NTU Journal of Pure Sciences (2024) 3 (1) : 1-9
DOI: https://doi.org/10.56286/ntujps.v3i1.710

IRAQI Academic Scientific Journals

ISSN: 2789-1089 EISSN: 2789-1097
NTU Journal of Pure Sciences
Available online at: https://journals.ntu.edu.iq/index.php/NTU-JPS/index

A comparative study on the performance of Gray Wolves Optimization Multi-Free Dynamic Schema

Radhwan Yousif Al-Jawadi[1]
1. Administrative Technical College, Northern Technical University, Mosul , Iraq.

Article Informations

Received: 07-10- 2023,
Accepted: 10-11-2023,
Published online: 06-04-2024

Corresponding author:
Name: Radhwan Y. Al-jawadi
Affiliation: Administrative Technical College, Northern Technical University, Mosul , Iraq.
Email: radwan.aljawadi@ntu.edu.iq

Key Words:
Single-objective optimization, Multi-free dynamic schema, Gray Wolves Optimization, Genetic algorithm.

A B S T R A C T

The most difficult step is to create new optimization algorithms and examine them using test functions. In this work, we present a comparison study on the performance of Gray Wolves Optimization (GWO) and Multi-free dynamic schema (MFDS) algorithms. The (MFDS) algorithm is a sophisticated optimization method created for solving optimization problems. It contains different operators (dynamic schema operator, dissimilarity operator, similarity operator and free dynamic schema operator). Where, The (GWO) is a meta-heuristic optimization algorithm inspired by the social behavior of grey wolves in a pack. This study focused on the run time and the number of iterations to reach the optimal solutions. The sample of this comparison was on ten functions. The results showed the superiority of an algorithm (MFDS) on (GWO) algorithm in most test functions, especially at the run time. The performance of any single-objective optimization algorithm is a tool to measures the effectiveness of any algorithm for determining out the best solution for a specific problem.

Introduction

Single objective optimization algorithms are computational methods designed to find the best solution to a specific optimization problem where a single objective or criterion needs to be either maximized or minimized [1]. The applications of optimization algorithms are used in different fields, like as: (an engineering design, financial portfolio optimization to machine learning, operations research, and data analysis) [2], [3], [4].

The performance of single-objective optimization algorithms is a crucial aspect that measures their effectiveness in getting the best solution for the problem at hand. The choice of optimization algorithm can significantly impact the efficiency and quality of the results. Therefore, assessing and comparing the performance of these algorithms is essential to ensure their suitability for different applications [4].

Benchmarking plays a major role in algorithm comparison and performance evaluation. The purpose of the benchmarking trials is to identify or estimate the optimal algorithm for resolving practical issues [3], [5].

The core principle of single objective optimization is to explore and search through a solution space, typically guided by an fitness function. This function returns a numerical value indicating the quality of a given solution. The optimization algorithm then iteratively refines and explores candidate solutions to approach the optimal or near-optimal solution for the specified objective [6], [7]. Several optimization algorithms have been developed to address single objective optimization problems, each with its own strengths and suitability for different types of problems. Common examples include:

Gradient Descent (GD): A first-order optimization algorithm that iteratively adjusts a solution based on the gradient of the objective function. It's widely used for smooth and differentiable functions [8], [9].

Genetic Algorithms (GA): Inspired by the principles of natural selection and evolution, genetic algorithms use genetic operators like mutation and crossover to explore the solution space and find optimal solutions [8], [10].

Particle Swarm Optimization (PSO): PSO mimics the social behavior of birds or particles in a swarm to search for optimal solutions. It's effective for problems with many local optima [11], [12].

These algorithms differ in their exploration-exploitation trade-offs, convergence speeds, and suitability for different types of objective functions. The choice of algorithm depends on the specific problem and its characteristics.

In this paper a comparative study was presented of (GWO) and (MFDS) algorithms. The (GWO) algorithm is a nature-inspired metaheuristic algorithm that mimics the hunting behavior of gray wolves. GWO's strength lies in its efficient exploration of solution spaces. However, GWO's applicability might be influenced by factors like parameter tuning and its adaptability to different optimization challenges [13], [14]. On the other hand, The (MFDS) algorithm is a complex optimization approach designed to solve optimization problems. It involves a combination of dynamic schema operators, dissimilarity operators, similarity operators, and a unique use of free dynamic schema operators with part of population which regenerates randomly new chromosomes in each iteration [15]. This integration enables it to traverse complex solution spaces. The strengths of the algorithm include its diverse set of operators, including dynamic scheme operators that promote fast convergence. However, their complexity and parameter dependence may pose challenges across diverse problem domains. Let's discuss the characteristics of each algorithm and their differences in Section (3, 4).

