Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.046606

ARTICLE

MOALG: A Metaheuristic Hybrid of Multi-Objective Ant Lion Optimizer and Genetic Algorithm for Solving Design Problems

Rashmi Sharma[1], **Ashok Pal**[1], **Nitin Mittal**[2], **Lalit Kumar**[2], **Sreypov Van**[3], **Yunyoung Nam**[3,*] **and Mohamed Abouhawwash**[4,5]

[1]Department of Mathematics, Chandigarh University, Gharuan, Mohali, 140413, India

[2]Deaprtment of Industry 4.0, Shri Vishwakarma Skill University, Palwal, 121102, India

[3]Department of ICT Convergence, Soonchunhyang University, Asan, 31538, Korea

[4]Department of Computational Mathematics, Science and Engineering (CMSE), College of Engineering, Michigan State University, East Lansing, MI 48824, USA

[5]Department of Mathematics, Faculty of Science, Mansoura University, Mansoura, 35516, Egypt

*Corresponding Author: Yunyoung Nam. Email: ynam@sch.ac.kr

Received: 08 October 2023 Accepted: 15 January 2024 Published: 26 March 2024

ABSTRACT

This study proposes a hybridization of two efficient algorithm's Multi-objective Ant Lion Optimizer Algorithm (MOALO) which is a multi-objective enhanced version of the Ant Lion Optimizer Algorithm (ALO) and the Genetic Algorithm (GA). MOALO version has been employed to address those problems containing many objectives and an archive has been employed for retaining the non-dominated solutions. The uniqueness of the hybrid is that the operators like mutation and crossover of GA are employed in the archive to update the solutions and later those solutions go through the process of MOALO. A first-time hybrid of these algorithms is employed to solve multi-objective problems. The hybrid algorithm overcomes the limitation of ALO of getting caught in the local optimum and the requirement of more computational effort to converge GA. To evaluate the hybridized algorithm's performance, a set of constrained, unconstrained test problems and engineering design problems were employed and compared with five well-known computational algorithms-MOALO, Multi-objective Crystal Structure Algorithm (MOCryStAl), Multi-objective Particle Swarm Optimization (MOPSO), Multi-objective Multiverse Optimization Algorithm (MOMVO), Multi-objective Salp Swarm Algorithm (MSSA). The outcomes of five performance metrics are statistically analyzed and the most efficient Pareto fronts comparison has been obtained. The proposed hybrid surpasses MOALO based on the results of hypervolume (HV), Spread, and Spacing. So primary objective of developing this hybrid approach has been achieved successfully. The proposed approach demonstrates superior performance on the test functions, showcasing robust convergence and comprehensive coverage that surpasses other existing algorithms.

KEYWORDS

Multi-objective optimization; genetic algorithm; ant lion optimizer; metaheuristic

1 Introduction

In recent years, to solve difficult problems, computers have recently gained a lot of popularity across a variety of industries. The use of computers to solve issues and develop systems is the focus of the field of computer-aided design. In the prior, direct human engagement was needed for solving complex problems. More time and resources were also needed. Similarly, there has been a significant advancement in the field of optimization.

Gaining the most from the other options is what is meant by optimization. Based on the number of objectives taken, optimization problems are mainly classified as Single objective optimization problems (SOOPs) and Multi-objective optimization problems (MOOPs). SOOPs are the problems having one objective and MOOPs have more than one objective. In SOOPs, a single solution, however for MOOPs a set of trade-off solutions called Pareto-optimal solutions is obtained. Edgeworth and Pareto developed the optimality notion for MOOPs in their books in 1881 and 1906, respectively [1,2]. Finding solutions to problems with many objectives is the major goal of this effort, which is MOO. The key problem in MOO is dealing with many objectives, which are generally in conflict with one another. Traditional optimization approaches are based on gradient-based calculation. So, to solve real-world complex problems it becomes difficult [3]. To overcome this problem metaheuristic techniques came into existence and are problem-independent and capable of finding near-optimal solutions [4]. In those stochastic optimization techniques, they begin the optimization procedure by improving a group of randomly selected solutions to a given problem over a predetermined number of steps. Metaheuristic algorithms are categorized based on the natural phenomenon they mimic such as evolutionary algorithms, and swarm intelligence-based algorithms. The fundamental idea behind the application of evolutionary algorithms is survival of the fittest means weak must phased out. The most prominent are Genetic Algorithm (GA) [5], Differential Evolution (DE) [6], and Non-dominated Sorting Genetic Algorithm (NSGA) [7]. Swarm intelligence-based algorithms draw inspiration from the social behavior seen across living organisms. Some of the algorithms can be seen as Grey Wolf Optimizer (GWO) [8], Particle Swarm Optimization (PSO) [9], Artificial Hummingbird Algorithm (AHA) [10], Artificial Bee Colony (ABC) [11], Ant Colony Optimization (ACO) [12], Cuckoo Search (CS) Algorithm [13], ALO [14], Monarch Butterfly Optimization (MBO) [15], Firefly Algorithm (FA) [16]. Some other algorithms are the Moth Search Algorithm (MSA) [17], Thermal Exchange Optimization (TEO) [18], Gravitational Search Algorithm (GSA) [19], and Binary Waterwheel Plant Optimization Algorithm [20]. To address the shortcomings of individual algorithms and enhance their effectiveness, hybrid algorithms are gaining more prominence these days. Some of these algorithms incorporate different areas such as structural optimization [21], combinatorial optimization [22], and energy and renewable energy [23].

The two key components of Metaheuristic algorithms are exploration and exploitation. The perfect balance between them is the performance assessment of any algorithm in solving given MOOPs. Exploration refers to the search for the unexplored area and exploitation refers to the search for the immediate area to deliver an accurate search and convergence.

No Free Lunch theorem (NFL) for optimization encourages experts to develop the algorithms and enhance the ones that are already in use [24]. It asserts that no algorithm is superior to another for all OPs. In response to this, numerous novel algorithms and hybrid ones have been evolved to address challenging OPs of the real world.

This paper proposes the hybrid of MOALO and GA for solving MOOPs. As best, this is the first comprehensive hybrid between MOALO and GA employed to solve MOOPs and design problems. According to the NFL, one single algorithm cannot have the expertise to solve all the existing problems.

Real-world problems are more complex nowadays and also have many objectives to satisfy, so to solve those problems more and more algorithms are needed. For handling challenging real-world situations, metaheuristic and hybrid metaheuristic algorithms are preferable. Inspired by this, two metaheuristics have been chosen MOALO and GA, this hybrid will try to overcome the limitation of MOALO of efficiency and accuracy and GA's computational effort to converge.

This work's primary contributions can be specified as follows:

- A hybridization strategy that combines MOALO and GA has been put forward as a means of tackling MOOPs and mitigating the limitations of each method.
- Next, the use of performance metrics like generational distance (GD), inverted generational distance (IGD), spacing, spread, and hypervolume (HV) has been statistically implemented to test the overall performance in comparison to other existing algorithms.
- Hybrid has been subjected to testing on constrained, unconstrained benchmark problems and design problems.

The remaining portion of the paper is presented as Section 2 covers the literature. MOALO, GA, and the proposed hybrid MOALG are included in Sections 3 and 4. Section 5 concludes the evaluation metrics involved. Section 6 and Section 7 display the outcomes of the MOALG algorithm. Section 8 solves the engineering design problems and discussion of the outcomes. Section 9 serves as the concluding section, providing a comprehensive summary of the work and exploring potential future research areas.

2 Related Work

2.1 The Beginnings of Multi-Objective Optimization

A general MOOP [25]:

$$\text{Minimize: } \mathrm{f}_z(\mathrm{o}),\ \mathrm{z} = 1, 2, \ldots \mathrm{Z} \tag{1}$$

Subject to:

$h_m(o) \geq 0, m = 1, 2, \ \ldots \mathrm{D}$

$k_n(o) = 0, \mathrm{n} = 1, 2, \ \ldots \mathrm{J}$

$Lb_k \leq u_k \leq Ub_k, \mathrm{k} = 1, 2, \ \ldots \mathrm{v}$

where Z-Number of objectives, D-Inequality constraints, k_n-n-th equality constraints, h_m-m-th inequality, J-Equality constraints, V-Number of variables and $[Lb_k, Ub_k]$-k-th variable boundaries.

