MOSOA: A New Multi-objective Seagull Optimization Algorithm

Gaurav Dhiman[1,*], Krishna Kant Singh[2], Mukesh Soni[3], Atulya Nagar[4], Mohammad Dehghani[5], Adam Slowik[6], Amandeep Kaur[7], Ashutosh Sharma[8], Essam H. Houssein[9], Korhan Cengiz[10]

[1,*]Department of Computer Science, Government Bikram College of Commerce, Patiala-147001, Punjab, India, Email: gdhiman0001@gmail.com

[2]Department of Electronics & Communication Engineering, KIET Group of Institution, Delhi-NCR, Ghaziabad, India, Email: krishnaiitr2011@gmail.com

[3]Department of Computer Engineering, Smt. S. R. Patel Engineering College, Unjha, Gujarat, 384170, India, Email: soni.mukesh15@gmail.com

[4]Pro-Vice-Chancellor for Research, Liverpool Hope University, Hope Park, Liverpool L16 9JD, United Kingdom, Email: nagara@hope.ac.uk

[5]Department of Electrical and Electronics Engineering, Shiraz University of Technology, Shiraz, Iran, Email: adanbax@gmail.com

[6]Department of Electronics and Computer Science, Koszalin University of Technology, Sniadeckich 2, 75-453, Koszalin, Poland, Email: aslowik@ie.tu.koszalin.pl

[7]Department of Computer Science and Engineering, Sri Guru Granth Sahib Wrold University, Fatehgarh Sahib, Punjab, India, Email: amandeep1426@sggswu.edu.in

[8]School of Electronics and Electrical Engineering, Lovely Professional University, Punjab, India, Email: sharmaashutosh1326@gmail.com

[9]Faculty of Computers and Information, Minia University, Minia 61519, Egypt, Email: essam.halim@mu.edu.eg

[10]Department of Electrical and Electronics Engineering, Trakya University, 22030, Edirne, Turkey, Email: korhancengiz@trakya.edu.tr

Affiliation

Abstract
This study introduces the extension of currently developed *Seagull Optimization Algorithm (SOA)* in terms of multi-objective problems, which is entitled as *Multi-objective Seagull Optimization Algorithm (MOSOA)*. In this algorithm, a concept of dynamic archive is introduced, which has the feature to cache the non-dominated *Pareto* optimal solutions. The roulette wheel selection approach is utilized to choose the effective archived solutions by simulating the migration and attacking behaviors of seagulls. The proposed algorithm is approved by testing it with twenty-four benchmark test functions, and its performance is compared with existing metaheuristic algorithms. The developed algorithm is analyzed on six constrained problems of engineering design to assess its appropriateness for finding the solutions of real-world problems. The outcomes from the empirical analyzes depict that the proposed algorithm is better than other existing algorithms. The proposed algorithm also considers those *Pareto* optimal solutions, which demonstrate high convergence.
Keywords: Convergence;Diversity;Pareto Solutions; Multi-objective Optimization;Seagull Optimization Algorithm; Engineering Design Problems.

1 Introduction

Metaheuristic strategies have received more attention from researchers to solve real-life problems (Wu, Qian, Ni, & Fan, 2012b; Hajiaghaei-Keshteli & Aminnayeri, 2013; Cheraghalipour, Hajiaghaei-Keshteli, & Paydar, 2018). These techniques were commonly used in various engineering fields (Dhiman & Kumar, 2017; Baykasoğlu & Akpinar, 2017, 2015; Rao, Savsani, & Vakharia, 2012) due to the computationally inexpensive. Hence, the distinguishing characteristic of these methods is the multi-objective approach.