Literature review

The study of [3] intends to evaluate the meta-heuristic algorithms performance on the Congress on Evolutionary Computation (CEC-2021) benchmark problems, considering factors such as convergence speed and solution quality on different problem types. The authors used binary operators such as bias, shift and rotation to parameterize objective functions. Also, they illustrate better understanding of the strengths and weaknesses of various single optimization algorithms such as Differential evolution (DE), Gaining sharing knowledge-based algorithm (GSK), (GWO), Particle swarm optimization (PSO). The results of [3] provide offering guidance on which algorithms may be best suited for different problem domains.

In [16] a comparative study of the Genetic Algorithm (GA) and (PSO) was presented, the authors conclude that the PSO is quite similar to GA. are evolutionary search methods change from a set of points to another set of points within an iteration. The GA and PSO are valuable components of evolutionary optimization techniques, their use is restricted to specific problems due to certain drawbacks. The overall performance can be

raised by combining GA and PSO in order to solve these problems. A combining these two algorithms results combination of benefits of GA and PSO in real-world applications. Thus, a hybrid GA and PSO algorithms is a promising area for further study.

In [17] the authors have compared GA and Ant Colony Optimization (ACO) on the guitar tab transcription problem, to convert a song in standard music notation (sheet music) into an alternative notation known as a guitar tab. The authors found that the results from experiments by using a new dataset of 148 songs show the efficacy of the ACO approach to gives the best solutions for this task.

A comparison of ACO and PSO algorithms for common problem optimization (distance optimization) was reported by the authors of [18]. Both methods were used to successfully solve the problem, and their respective performances were then quantitatively compared. The outcomes of the simulation demonstrate that the newly created ant colony optimization technique is the more reliable and superior of the two.

In [19] the authors conclude that the field of meta-algorithms, it is well known that analyzing the algorithm's solution over N trial runs is required to determine performance; one run is insufficient. Subsequently, the algorithm's performance is examined in terms of efficiency and effectiveness, including a wide range of problem types such as discrete and continuous problems, as well as single and multiple objectives. The primary focus of the efficiency measure for single-objective problems is the rate of problem-solving, which encompasses several factors such as complexity, computational cost, diversity of search operators, convergence rate, and statistical measurements like ordered alternatives and cumulative distribution. Discrete problems with a single objective fall under a similar area. Running time is one of the important metrics, particularly for combinatorial problems.

In [20] the authors present “Performance of Six Metaheuristic Algorithms for Multi-Objective Optimization of Nonlinear Inelastic Steel Trusses”. The work suggests new strategy for solving the steel trusses using direct analysis by using multi-objective optimization. The overall weight of the structure and its interstory drift or displacements were two competing goals that had been evaluated by nonlinear inelastic and nonlinear elastic models, respectively, to determine the limitations related to strength and serviceability load combinations. The six popular meta-heuristic algorithms were used to solve the developed MOO problem: generalized differential evolution (GDE3), PSO-based MOO using crowding, mutation, and ε-dominance (OMOPSO), non-dominated sorting genetic algorithm-II (NSGA-II), NSGA-III, improving the strength Pareto evolutionary algorithm (SPEA2), and multi-objective evolutionary algorithm based on decomposition (MOEA/D). A planar 10 bar truss, a spatial 72 bar truss, a planar 47 bar power-line truss, and a planar 113-bar truss bridge were the four truss structures that were examined. A planar 10 bar truss, a spatial 72 bar truss, a planar 47 bar power-line truss, and a planar 113 bar truss bridge were the four truss structures that were examined.

The numerical outcomes demonstrated an inverse proportion and nonlinear relationship between the two objectives. All six algorithms were also effective in locating feasible, optimal solutions. While MOEA/D and NSGA-II appeared to be more adept at finding anchor points as well as Pareto, no algorithm was shown to perform better than the others. Additionally more stable and giving a better solution spread was MOEA/D . OMOPSO was also good at solution spread, but its stability was worse than MOEA/D. NSGA-III was less efficient at finding anchor points, although it can effectively search for Pareto points [20].

A comparative study is presented in this paper for both GWO and MFDS on performance on 10 test functions.