2.2 Existing Metaheuristic Multi-Objective Optimization Algorithms

Different algorithms have been employed to resolve MOOPs including the Multi-objective Bat algorithm [26], the MO version of the bat algorithm that is inspired by bat echolocation. MO Bee Algorithm [27] is inspired by honeybees well-known for their efficient foraging behavior, in which they explore their surroundings in search of the best resources to exploit. Inspired by forensic investigation techniques another well-known algorithm is MO Forensic-based Investigation Algorithm (MOFBI) [28]. Mrijalili's and other's hard work gave us an algorithm that follows the hunting behavior of grey wolves (MOGWO) [29]. MOPSO, MO version of PSO [30]. Some of the recently proposed algorithms (see Table 1) are the MOCryStAl [31], MOMVO [32], and MO Thermal Exchange Optimization (MOTEO) algorithm [33].

Some of the hybrid algorithms are as follows:

Table 1: Some known hybrid algorithms

Year	Algorithms
2014	Particle Swallow Swarm Optimization (HPSSO) [34]
2015	PSO and Estimation of Distribution Algorithm (EDA) [35]
2017	Backtracking Search and Genetic Algorithm (HMOBSA) [36]
2020	Gravitational Search Algorithm and BAT Algorithm (MOGSABAT) [37]
2020	Spotted Hyena Optimizer and Emperor Penguin Optimizer (MOSHEPO) [38]
2021	ALO Algorithm and Sine Cosine Algorithm (SCA) [39]
2022	GWO and Support Vector Machine (SVM) [40]
2022	ABC and DE (HABC-DE) [41]
2023	Hybrid Multi-Objective Evolutionary Algorithm (MOEA) [42]

3 Background of MOALO and GA

3.1 Multi-Objective Ant Lion Algorithm (MOALO)

Seyedali Mirjalili established the ant lion algorithm in 2015. The ALO algorithm is inspired by antlion and ant interaction in the trap. To resolve MOOPs, MOALO [43] is an enhanced version of ALO. Similar to ALO, they have antlions and ants as population.

The fundamental ALO processes to modify ants and antlions position and finally calculate the global optimum for a specific problem of optimization are:

1. Ants are the prime agents of search in ALO. So, in the initialization process, random values are assigned for the ants set.

2. At each iteration each ant's fitness value is measured by the use of an objective function.

3. It is assumed that antlions are at one place and ants move randomly around antlions and over the search space.

4. Antlion populations are never determined. In the first iteration, antlions are expected to be where the ants are, but as the ants improve, they move to new positions in successive iterations.

5. Each ant is assigned an antlion and the position gets updated as the ant becomes fitter.

6. The best antlion obtained is defined as elite and affects the ant's movement.

7. By the successive iterations, any antlion that outperforms the elite will get replaced by the elite.

8. The steps numbered 2 and 7 are executed iteratively multiple times until a certain condition is met, which serves as the termination criterion for the algorithm.

9. For the global optimum value the elite antlion's position and fitness value are considered.

The changes made to MOALO set it apart from ALO since they used archives to store Pareto optimum solutions. To solve this issue of identifying Pareto optimal solutions with large diversity, "leader selection and archive maintenance" have been used. Concerning assessing the distribution of solutions within the archive, the niching technique was utilized. A further modification was the use of the roulette wheel selection and elitism, which was employed in selecting a non-dominated solution.

3.2 *Genetic Algorithm*

Genetic algorithm (GA) was first discovered by Holland and extensions by Goldberg in the 1960s influenced by Darwin's evolutionary theory [5]. Firstly, a random set of solutions is generated which are expressed as chromosomes defined as population. After that fitness function of each chromosome is calculated. Solutions that are best in terms of fitness function are selected to generate the next population and undergo "crossover and mutation". This procedure is executed till the defined condition is satisfied.

Basic operators involved in GA for finding the optimal solutions are defined as:

1. Selection: For the next generation to benefit from good individuals, the selection operator must make sure that they do so. So, the best solution should move to the reproduction process for the next generation, and the bad solution left behind. Selection operators are designated as Tournament Selection, Random Selection, Rank-Based Selection, Elitist Selection, Roulette wheel selection, and Boltzmann Selection.

2. Crossover: Two or more parents' genetic material are combined to form one or more new individuals. Single-point, Uniform, Two-point, and Arithmetical crossover are the defined operators.

3. Mutation: In GA, maintaining genetic diversity is crucial for the success of the process of optimization. To preserve genetic diversity, one common strategy is to alter "one or more gene values" of the population to create a new individual. This alteration can be done in various ways, such as by randomly selecting a gene value or by using a mutation operator to change the gene value based on a specific probability distribution. The mutation is normally applied at low probability. Lip bit, Non-uniform, Uniform, and Gaussian are mutation operators defined.

4 Hybrid Multi-Objective Ant Lion Optimizer Algorithm with Genetic Algorithm (MOALG)

MOALG is developed in this paper which is a hybridised algorithm. The population of ants and antlions is generated randomly in the proposed hybrid like ALO [14]. Following the identification of each ant and antlion's fitness value, random walks in the search space's boundaries and the area of antlions are conducted like MOALO [43]. In MOALG, after the first iteration archive is updated by utilizing the operator's selection, crossover, and mutation to improve the solutions. After that the solutions again go through the ranking process and the archive is updated. Then, antlion selection, elitism, random walk of ants, and other steps of MOALO are completed.

4.1 *Steps of Proposed MOALG*

This section involves the steps utilized in the proposed hybrid MOALG. The following steps are:

1. Initialization

The population of ant and antlion is generated by

$$x = rand\,(Ub - Lb) + Lb \tag{2}$$

2. Random walks

The ant's random walk (RW) is entitled as:

$$\mathrm{X}\,(\mathrm{t}) = \left[0, cumsum\left(2\mathrm{r}\,(\mathrm{t}_1\right), \ldots, cumsum\left(2\mathrm{r}\left(\mathrm{t}_p\right) - 1\right)\right] \tag{3}$$

where iteration is illustrated by t as RW steps

The stochastic function is illustrated as r(t)

r(t) is 1 *if rand* $>$ 0.5, otherwise 0, if not. "rand" is used to generate a random number spread uniformly over the range [0,1]. Since Eq. (2) should not be the only equation for position updating of ants and every search space should have a boundary. So RW should be normalized:

$$\mathrm{X}_\mathrm{i}^\mathrm{t} = \frac{\left(X_i^t - a_i\right) \times \left(d_i^t - c_i^t\right)}{\left(b_i - a_i\right)} + c_i^t \tag{4}$$

where at the iteration, d_i^t, c_i^t is the maximum and minimum of RW, of ith variable, respectively.

3. Trapping, building trap by antlion

By modifying the RW of ants around antlions, ALO facilitates the entrapment of ants in antlion pits. The equation proposed for this is as follows:

$$c_i^t = Antlion_j^t - c^t \tag{5}$$

$$d_i^t = Antlion_j^t - d^t \tag{6}$$

$Antlion_j^t$ demonstrates as the selected antlion's position jth antlion. To imitate this, ALO employs a Roulette wheel operator. More ants are drawn to the fitter antlions due to the roulette wheel.

4. Sliding ants towards antlion

To simulate the movement of ants towards antlions in a simulation using random walks, it is often necessary to decrease the boundaries of the random walks adaptively. This adaptive adjustment of the boundaries is done to simulate the ants sliding over to the antlion.

$$c^t = \frac{c^t}{I} \tag{7}$$

$$d^t = \frac{d^t}{I} \tag{8}$$

where

$$\mathrm{I} = 1 + 10^w \frac{t}{T} \tag{9}$$

I is a ratio. t, T are the current and maximum number of iterations, respectively.

The parameter "w" in the Eq. (9) for "I" enables precise adjustment of the exploitation accuracy level as shown in Table 2.

