., Mhada, F., & Chiheb, R. (2022). Simulation optimization of an inventory control model for a reverse logistics system. *Decision Science Letters, 11*(1), 43-54.

Reda, M., Onsy, A., Elhosseini, M. A., Haikal, A. Y., & Badawy, M. (2022). A discrete variant of cuckoo search algorithm to solve the Travelling Salesman Problem and path planning for autonomous trolley inside warehouse. *Knowledge-Based Systems, 252*, 109290.

Rehman, G., & Mamoon-ur-Rashid, M. (2022). Evaluation of entomopathogenic nematodes against red palm weevil, Rhynchophorus ferrugineus (Olivier)(Coleoptera: Curculionidae). *Insects, 13*(8), 733.

Salhi, S., & Thompson, J. (2022). An Overview of Heuristics and Metaheuristics. *The Palgrave Handbook of Operations Research*, 353-403.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization, 11*(4), 341-359.

Thulasiraman, K. (2016). *Handbook of graph theory, combinatorial optimization, and algorithms* (Vol. 34). S. Arumugam, A. Brandstädt, & T. Nishizeki (Eds.). CRC Press.

Woeginger, G. J. (2003). *Exact algorithms for NP-hard problems: A survey. In Combinatorial optimization—eureka, you shrink!* (pp. 185-207). Springer, Berlin, Heidelberg.

Yang, X.-S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver Press.

Yao, Y., & Zeilberger, D. (2022). Numerical and symbolic studies of the peaceable queens problem. *Experimental Mathematics, 31*(1), 269-279.

Yuan, G., & Yang, W. (2019). Study on optimization of economic dispatching of electric power system based on Hybrid Intelligent Algorithms (PSO and AFSA). *Energy, 183*, 926-935.

Zhan, Z. H., Shi, L., Tan, K. C., & Zhang, J. (2022). A survey on evolutionary computation for complex continuous optimization. *Artificial Intelligence Review, 55*(1), 59-110.