Characteristics of Grey Wolf Optimization (GWO) algorithm

The GWO is a population-based, nature-inspired optimization technique that uses a leader-follower structure to mimic the cooperative hunting behavior of grey wolves. It is known for their ability to balance exploration and has been applied to a wide range of optimization problems in various domains [13], [14].

GWO balances exploration (searching for new and potentially better solutions) and exploitation (refining known good solutions). This balance is achieved through the leader-follower structure and position updates [13].

It emulates how wolves work together to locate and capture prey and speed the convergence, with alpha, beta, and delta wolves leading the pack. Like many other metaheuristic algorithms, GWO operates on a population of potential solutions (wolves). The quality of these solutions is assessed using a fitness function [13]. Also, GWO can be applied to many different kinds of problems in optimization, including as discrete, mixed-variable, and continuous problems. It can be adapted to various domains.

Like many optimization algorithms, GWO may require parameter tuning to achieve optimal performance for specific problems. Parameters may include the population size, the exploration-exploitation balance, and the rate of convergence. The convergence speed of GWO can vary depending on the problem and parameter settings. It may converge relatively quickly to a good solution in many cases [11]. On the other hand GWO can be parallelized, allowing multiple instances of the algorithm to run concurrently, potentially speeding up the optimization process.

The Grey Wolf Optimization algorithm

The Grey Wolf Optimization algorithm aims to simulate the social behavior of grey wolves in a pack to optimize a given objective function. It leverages the hierarchy within wolf packs to guide the exploration of the solution space, with the Alpha, Beta, and Delta wolves playing distinct roles in this exploration. The algorithm's effectiveness often depends on parameter tuning and its application to specific optimization problems [11], [13], [14].

The GWO was initially presented in 2014. The GWO algorithm was modeled after the social intelligence taken by behavior of grey wolves in terms of their pack leadership and in-wild hunting practices. A same social hierarchy controls the balance of power and dominance within each grey wolf pack (see Figure 1). Alpha is the strongest wolf in the pack and leads the others in feeding, migrating, and hunting. The strongest of the β wolves becomes leadership of the pack in the event that the α wolf is absent due to illness or another reason, or in the event that an α wolf passes away [21], [22].

Figure 1 illustrates how β and δ's combined dominance and power are less than α and β's. The GWO algorithm is based on this type of social intelligence. It draws inspiration from both the hunting strategy and the hierarchical behavior of grey wolves. Grey wolves have an effective set of movements when hunting in packs: pursuing, encircling, bothering, and attacking. They can now hunt big creatures as prey because of this [21].