Table 2: Constant w depends on the current iteration

w	t
2	>10% T
3	>50% T
4	>75% T
5	>90% T
6	>95% T

5. Catching prey and re-building the pit

The next step is to update the position after catching the ants. The equation below is expressed in this context:

$$Antlion_j^t = Ant_i^t \text{ if } f\left(Ant_i^t\right) > f\left(Antlion_j^t\right) \tag{10}$$

where t is determined by the current iteration and ith and jth are ant and selected antlion's position, respectively.

6. Elitism

The final phase is elitism, where the antlion having the fittest value is saved during optimization at each iteration. The following process is shown:

$$Ant_i^t = \frac{\left(R_A^t + R_E^t\right)}{2} \tag{11}$$

R_A^t and R_E^t-RW around the antlion selected by "roulette wheel" and elite, respectively.

7. Use of leader selection

The solutions with the fewer neighboring solutions are chosen for the antlions.

$$P_i = \frac{c}{N_i} \tag{12}$$

8. Archive maintenance

The solution with the densely packed neighboring is discarded when the archive is full to free up space for new solutions.

$$P_i = \frac{N_i}{c} \tag{13}$$

where c > 1 and is a constant. At ith solution, N_i is a variety of neighboring solutions.

After finding the fitness of each ant and elite. In MOALG, the Archive has been updated by applying the ranking process. From the archive, two solutions are selected as parent 1 and parent 2 by the roulette wheel selection. The new generation (offspring) is generated by applying a one-point crossover operator with a certain probability P_c as 0.9 [44].

After that, elements of the new offspring mutate at a certain probability P_m as 0.01. The ranking process is again applied in the archive which is followed by the selection of the elite. After that, all the steps are followed same as of MOALO.

A random walk around antlion includes steps Eqs. (3) and (4) and which includes antlion's pits entrapment Eqs. (5) and (6). After that, the building trap and ants falling towards the antlion include Eqs. (7) and (8). The last step includes a random walk around the antlion and elite in which the position of the ant is updated including Eq. (11). For storing the non-dominated solutions, an archiving approach has been employed and for finding the high diversity among the Pareto optimal solutions niching have been employed. This antlion is composed of individuals from the least populated neighboring and the solutions with the densely packed neighboring are discarded when the archive is full to free up space for new solutions Eqs. (12) and (13). The computational complexity is entitled $O(mn^2)$, where m and n are some objectives and individuals, respectively. The Pseudocode of MOALG is presented in Algorithm 1 and the flowchart is shown in Fig. 1.

Algorithm 1: Pseudo-code of MOALG

```
 Begin
1. Create initial population of ants and antlions&ensp;
2. Calculate fitness function of ants and antlions&ensp;
3. Find the non-dominated solutions and create the archive
4. While the end condition is not met
        For i = 1: size of initial population
            For updating archive
                Selection of two solutions from the archive
                Perform crossover with probability crossover $P_c$
                Perform mutation with probability mutation $P_m$
                Return solution to the archive and undergo ranking process
            End for
            For every ant
                Select a random antlion from the updated archive
                Select the elite using the Roulette wheel from the updated archive
                Update c and d using Eqs. (7) and (8)
                Create random walk and normalize it using Eqs. (3) and (4)
                Update the position of ant using Eq. (11)
            End for
        End for
5. Evaluate the objective values of all ants
6. Find the non-dominated solutions
7. Update the archive
            If the archive is full
            Delete some solutions using the Roulette wheel and Eq. (13) from the archive to accommodate new solutions
            End
8. End while
9. Return archive
End procedure
```

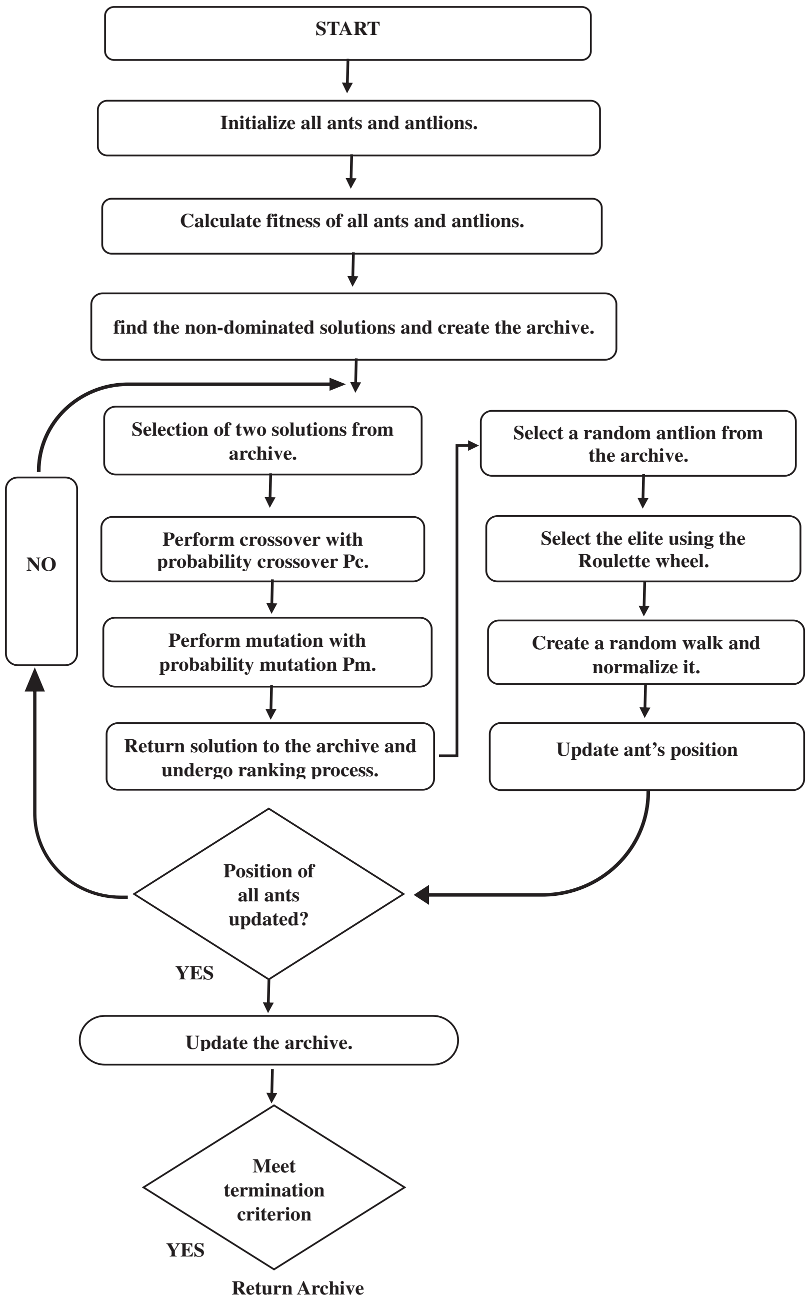

START
Initialize all ants and antlions.
Calculate fitness of all ants and antlions.
find the non-dominated solutions and create the archive.
Selection of two solutions from archive.
Perform crossover with probability crossover Pc.
Perform mutation with probability mutation Pm.
Return solution to the archive and undergo ranking process.
Select a random antlion from the archive.
Select the elite using the Roulette wheel.
Create a random walk and normalize it.
Update ant's position
NO
Position of all ants updated?
YES
Update the archive.
Meet termination criterion
YES
Return Archive

Figure 1: Flowchart of MOALG

5 Evaluation Method

In this section, the MOALG algorithm's efficiency for test problem having many objectives, and unconstrained and constrained problem have been assessed. These problems are employed to test the MOO handling of non-convex and non-linear situations. The algorithm was programmed in MATLAB 2021. To complete the work at that point, the computer's following features are employed: 8 GB of RAM of Intel Core i9 with CPU (1.19 GHz).

5.1 Performance-Based Metrics for Multiple-Objective

When analyzing the Pareto-optimal solutions those retrieved by the MOO techniques are frequently evaluated using the following criteria [45]:

1. Uniformity: The P_f has an evenly distributed range of good solutions.
2. Convergence: The solutions that come closest to the P_f are the best.
3. Coverage: The exact solution should cover the P_f as much as possible.