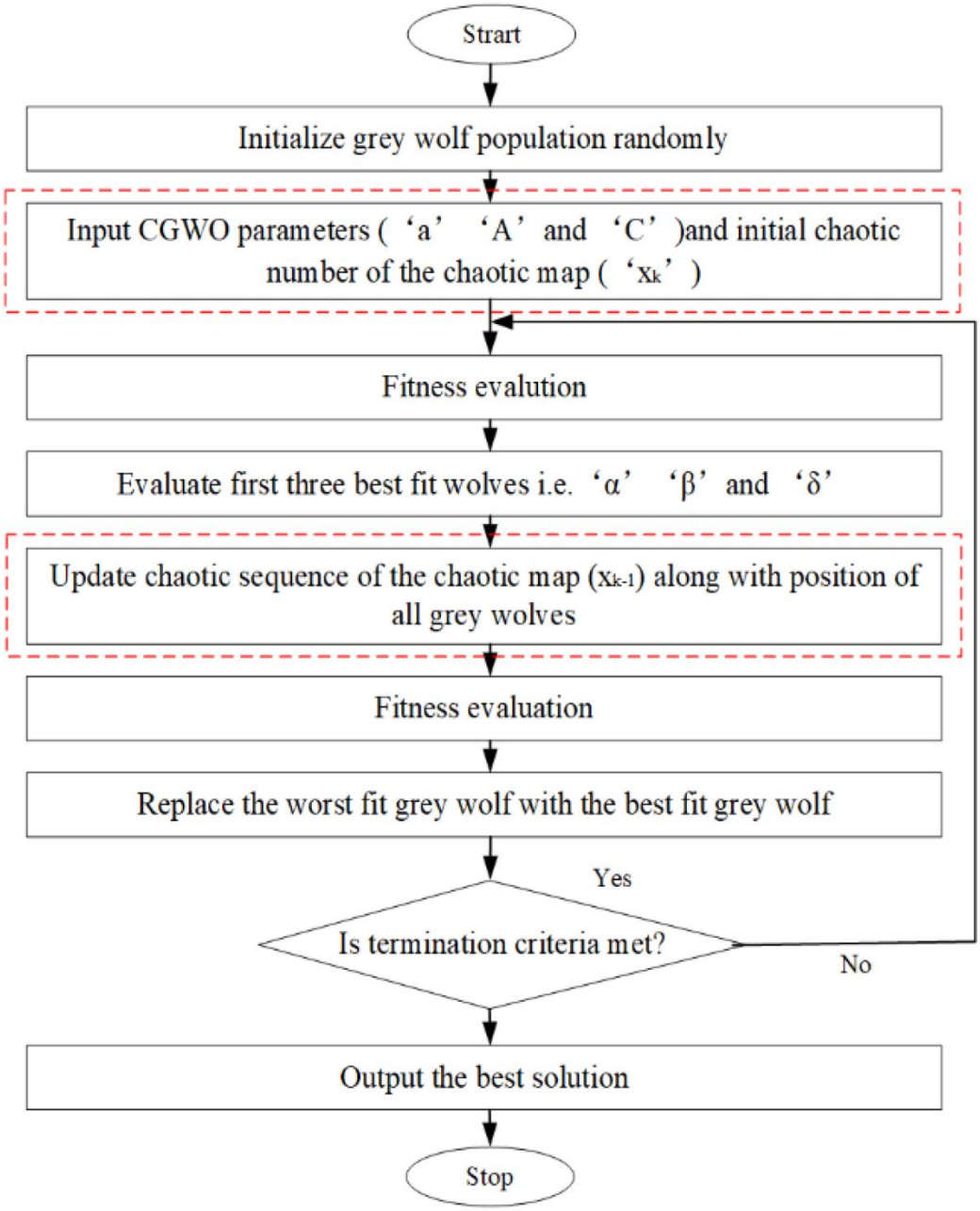

Figure 1: GWO Flowchart.

Characteristics of Multi-free Dynamic Schema (MFDS) algorithm

The MFDS algorithm presents a set of operators and techniques for solving optimization problems. Its use of dynamic schema operators, adaptive grouping, and randomization can potentially lead to efficient exploration of the solution space. This algorithm has some characteristic that improve the way to found optimal solution [15]. In following part a short description of these characteristic:

1. MFDS uses a set of operations, such as random chromosomal generation, dynamic dissimilarity, similarity, dynamic schema, and free dynamic schema. This diverse operators allows the algorithm to explore different area of the solution space, potentially leading to improved convergence and solution quality.
2. The concept of dynamic schema operator is an interesting aspect of the algorithm. This operator aims to fixed high significant bits of each variable in the best chromosome and repeat it across the population, then put 0 or 1 randomly in rest low significant bits. This can lead to rapid convergence to promising regions in the solution space.

3. The algorithm introduces random aspect in part of population in each iteration. This randomization can enhance diversity within the population, preventing premature convergence and promoting exploration of the solution search space.
4. Adaptive Grouping: The algorithm divides the population into different groups, each group has specific operator. This adaptive grouping is improving the efficiency of the optimization process.

(MFDS) algorithm

The MFDS algorithm is designed for complex optimization problems with constraints and leverages a combination of schema operators, dissimilarity operators, and free dynamic schema operators to explore the solution space effectively. It balances the exploitation of promising regions and the exploration of unexplored areas to find optimal solutions [15].

The MFDS algorithm aims to find the optimal solution to a given optimization problem where a functionf: $\mathbb{R}^n \longrightarrow \mathbb{R}$

$$\text{minimize}|\text{maximize } f(x_1, \dots, x_n) \text{ subject to}$$
$$x_i \in [a_i, b_i], i = 1, \dots, n$$

where f: $\mathbb{R}^n \longrightarrow \mathbb{R}$ is a given function.

Algorithm Steps:

1. Produce 2 M chromosomes, each one chromosome representing one point. Separate chromosomes into two population (P0 and P1). P0 has four groups (G1, ..., G4), P1 has eight groups (G5,..., G12).
2. Compute the fitness function values (f) for every chromosome in two populations (P0) and (P1).
3. Sorting of the chromosomes is done according to the descending (for maximization) or ascending (for minimization) values of the fitness function.
4. Replace the original chromosomes in groups G5 and G6 with the best 40% of the P0 chromosomes. To replace the original chromosomes in the first half of P0, duplicate the first chromosome (Ch1) C times and insert it in C randomly selected locations. In this case, C represents a portion of the population.
5. Create a one chromosome to represent the dynamic schema operator, from chromosomes A = Ch1 and B = ChM/4 in the populations (P0). Duplicate this schema M/4 times and add it to (G3).
6. Apply the dynamic dissimilarity and similarity operators, respectively, to groups (G1) and (G2). The dissimilarity and dynamic dissimilarity operators should be felt by groups (G5) and (G6), respectively.
7. To create six groups, G7 through G12, apply the free dynamic schema operator six times. A chromosome is selected at random from the first quarter of the P0 solutions for every group. In each group, at random, place 0s or 1s in the places indicated by asterisks (*).
8. In populations (P0) and (P1), every chromosome generated in Steps 4 to 8 replaces the previous ones in positions ranging from 2 to 2M. Next, create the group's (G4) chromosomes at random.
9. Return to Step 2 and keep going until the stopping requirement is met.

Experimental results

Ten test functions are used to check the performance of two algorithms MFDS and GWO on these ten test function. Table 1 show the 10 functions that are used in our study with their optimum solutions. In this work a run time in seconds, mean number of iterations to reach the optimum solutions, the mean number of iterations are used. The results was taken from 50 runs on each function for two algorithms MFDS and GWO, the maximum number of iteration was 2500 iteration for each algorithm. Table 2 show the results of ten functions and the names of optimization functions used in the evaluation with threshold value. Also, The success rate of GWO and MFDS for each function.

For most functions, both the MFDS and GWO algorithms achieve a 100% success rate, indicating that they were able to find a valid solution in all 50 runs. But some functions have a slightly lower success rate, like the Goldstein-Price function, by using GWO which get a 66% success rate, where MFDS has get 100%, see Table 2.

The performance metrics was including mean number of iterations (or average time) and standard deviation, where the (stander deviation and mean number of iterations) in MFDS almost were less than GWO.

Overall, Table 2 provides valuable insights into the comparative performance of the MFDS and GWO across a range of optimization problems. The choice between these algorithms for a particular problem should depend on the problem's characteristics and the trade-offs between convergence speed and solution quality.

Table 1: show the ten test functions [23]

Name	Function	D	Range	global optimum value
Martin and Gaddy function	$f(x_1, x_2) = (x_1 - x_2)^2 \cdot ((x_1 + x_2 - 10)/3)^2$	2	$x_1, x_2 \in [0, 10]$	$f(5,5) = 0$
Easom function	$f(x, y) = -\cos(x).\cos(y).\exp(-(x - \pi)^2 + (y - \pi)^2))$	2	$x_1, x_2 = [-100,100]$	$f(\pi, \pi) = -1$
Matyas function	$f(x, y) = 0.26(x^2 + y^2 - 0.48xy)$	2	$x_1, x_2 \in [-10, 10]$	$f(0,0) = 0$
Beale's function	$f(x, y) = (1.5 - x - xy)^2 + (2.25 - x + x\,y^2)^2 + (2.625 - x - xy^2)^2$	2	$x_1, x_2 \in [-4.5, 4.5]$	$f(3,0.5) = 0$
Booth's function	$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$	2	$x_1, x_2 \in [-10, 10]$	$f(1,3) = 0$
Goldstein–Price function	$f(x, y) = (1 + (x + y + 1)^2 (19 - 14x + 3x^2 - 14y + 6xy + 3y^2)) * (30 + (2x - 3y)^2 (18 - 32x + 12x^2 + 48y - 36xy + 27y^2))$	2	$x_1, x_2 \in [-2, 2]$	$f(0, -1) = 3$
Schaffer N.2 function	$f(x, y) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	2	$x_1, x_2 \in [-100, 100]$	$f(0, 0) = 0$
Schwefel's function	$f(x) = 418.9829 * n + \sum_{i=1}^{n} -x_i \,.\sin\left(\sqrt{\lvert x_i \rvert}\right)$	2	$x_1, x_2 \in [-500, 500]$	$f(420.9687, 420.9687) = 0$
Drop-wave function	$(x_1, x_2) = -\frac{1 + \cos\left(12\sqrt{x_1^2 - x_2^2}\right)}{0.5(x_1^2 + x_2^2) + 2}$	2	$x_1, x_2 \in [-5.12, 5.12]$	$f(0,0) = -1$
Levy N. 13 function	$f(x_1, x_2) = \sin^2(3\pi x_1) + (x_1 - 1)^2[1 + \sin(3\pi x_2)] + (x_2 - 1)^2[1 + \sin(2\pi x_2)]$	2	$x_1, x_2 \in [-10, 10]$	$f(1, 1) = 0$