5.1.1 GD

Veldhuizen (1998) addressed the GD [46] to evaluate the distance between the obtained and true P_f.

$$\mathrm{GD} = \frac{\sum_{p=1}^{m} d_i}{m} \tag{14}$$

$$d_i = \left(\left(\sum_{i=1}^{n}\left(P_{f^{\mathrm{o}}_{\mathrm{I,p}}} - P_{f^{\mathrm{t}}_{\mathrm{I,p}}}\right)^2\right)^{\frac{1}{2}}\right)$$

where m, n are the number of Pareto-optimal solution and objective functions, respectively. $P_{f^{\mathrm{o}}_{\mathrm{I,p}}}$ and $P_{f^{\mathrm{t}}_{\mathrm{I,p}}}$ indicate pth obtained Pareto front (OPF) and the nearest point on true Pareto front (TPF) for the i-th objective. An algorithm with more effective convergence and coverage has a lower GD value.

5.1.2 IGD

It is common practice to compare meta-heuristics using IGD [47], which evaluates distance calculations from each reference point.

$$\mathrm{IGD} = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{m} \tag{15}$$

5.1.3 Spacing (S)

Spacing [48,49] is entitled as

$$\mathrm{S} = \frac{1}{m-1}\sum_{\mathrm{p}=2}^{\mathrm{m}}\left(D_p - \overline{D}\right)^2 \tag{16}$$

where in the OPF, D_p represents the absolute difference between two consecutive solutions and is defined as

$$\mathrm{D_p} = \left|\sum_{\mathrm{i}}^{\mathrm{n}} |P_{f^{\mathrm{o}}_{\mathrm{i,p-1}}} - P_{f^{\mathrm{t}}_{\mathrm{i,p}}}\right|$$

$\overline{D}$ is an average of calculated D_p values. Here, for the OPF small value of S indicates uniform spacing. A smaller spacing value corresponds to an improved uniform distribution of solutions.

5.1.4 Spread (Δ)

Deb in 2002 [50] addressed this indicator and used it to determine how extensively the generated non-dominated solutions have been spread out. It is defined in mathematical form as

$$\Delta = \frac{\sum_{j=1}^{n} d_j + \sum_{p=2}^{m} |d_p - \overline{d}|}{\sum_{j=1}^{n} d_j + (m-1)\,\overline{d}} \tag{17}$$

where d_j-Euclidean distance between TPF and OPF. d_p-Euclidean distance between two consecutive P_f and $\overline{d}$-an average of calculated d_p.

A smaller value of spread denotes a more diverse and even distribution of the non-dominated solutions.

5.1.5 HV

Brest in 2006 [51] addressed this indicator and used it to quantify the solution spreading in the objective space by measuring the volume occupied by them.

$$\mathrm{HV} = \bigcup_{i=1}^{K} v_i \tag{18}$$

K-non-dominated set of solutions and v_i-hypercube.

For, each x_i belongs to K, using a reference point W and for v_i at diagonal corners place the solution x_i. So, v_i union is used to calculate HV. An algorithm having a large value of HV is considered to be better.

5.2 Experimental Setup

The results of MOALO, MSSA, MOPSO, MOMVO, and MOCryStAl have been compared to MOALG. The Pareto optimal curve of MOALO and MOALG is shown in Fig. 2. The initial parameter for each algorithm is displayed in Table 3. A maximum of 1000 iterations and 100 search agents and an archive size of 100 and 30 independent runs were allotted for each experiment.

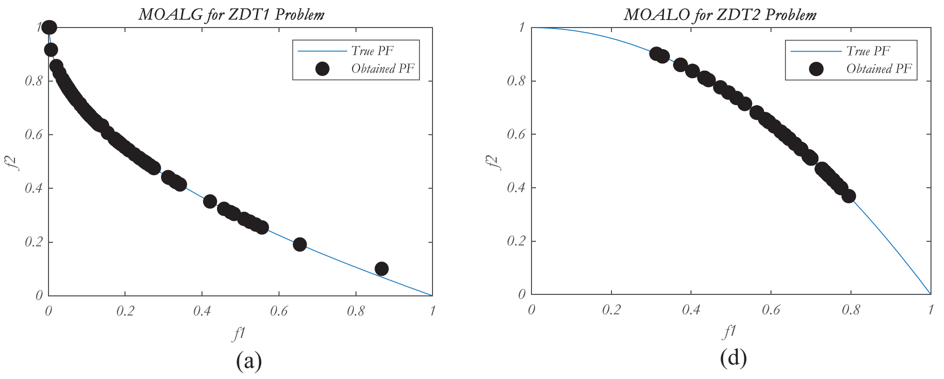
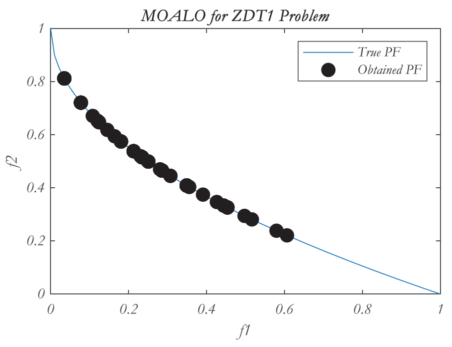
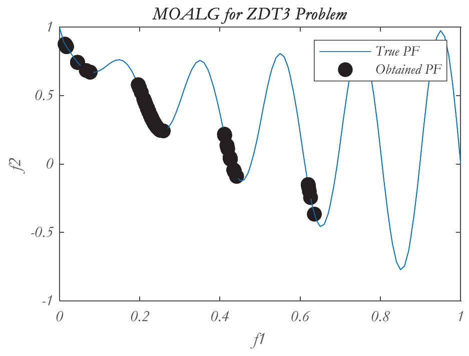


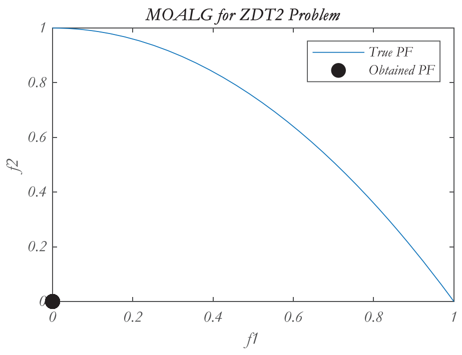
Figure 2: (Continued)



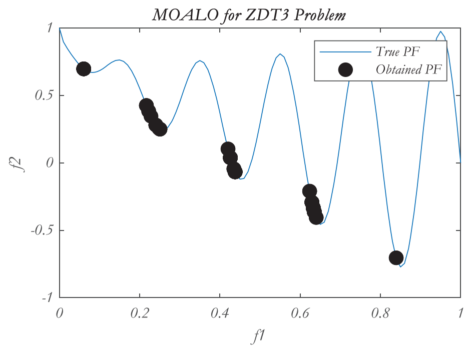
(b)



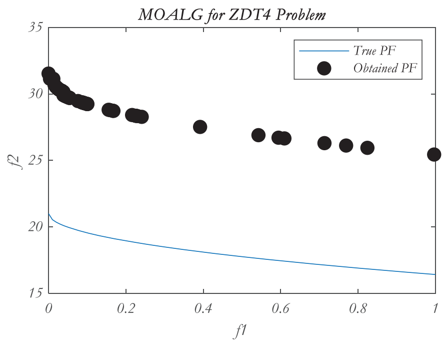
(e)



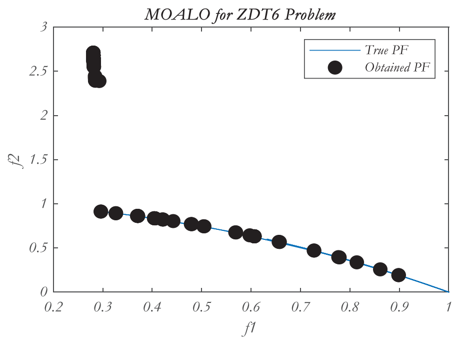
(c)



(f)



(g)



(j)