Table 2: Results of MFDS and GWO algorithms

MFDS algorithm							GWO algorithm				
Name of Function	**Threshold**	**Minimum number of iter. with Minimum time in seconds**	**Maximum number of iter. with Maximum time in seconds**	**Mean number of iter. for all successful runs with Average time**	**Stander Deviation of mean number. of iter.**	**Success. rate of MFDS**	**Minimum number of iter. with Minimum time in seconds**	**Maximum number of iter. with Maximum time in seconds**	**Mean number of iter. for all successful runs with Average time**	**Stander Deviation of mean number. of iter.**	**Success. rate of GWO**
Martin and Gaddy	0.001	2 0.006	16 0.019	**6** **0.011**	2	100%	196 1.632622	724 2.05501	505 1.759751	200	100%
Easom	0.001	6 0.018	167 0.253	58 **0.087**	42.5	100%	42 1.576	257 1.877	143 1.683	67	100%
Matyas	0.001	3 0.008	15 0.017	5 0.011	1.5	100%	5 1.665	7 1.835	7 1.754	0.65	100%
Beale's	0.001	4 0.010	34 0.037	7 0.014	3.7	100%	42 1.569	487 1.784	197 1.675	111	100%
Booth's	0.001	4 0.010	32 0.043	**10** **0.018**	4.5	100%	131 2.259	721 2.685	460 2.470	167	100%
Goldstein–Price	0.001	7 0.018	65 0.079	**21** **0.034**	10.4	100%	51 1.580	212 1.855	122 1.682	46	**66%**
Schaffer N.2	0.001	4 0.006	26 0.043	11 0.019	6.3	100%	7 1.413682	11 1.6397	10 1.507925	1.35	100%
Schwefel's	0.001	4 0.008	70 0.110	**31** **0.047**	14.1	100%	2193 2.39948	3000 2.813568	2378 2.52597	204	100%
Drop-wave	0.001	5 0.019	69 0.1005	30 0.046	16.6	100%	12 1.629516	26 1.793494	18 1.718351	4.5	100%
Levy N. 13	0.001	5 0.013	37 0.049	**11** **0.020**	6	100%	196 1.6546	1318 2.1311	730 1.896	371	100%

Conclusions

In this comparative study between the Multi-Free Dynamic Schema (MFDS) algorithm and the Grey Wolf Optimization (GWO) algorithm, we evaluated their performance on a set of 10 test functions. The algorithms were assessed based on runtime, the mean number of iterations needed to get the optimal results, both MFDS and GWO have got 100% success rate in most test functions but with different runtime and number of iterations. The MFDS was faster than GWO in runtime and number of iterations, we found valid solutions in all 50 runs. However, there were a few exceptions, such as the Goldstein-Price function, where GWO achieved a 66% success rate, while MFDS maintained a 100% success rate.

References

[1] I. Younas, "Using Genetic Algorithms for Large Scale Optimization of Assignment , Planning and Rescheduling Problems," Doctoral Thesis in Electronics and Computer Systems Stockholm, Sweden, 2014.

[2] S. Mirjalili, "Knowledge-Based Systems Moth-flame optimization algorithm : A novel nature-inspired heuristic paradigm," Knowledge-Based Syst., vol. 89, pp. 228–249, 2015, doi: 10.1016/j.knosys.2015.07.006.

[3] A. W. Mohamed, K. M. Sallam, P. Agrawal, A. A. Hadi, and A. K. Mohamed, "Evaluating the performance of meta-heuristic algorithms on CEC 2021 benchmark problems," Neural Comput. Appl., vol. 35, no. 2, pp. 1493–1517, 2023, doi: 10.1007/s00521-022-07788-z.

[4] A. Forestiero, "Metaheuristic algorithm for anomaly detection in Internet of Things leveraging on a neural-driven multiagent system," Knowledge-Based Syst., vol. 228, p. 107241, 2021, doi: 10.1016/j.knosys.2021.107241.