Figure 2: (Continued)

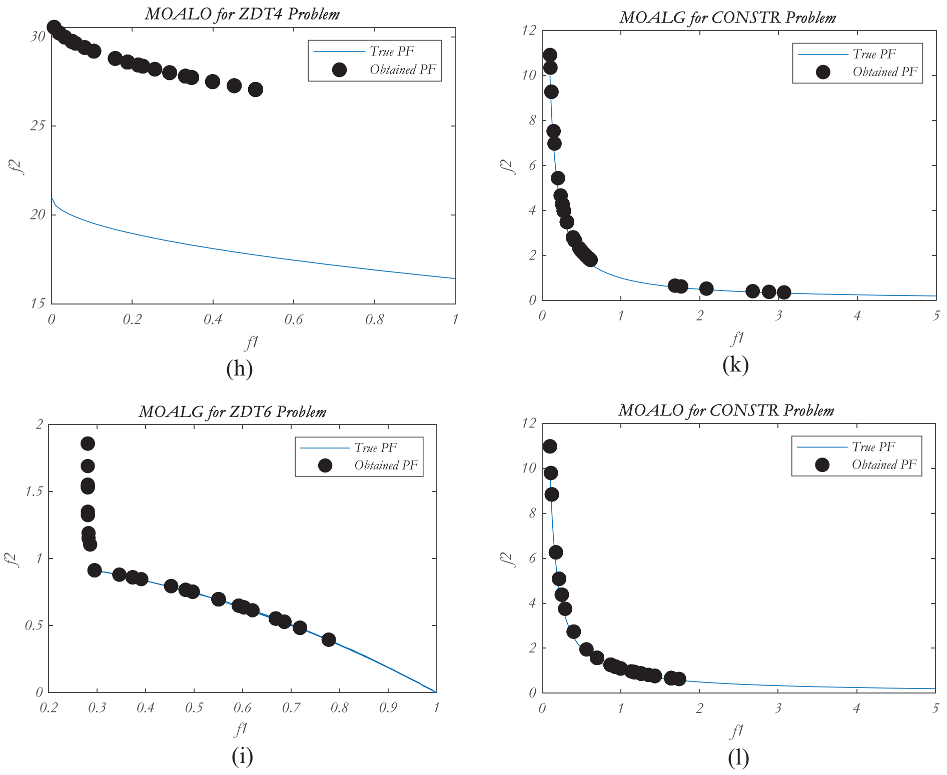


Figure 2: Pareto optimal front for the ZDT problems and CONSTR

Table 3: MOALG and other algorithms parameter settings

Parameters	MOALG	MOALO	MOPSO	MSSA	MOMVO	MOCryStAl
Population size (N_{pop})	**100**	**100**	**100**	**100**	**100**	**100**
Archive size (N_{rep})	**100**	**100**	**100**	**100**	**100**	**100**
Global learning coefficient (C_2)	**-**	**-**	**2**	**-**	**-**	**-**
Personal learning coefficient (C_1)	**-**	**-**	**1**	**-**	**-**	**-**
Inertia weight (w)	**-**	**-**	**0.4**	**-**	**-**	**-**
Beta	**-**	**-**	**4**	**4**	**4**	**4**
Gamma	**-**	**-**	**2**	**2**	**2**	**2**

6 Results

This section examines the performance of existing algorithms on various benchmark functions constrained and unconstrained problems for performance metrics such as spread, spacing, HV, GD, and IGD. MOALG outperforms various algorithms in GD and IGD results. Spread, spacing, and HV results indicate convergence and diversity, MOALG surpasses the MOALO algorithm on various benchmark functions.

Tables 4 and 5 display the outcomes of GD and IGD performance metrics, demonstrating superior convergence. In Table 4, for the unconstrained problem ZDT1, ZDT4, ZDT6 [52] and for the constrained CONSTR [53] problem, MOALG outperforms other existing algorithms. In Table 5, for the unconstrained problems, ZDT3, ZDT4, and ZDT6 the proposed hybrid MOALG gives competitive results. As well as for the constrained problem CONSTR it proves to be the best in terms of mean which shows MOALG has better convergence property.

Table 4: GD results of MOALG against other algorithms

Functions	Algorithms					
	MOALG	MOALO	MOPSO	MSSA	MOMVO	MOCrystal
ZDT1	**5.2226E-06**	5.3281E-06	1.7932E-02	1.2035E-02	7.9682E-03	2.5915E-04
Mean SD	**1.9039E-05**	1.9638E-05	5.0496E-03	1.9168E-03	1.2073E-02	4.0091E-04
ZDT2	9.2704E-04	**4.7519E-06**	1.4749E-01	1.1652E-02	5.3988E-03	1.6997E-04
Mean SD	4.7350E-04	**2.0198E-05**	6.5211E-02	2.1916E-03	2.9609E-04	4.7636E-04
ZDT3	3.6388E-05	**2.2862E-05**	2.0489E-04	1.6015E-02	1.0954E-02	2.0043E-04
Mean SD	1.9542E-04	1.1107E-04	**2.7322E-05**	2.0859E-03	1.5715E-02	1.6713E-04
ZDT4	**7.60E-03**	8.00E-03	4.1299E+00	6.0099E-01	1.2363E+00	1.8557E+01
Mean SD	1.08E-02	**7.60E-03**	3.9519E+00	3.1030E-01	7.0133E-01	1.1926E+01
ZDT6	**4.0856E-04**	6.2661E-04	2.5805E-02	2.1047E-02	1.0183E-02	1.0071E-02
Mean SD	**2.80E-03**	4.20E-03	5.8008E-02	9.6106E-03	8.4995E-03	3.0359E-03
CONSTR	**8.9160E-05**	9.7941E-05	8.3098E-04	1.6878E-03	9.2588E-04	1.5219E-03
Mean SD	1.40E-03	6.3863E-04	**3.3369E-05**	5.8950E-04	2.4850E-04	5.9590E-04

Table 5: IGD results of MOALG against other algorithms

Functions	Algorithms					
	MOALG	MOALO	MOPSO	MSSA	MOMVO	MOCrystal
ZDT1	2.3977E-04	1.00E-03	8.0879E-04	4.5908E-03	1.5496E-03	**2.3138E-04**
Mean SD	3.4539E-04	1.3E-03	1.4764E-03	1.1201E-03	1.0733E-04	**1.3159E-05**
ZDT2	8.70E-03	**8.1292E-04**	5.2023E-02	5.2059E-03	1.9238E-03	1.6173E-03
Mean SD	0	1.10E-03	9.5007E-03	1.6220E-03	5.1549E-04	**1.0133E-04**
ZDT3	**2.60E-04**	2.80E-03	2.6241E-04	5.1290E-03	1.6411E-03	1.3612E-03
Mean SD	**2.80E-05**	2.50E-03	3.9884E-05	8.9560E-04	2.1512E-04	1.7230E-03
ZDT4	**7.89E-02**	9.29E-02	7.0747E-01	1.8230E-01	2.4965E-01	8.8854E-01
Mean SD	1.14E-02	**1.11E-02**	3.1784E-01	9.5457E-01	1.3922E-01	1.0955E-01
ZDT6	**3.50E-04**	2.30E-03	8.2281E-03	2.0670E-03	5.0062E-04	3.5820E-04
Mean SD	6.6194E-04	1.10E-03	2.4862E-02	8.0527E-04	1.9857E-04	**1.6232E-04**
CONSTR	**5.1023E-04**	9.60E-03	5.1838E-04	2.1706E-03	1.8516E-03	1.1648E-03
Mean SD	4.1632E-04	9.60E-03	**5.5618E-05**	7.3799E-04	3.6526E-04	2.6604E-04

In Table 6, the results of the spread indicator in terms of mean have been displayed. So, due to the diversity of non-dominated solutions and better distribution MOALG outperforms MOALO for all constrained and unconstrained problems. Table 7 shows spacing which means better uniform distribution of solutions, MOALO has better results in all problems except ZDT3.

Table 6: Spread results of MOALG against MOALO algorithm

Algorithms	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	CONSTR
MOALG (Mean)	**1.13E-02**	**6.50E-03**	**1.00E-02**	**1.01E-02**	**1.32E-02**	**1.06E-02**
MOALO (Mean)	1.17E-02	1.17E-02	1.08E-02	1.02E-02	1.53E-02	1.22E-02

Table 7: Spacing results of MOALG against MOALO algorithm

Algorithms	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	CONSTR
MOALG (Mean)	1.1853E-04	7.6394E-04	**1.4412E-04**	8.30E-03	4.20E-03	1.7E-03
MOALO (Mean)	**2.7877E-05**	**3.2897E-05**	1.7138E-04	**6.60E-03**	**2.80E-03**	**4.3538E-04**

Table 8 shows the results of hypervolume, which means that if any algorithm has a higher HV value that means better convergence and diversity simultaneously. MOALG obtained a better HV score for four problems ZDT1, ZDT3, ZDT4, and CONSTR, and for the rest of the problems, MOALO proved to be better. Fig. 2 shows the visualization of the Obtained and True Pareto fronts. For the ZDT1 problem, MOALG has obtained the efficient Pareto front as compared to the MOALO concerning the convergence to the TPF and distributing the non-dominated solutions to the entire PF. For ZDT2, MOALG did not perform well and has a disconnected Obtained PF. The test problem, ZDT3, MOALG, and MOALO have competitive results, but if the results of HV are seen, it is obvious that for convergence and diversity together MOALG performs better than MOALO. Similarly, for ZDT4 both the algorithms are not performing well but MOALG has better results than MOALO for GD, Spread, and HV performance metrics. For the rest of the problem, ZDT6 the proposed algorithm solutions that are non-dominated are closer to True PF, and the same for CONSTR which is a constrained problem solutions are diverse on True PF and convergence can also be seen in the Fig. 2. As a result, it can be concluded that MOALG is capable of creating better solutions those are close to True PF and performs better than MOALO. The time taken by MOALG to complete 1000 iterations is 1 min 3 s whereas by MOALO is 33 s. Which is slightly more in MOALG. This can be ignored as the results of MOALG are better than MOALO.

Table 8: HV results of MOALG against MOALO algorithm

Algorithms	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	CONSTR
MOALG (Score)	**6.80E-03**	0	**9.0909E-03**	**4.80E-03**	3.30E-03	**9.10E-03**
MOALO (Score)	6.10E-03	**3.60E-03**	7.60E-03	0	**3.50E-03**	9.0E-03

7 Results Obtained for Engineering Design Problem

Two design problems have been solved. Four-bar truss design problem [54] is a structural optimization problem where objective functions are volume and displacement. The cross-sectional area relates to design variables. Welded beam [55] has objective functions fabrication cost and deflection of the beam which should be minimized.

7.1 Experimental Results

Tables 9 and 10 show that MOALG surpasses other algorithms in design problems. The findings of the four-bar truss design problem are provided in Table 9, and the results of the welded beam design problem are provided in Table 10. Fig. 3 gives the representation of True PF and Obtained PF. MOALG shows better performance and coverage in truss-bar problems. For welded beam problems, hybrid outperforms another algorithm for GD, IGD, Spread, and HV showing convergence and diversity together. The notations used for MOALG are represented in Table 11.

Table 9: Results for four-bar truss engineering design problem

Algorithms	GD (Mean & SD)	IGD (Mean & SD)	Spread (Mean)	Spacing (Mean)	HV (Score)
MOALG	**1.20E-01** **2.541E-02**	**1.85E-02** **2.0025E-05**	**1.0965E-02**	**1.10E-01**	**6.19E-02**
MOALO	1.264E-01 3.27E-02	1.062E-01 1.52E-02	3.70E-01	1.18E+00	3.70E-02
MOPSO	1.4095E+01 5.1580E-01	2.0010E-02 3.9632E-05	1.4876E+00	5.3605E+00	1.16E-04
MSSA	7.7966E+00 3.5486E+00	2.1434E-02 3.8292E-04	1.2019E+00	6.1160E+00	8.90E-04
MOMVO	1.1017E+01 4.6552E+00	2.1020E-02 4.6879E-04	1.4014E+00	4.8245E+00	6.17E-04
MOCryStal	6.8970E+00 1.2099E+00	2.0005E-02 3.5863E-05	1.5436E+00	1.1012E+00	5.69E-05

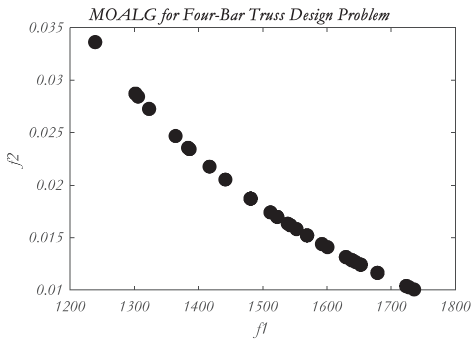
Table 10: Results of welded beam engineering design problem

Algorithms	GD (Mean & SD)	IGD (Mean & SD)	Spread (Mean)	Spacing (Mean)	HV (Score)
MOALG	**4.632E-03** **1.923E-03**	**4.888E-04** **3.652E-05**	**1.523E-01**	5.62E-01	**1.71E-03**
MOALO	6.65E-03 7.42E-03	1.52E-03 4.65E-03	1.978E-01	4.26E-02	1.49E-03

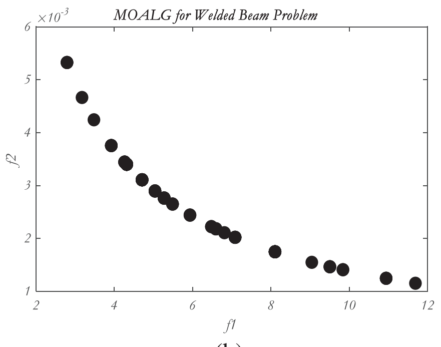
(Continued)

Table 10 (continued)

Algorithms	GD (Mean & SD)	IGD (Mean & SD)	Spread (Mean)	Spacing (Mean)	HV (Score)
MOPSO	1.1946E-02 1.9956E-03	5.9705E-04 4.6341E-05	1.0072E+00	2.3432E-01	1.16E-04
MSSA	6.3467E-03 3.0983E-03	4.8242E-03 3.5852E-03	7.9085E-01	**1.8912E-01**	1.90E-04
MOMVO	1.5031E-02 4.7477E-03	2.1075E-03 4.2017E-04	1.0552E+00	2.0967E-01	2.17E-04
MOCryStal	6.6659E-02 1.1624E-01	1.3180E-03 3.4394E-04	1.0709E+00	3.6825E-01	1.69E-03



(a)



(b)

Figure 3: True PF and obtained PF for four-bar truss and welded beam design problem

Table 11: Notations used for MOALG

Symbol	Description
Ub	Upper bound
Lb	Lower bound
p	Maximum number of iterations
cumsum	Cumulative sum
RW	Random walk
b_i	Maximum of RW in ith variable
a_i	Minimum of RW in ith variable
d^t	Maximum of all variable at tth iteration
c^t	Minimum of all variable at tth iteration
R_A^t	RW around the antlion
R_E^t	RW around the elite
Ant_i^t	Position of ith at tth iteration
P_f	Pareto optimal front

8 Conclusion

This paper proposed a hybrid of MOALO and GA named MOALG. Selection mechanism and crossover, mutation operator have been used for bettering of non-dominated solutions in the archive. The primary objective behind the development of MOALG is to overcome the limitations of ALO of getting caught in a local optimum and more computational effort to converge GA. The algorithm tests various problems using performance metrics such as GD, IGD, Spread, Spacing, and HV. For the comparison, the other competitive algorithms were utilized as MOALO, MOPSO, MSSA, MOMVO, and MOCryStAl. MOALG has been observed to benefit from strong convergence and coverage. MOALG has performed better than MOALO which can be seen through the numerical and statistical results. It can be viewed that MOALG can also find the Pareto optimal front of any shape. Another conclusion is that as per the NFL theorem, MOALG can perform better for other problems as well.