[5] O. Mersmann, M. Preuss, H. Trautmann, B. Bischl, and C. Weihs, "Analyzing the BBOB results by means of benchmarking concepts," Evol. Comput., vol. 23, no. 1, pp. 161–185, 2015, doi: 10.1162/EVCO_a_00134.

[6] S. Mirjalili, "Knowledge-Based Systems SCA : A Sine Cosine Algorithm for solving optimization problems," vol. 000, pp. 1–14, 2016, doi: 10.1016/j.knosys.2015.12.022.

[7] G. P. Rajappa, "Solving Combinatorial Optimization Problems Using Genetic Algorithms and Ant Colony Optimization," 2012.

[8] M. Gendreau and J.-Y. Potvin, Handbook of Metaheuristics, Springer N., vol. 146. Springer New York Dordrecht Heidelberg London, 2010. doi: 10.1007/978-1-4614-1900-6.

[9] O. Design, Numerical Methods for Constrained Optimum Design. 2017. doi: 10.1016/B978-0-12-800806-5/00012-3.

[10] S. N. Sivanandam and S. N. Deepa, Introduction to Genetic Algorithms. Springer-Verlag Berlin Heidelberg, 2008.

[11] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," Adv. Eng. Softw., vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[12] I. Montalvo, J. Izquierdo, R. Pérez-García, and M. Herrera, "Improved performance of PSO with self-adaptive parameters for computing the optimal design of Water Supply Systems," Eng. Appl. Artif. Intell. Elsevier, vol. 23, no. 5, pp. 727–735, 2010, doi: 10.1016/j.engappai.2010.01.015.

[13] H. Faris, I. Aljarah, M. A. Al-Betar, and S. Mirjalili, "Grey wolf optimizer: a review of recent variants and applications," Neural Comput. Appl., vol. 30, no. 2, pp. 413–435, 2018, doi: 10.1007/s00521-017-3272-5.

[14] M. Karakoyun, A. Ozkis, and H. Kodaz, "A new algorithm based on gray wolf optimizer and shuffled frog leaping algorithm to solve the multi-objective optimization problems," Appl. Soft Comput. J., vol. 96, p. 106560, 2020, doi: 10.1016/j.asoc.2020.106560.

[15] R. Al-Jawadi, M. Studniarski, and A. Younus, "An Optimization Algorithm Based on Multi-free Dynamic Schema of Chromosomes," Adv. Intell. Syst. Comput., vol. 1051, pp. 146–156, 2020, doi: 10.1007/978-3-030-30604-5_13.

[16] S. Shabir, "A Comparative Study of Genetic Algorithm and the Particle Swarm Optimization," vol. 9, no. 2, pp. 215–223, 2016.

[17] D. S. Sanches and C. Music, "Comparative study of Genetic Algorithm and Ant Colony Optimization algorithm performances for the task of guitar tablature transcription," 2015, doi: 10.1109/BRACIS.2015.46.

[18] A. Gupta and S. Srivastava, "Comparative Analysis of Ant Colony and Particle Swarm Optimization Algorithms for Distance Optimization," Procedia Comput. Sci., vol. 173, no. 2019, pp. 245–253, 2020, doi: 10.1016/j.procs.2020.06.029.

[19] A. H. Halim, I. Ismail, and S. Das, Performance assessment of the metaheuristic optimization algorithms: an exhaustive review, vol. 54, no. 3. Springer Netherlands, 2021. doi: 10.1007/s10462-020-09906-6.

[20] M. Optimization, N. Inelastic, and S. Trusses, "Performance of Six Metaheuristic Algorithms for Multi-Objective Optimization of Nonlinear Inelastic Steel Trusses," vol. 13, no. 4, pp. 1–26, 2023.

[21] M. Zhang et al., "The Strain Distribution Reconstructions Using GWO Algorithm and Verification by FBG Experimental Data," Appl. Sci., vol. 13, no. 3, 2023, doi: 10.3390/app13031259.

[22] G. Negi, A. Kumar, S. Pant, and M. Ram, "GWO: a review and applications," Int. J. Syst. Assur. Eng. Manag., vol. 12, no. 1, 2021, doi: 10.1007/s13198-020-00995-8.

[23] R. Al-jawadi, "New Evolutionary Optimization Algorithms Using Similarities and Dissimilarities in Binary Strings," ph.D. thesis, Univ. Warsaw, 2018.