For future work, MOALG can be tested for various engineering design problems and many other real-life problems of multi-objectives.

Acknowledgement: The authors wish to acknowledge the resources of Chandigarh University, Mohali, India, Soonchunhyang University, Korea, Michigan State University, USA, and Mansoura University, Egypt for carrying out the research work.

Funding Statement: This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea government (MSIT) (No. RS-2023-00218176), and the Soonchunhyang University Research Fund.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: R. Sharma, A. Pal, N. Mittal; data collection: S. Van, L. Kumar; analysis and interpretation of results: R. Sharma, Y. Nam; draft manuscript preparation: L. Kumar, R. Sharma, A. Pal, M. Abouhawwash. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data is available from the authors upon reasonable request from the authors.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. G. D. Allen and F. Y. Edgeworth, "Mathematical psychics," *Econ. J.*, vol. 42, no. 166, pp. 307, 1932.
[2] V. Pareto, *Manual of Political Economy*. Oxford, England: Oxford University Press, 1906.
[3] S. Kumar, G. G. Tejani, and S. Mirjalili, "Modified symbiotic organisms search for structural optimization," *Eng. Comput.*, vol. 35, no. 4, pp. 1269–1296, 2019. doi: 10.1007/s00366-018-0662-y.
[4] S. M. Almufti, "Historical survey on metaheuristics algorithms," *Int. J. Sci. World*, vol. 7, no. 1, pp. 1–12, 2019.
[5] J. H. holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. London, England: MIT Press, 1992.
[6] K. Storn and R. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 1, pp. 341–359, 1997. doi: 10.1023/A:1008202821328.
[7] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1994.

[8] 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.

[9] R. Eberhart and J. Kennedy, "New optimizer using particle swarm theory," in *Proc. Sixth Int. Symp. Micro Mach. Human Sci.*, Nagoya, Japan, 1995, pp. 39–43.

[10] W. Zhao, L. Wang, and S. Mirjalili, "Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications," *Comput. Methods. Appl. Mech. Eng.*, vol. 388, pp. 114194, 2022. doi: 10.1016/j.cma.2021.114194.

[11] B. Akay and D. Karaboga, "Artificial bee colony algorithm for large-scale problems and engineering design optimization," *J. Intell. Manuf.*, vol. 23, no. 1, pp. 1001–1014, 2012. doi: 10.1007/s10845-010-0393-4.

[12] M. Dorigo and K. Socha, "Ant colony optimization," in *Handbook of Approximation Algorithms and Metaheuristics*, Santa Barbara, USA: CRC Press, 2007, vol. 1, pp. 26-1–26-14.

[13] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *Int. J. Math. Model. Numer. Optim.*, vol. 1, no. 4, pp. 330–343, 2010. doi: 10.1504/IJMMNO.2010.035430.

[14] S. Mirjalili. "The ant lion optimizer," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, 2015. doi: 10.1016/j.advengsoft.2015.01.010.

[15] G. G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural. Comput. Appl.*, vol. 31, no. 7, pp. 1995–2014, 2019.

[16] X. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *Int. J. Bio-Inspired Computing*, vol. 2, no. 2, pp. 78–84, 2010.

[17] G. G. Wang. "Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Memet. Comput.*, vol. 10, no. 2, pp. 151–164, 2018. doi: 10.1007/s12293-016-0212-3.

[18] A. Kaveh and A. Dadras, "A novel meta-heuristic optimization algorithm: Thermal exchange optimization," *Adv. Eng. Softw.*, vol. 110, pp. 69–84, 2017. doi: 10.1016/j.advengsoft.2017.03.014.

[19] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009. doi: 10.1016/j.ins.2009.03.004.

[20] A. A. L. I. Alhussan *et al.*, "A binary waterwheel plant optimization algorithm for feature selection," *IEEE Access*, vol. 11, pp. 94227–94251, 2023.

[21] D. Golbaz, R. Asadi, E. Amini, H. Mehdipour, and M. Nasiri, "Layout and design optimization of ocean wave energy converters: A scoping review of state-of-the-art canonical, hybrid, cooperative, and combinatorial optimization methods," *Energy Rep.*, vol. 8, pp. 15446–15479, 2022. doi: 10.1016/j.egyr.2022.10.403.

[22] E. H. Houssein, M. A. Mahdy, M. J. Blondin, D. Shebl, and W. M. Mohamed, "Hybrid slime mould algorithm with adaptive guided differential evolution algorithm for combinatorial and global optimization problems," *Expert Syst. Appl.*, vol. 174, pp. 114689, 2021. doi: 10.1016/j.eswa.2021.114689.

[23] B. Alexander and M. Wagner, "Optimisation of large wave farms using a multi-strategy evolutionary framework," in *Proc. Genet. Evol. Comput.*, New York, USA, 2020, pp. 1150–1158.

[24] D. Whitley and J. Rowe, "Focused no free lunch theorem," in *Proc. 10th Annual Conf. Genet. Evol. Comput.*, Atlanta, GA, USA, 2008, pp. 811–818.

[25] P. Ngatchou, A. Zarei, and M. A. El-Sharkawi, "Pareto multi objective optimization," in *Proc. 13th Int. Conf. Intell. Syst. Appl. Power Syst.*, Arlington, VA, USA, 2005, pp. 84–91.

[26] X. S. Yang. "Bat algorithm for multi-objective optimisation," *Int. J. Bio-Inspir. Com.*, vol. 3, no. 5, pp. 267–274, 2011.

[27] R. Akbari, R. Hedayatzadeh, K. Ziarati, and B. Hassanizadeh, "A multi-objective artificial bee colony algorithm," *Swarm Evol. Comput.*, vol. 2, pp. 39–52, 2012. doi: 10.1016/j.swevo.2011.08.001.

[28] J. S. Chou and D. N. Truong, "Multiobjective forensic-based investigation algorithm for solving structural design problems," *Autom. Constr.*, vol. 134, pp. 104084, 2022. doi: 10.1016/j.autcon.2021.104084.

[29] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. D. S. Coelho, "Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization," *Expert Syst. Appl.*, vol. 47, pp. 106–119, 2016. doi: 10.1016/j.eswa.2015.10.039.

[30] C. A. Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. Congress Evol. Comput.*, NW, Washington DC, USA, vol. 2, 2002, pp. 1051–1056.

[31] N. Khodadadi, M. Azizi, S. Talatahari, and P. Sareh, "Multi-objective crystal structure algorithm (MOCryStAl): Introduction and performance evaluation," *IEEE Access*, vol. 9, pp. 117795–117812, 2021.
[32] S. Mirjalili, P. Jangir, S. Z. Mirjalili, S. Saremi, and I. N. Trivedi, "Optimization of problems with multiple objectives using the multi-verse optimization algorithm," *Knowl.-Based Syst.*, vol. 134, pp. 50–71, 2017.
[33] S. Kumar, P. Jangir, G. G. Tejani, and M. Premkumar, "MOTEO: A novel physics-based multiobjective thermal exchange optimization algorithm to design truss structures," *Knowl.-Based Syst.*, vol. 242, pp. 108422, 2022. doi: 10.1016/j.knosys.2022.108422.
[34] A. Kaveh, T. Bakhshpoori, and E. Afshari, "An efficient hybrid particle swarm and swallow swarm optimization algorithm," *Eng. Comput.*, vol. 143, pp. 40–59, 2014. doi: 10.1016/j.compstruc.2014.07.012.
[35] J. Luo, Y. Qi, J. Xie, and X. Zhang, "A hybrid multi-objective PSO-EDA algorithm for reservoir flood control operation," *Appl. Soft Comput.*, vol. 34, pp. 526–538, 2015. doi: 10.1016/j.asoc.2015.05.036.
[36] C. Lu, L. Gao, X. Li, Q. Pan, and Q. Wang, "Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm," *J. Clean. Prod.*, vol. 144, pp. 228–238, 2017. doi: 10.1016/j.jclepro.2017.01.011.
[37] I. Tariq *et al.*, "MOGSABAT: A metaheuristic hybrid algorithm for solving multi-objective optimisation problems," *Neural. Comput. Appl.*, vol. 32, no. 8, pp. 3101–3115, 2020.
[38] G. Dhiman, "MOSHEPO: A hybrid multi-objective approach to solve economic load dispatch and micro grid problems," *Appl. Intell.*, vol. 50, no. 1, pp. 119–137, 2020.
[39] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "A hybrid multi-objective metaheuristic optimization algorithm for scientific workflow scheduling," *Clust. Comput.*, vol. 24, no. 2, pp. 1479–1503, 2021. doi: 10.1007/s10586-020-03205-z.
[40] M. A. Deif, A. A. A. Solyman, M. H. Alsharif, S. Jung, and E. Hwang, "A hybrid multi-objective optimizer-based SVM model for enhancing numerical weather prediction: A study for the seoul metropolitan area," *Sustain.*, vol. 14, no. 1, pp. 1–17, 2022. doi: 10.3390/su14010296.
[41] M. H. Marghny, E. A. Zanaty, W. H. Dukhan, and O. Reyad, "A hybrid multi-objective optimization algorithm for software requirement problem," *Alex. Eng. J.*, vol. 61, no. 9, pp. 6991–7005, 2022. doi: 10.1016/j.aej.2021.12.043.
[42] R. K. Avvari and V. D. M. Kumar, "A novel hybrid multi-objective evolutionary algorithm for optimal power flow in wind, PV, and PEV systems," *J. Oper. Autom. Power Eng.*, vol. 11, no. 2, pp. 130–143, 2023.
[43] S. Mirjalili, P. Jangir, and S. Saremi, "Multi-objective ant lion optimizer: A multi-objective optimization algorithm for solving engineering problems," *Appl. Intell.*, vol. 46, no. 1, pp. 79–95, 2017.
[44] A. Yaghoubzadeh-Bavandpour, O. Bozorg-Haddad, B. Zolghadr-Asli, and V. P. Singh, "Computational intelligence: An introduction," *Stud. Comp. Intell.*, vol. 1043, pp. 411–427, 2022.
[45] J. S. Chou and D. N. Truong, "Multiobjective optimization inspired by behavior of jellyfish for solving structural design problems," *Chaos, Solit. Fractals*, vol. 135, pp. 109738, 2020. doi: 10.1016/j.chaos.2020.109738.
[46] J. A. Nuh, T. W. Koh, S. Baharom, M. H. Osman, and S. N. Kew, "Performance evaluation metrics for multi-objective evolutionary algorithms in search-based software engineering: Systematic literature review," *Appl. Sci.*, vol. 11, no. 7, pp. 1–25, 2021.
[47] N. Khodadadi, L. Abualigah, E. S. M. El-Kenawy, V. Snasel, and S. Mirjalili, "An archive-based multi-objective arithmetic optimization algorithm for solving industrial engineering problems," *IEEE Access*, vol. 10, no. 1, pp. 106673–106698, 2022. doi: 10.1109/ACCESS.2022.3212081.
[48] A. J. Nebro, J. J. Durillo, and C. A. C. Coello, "Analysis of leader selection strategies in a multi-objective particle swarm optimizer," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, 2013, pp. 3153–3160.
[49] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *Evol. Comput.*, vol. 8, no. 2, pp. 149–172, 2000.
[50] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 892–899.

[51] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evolut. Comput.*, vol. 10, no. 6, pp. 646–657, 2006.
[52] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000. doi: 10.1162/106365600568202.
[53] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002. doi: 10.1109/4235.996017.
[54] C. A. Coello and G. T. Pulido, "Multiobjective structural optimization using a microgenetic algorithm," *Struct. Multidiscipl. Optim.*, vol. 30, no. 5, pp. 388–403, 2005. doi: 10.1007/s00158-005-0527-z.
[55] T. Ray and K. M. Liew, "A swarm metaphor for multiobjective design optimization," *Eng. Optim.*, vol. 34, no. 2, pp. 141–153, 2002.

Appendix

Appendix A: Multi-Objective Unconstrained ZDT Test Problems

ZDT1:

$$f_1 = u_1, f_2 = h\left(1 - \sqrt{\frac{f_1}{h}}\right) \text{ where } h = 1 + \frac{9}{d-1}\sum_{i=2}^{d} u_i$$

$$0 \le u_i \le 1, 1 \le i \le 30$$

ZDT2:

$$f_1 = u_1, f_2 = h\left(1 - \left(\frac{f_1}{h}\right)^2\right) \text{where } h = 1 + \frac{9}{d-1}\sum_{i=2}^{d} u_i$$

$$0 \le u_i \le 1, 1 \le i \le 30$$

ZDT3:

$$f_1 = u_1, f_2 = h\left(1 - \sqrt{\frac{f_1}{h}} - \frac{f_1}{h}\sin(10\pi f_1)\right) \text{where } h = 1 + \frac{9}{d-1}\sum_{i=2}^{d} u_i$$

$$0 \le u_i \le 1, 1 \le i \le 30$$

ZDT4:

$$f_1 = u_1, f_2 = \mathrm{h}\left(1 - \sqrt{\frac{f_1}{\mathrm{h}}}\right) \text{where } h = 1 + 10\,(d-1) + \sum\nolimits_{i=2}^{d} (u_i^2 - 10\cos(4\pi u_i))$$

$$0 \le u_1 \le 1, 1 \le i \le 10, -5 \le u_i \le 5$$

ZDT6:

$$f_1 = 1 - \exp(-4u_1)\sin^6(6\pi u_1), f_2 = h\left(1 - \left(\frac{f_1}{h}\right)^2\right) \text{where } h = 1 + 9\left(\frac{\sum_{i=2}^{d} u_i}{d-1}\right)^{0.25}$$

$$0 \le i \le 10, 0 \le u_i \le 1$$

Appendix B: Multi-Objective Constrained Test Problems

CONSTR:

A convex Pareto-front is obtained here equation containing two constraints:

$f_1 = x_1, f_2 = \frac{1 + x_2}{x_1}$ and $g_1 = 6 - (x_2 + 9x_1)$

$0.1 \leq x_1 \leq 1, 0 \leq x_2 \leq 5$

Appendix C: Multi-Objective Engineering Design Problems

FOUR-BAR-TRUSS DESIGN PROBLEM:

This is a well-known field in structural optimization were f_1 and f_2 are volume and displacement.

Minimize:

$f_1 = 200 \times x_1 + sqrt\,(2 \times x_2 + sqrt\,(x_3) + x_4)$ and

$$f_2 = 0.01 \times \left(\left(\frac{2}{x_1} \right) + \left(\frac{2 \times sqrt\,(2)}{x_2} \right) - \left(\frac{2 \times sqrt\,(2)}{x_3} \right) + \left(\frac{2}{x_1} \right) \right)$$

$1 \leq x_1 \leq 3, 1.4142 \leq x_2 \leq 3, 1.4142 \leq x_3 \leq 3, 1.4142 \leq x_4 \leq 3$

WELDED BEAM DESIGN PROBLEM:

Minimize: $f_1 = 1.10471 \times x_1^2 \times x_2 + 0.04811 \times x_3 \times x_4 \times (14 + x_2)$

Minimize: $f_2 = 65856000/(30 \times 10^6 \times x_4 \times x_3^3)$

where $g_1(x) = \tau - 13600, g_2(x) = \sigma - 30000, g_3(x) = x_1 - x_4, g_4(x) = 6000 - P$

$0.125 \leq x_1 \leq 5, 0.1 \leq x_2 \leq 10, 0.1 \leq x_3 \leq 10, 0.125 \leq x_4 \leq 5$

$$Q = 6000 \times \left(14 + \frac{x_2}{2}\right), D = sqrt \times \left(\frac{{x_2}^2}{4} + \frac{(x_1 + x_3)^2)}{4} \right),$$

$$J = 2 \times \left(x_1 \times x_2 \times sqrt\,(2) \times \left(\frac{{x_2}^2}{12} + \frac{(x_1 + x_3)^2}{4} \right) \right), \alpha = \frac{6000}{sqrt\,(2) \times x_1 \times x_2}, \beta = Q \times \frac{D}{J}$$