The multi-objective optimizer has the capability of operating together with multiple objective functions. The multi-objective optimizer must compromise for optimal solutions in most of the situations. Conflicts are also often found among the optimal solutions. Using three main stages, such as *priori*, *posteriori* and *interactive* approaches (R. T. Marler & Arora, 2004; Branke, Deb, Dierolf, & Osswald, 2004; R. Marler & Arora, 2004; Y. Zhang, Gong, & Ding, 2011), these conflicts can be eliminated. In the *priori* step, a single-objective with a collection of weights is transferred from a multi-objective problem that characterizes each segment's importance in solving domain-specific problems. On the other hand, in the *posteriori* step, by solving the benchmark problem (Deb, 2012), an exclusive result is attempted to obtain based on certain obligations. *Interactive* approaches also known as the human-in-the-loop technique, continuously extracting decision-maker preferences and integrating them during the optimization process to find the correct *Pareto* optimal solutions (R. T. Marler & Arora, 2004).

Conversely, multi-objective optimization (*MOO*) does not have a single solution and varying consensus among the different objectives. To get the *Pareto* fronts is extremely difficult for any *MOO*, as it needs delivering multiple points for successful hypothesis on the *Pareto* line. Even so, there is no assurance that the *MOO's* solutions on a *Pareto* front will spread equally on the front (Yang, Karamanoglu, & He, 2014). It can generate extremely complex hyper-surface in terms of the *Pareto* front (R. Marler & Arora, 2004; Yang, 2010; Madavan, 2002) when dealing with multi-dimensional issues. Therefore, predicting the solution of these multi-dimensional problems is quite difficult. The concept of *MOO* using stochastic techniques was initially introduced by David Schaffer (Coello,

2006). The mastery of these methods is limited to optimal prevention and gradient-free method, which makes them applicable to various problems. Such multi-objective methods can be implemented in various fields of engineering and science, such as: bio-informatics (Handl, Kell, & Knowles, 2007), civil engineering (Luh & Chueh, 2004), network engineering (Chen & Hammami, 2015), mechanical engineering (Kipouros et al., 2008; Dhiman & Kumar, 2019b), software engineering (Kaur & Dhiman, 2019), and so on. Some examples of optimization techniques that solve multi-objective problems include the Non-dominated Genetic Algorithm 2 (*NSGA-II*) (Deb, Pratap, Agarwal, & Meyarivan, 2002), Multi-objective Evolutionary Algorithm based on Decomposition (*MOEA/D*) (Q. Zhang & Li, 2007), and Multi-objective Particle Swarm Optimization (*MOPSO*) (Coello Coello & Lechuga, 2002). Though, these strategies can not provide solutions to all kinds of optimization problems, but they can approximate the true optimal *Pareto* solutions (*POSs*) (Wolpert & Macready, 1997). A *MOO* algorithm is used in this study which is an advancement of the newly developed Seagull Optimization Algorithm (*SOA*) (Dhiman & Kumar, 2019a). The proposed algorithm is applied to solve the *MOO* problem, so entitled as *Multi-objective Seagull Optimization Algorithm (MOSOA)*. Contributions are arranged according to the following:

- The *SOA* algorithm integrates an archive part, which is liable to accumulate the non-dominated *Pareto* solutions all over.
- It is suggested that a leader selection method choose the solutions regarding the prey position from the collection.
- To improve the non-dominated solutions, a grid mechanism is embedded in the *MOSOA* to remove the most crowded sections.

The efficacy of the proposed MOSOA is tested on ten *IEEE CEC-9* multi-objective research problems (Q. Zhang et al., 2008). It is also tested on five multi-objective evaluation problems with the *ZDT* (Zitzler, Deb, & Thiele, 2000) and nine *DTLZ* (Deb, Thiele, Laumanns, & Zitzler, 2005). Six well-known optimization methods, such as *MOPSO* (Coello Coello & Lechuga, 2002), *NSGA-II* (Deb et al., 2002), *MOEA/D* (Q. Zhang & Li, 2007), *PESA-II* (Corne, Jerram, Knowles, Oates, & J, 2001), *MOSHEPO* (Dhiman, 2020), and *MOACO* (Angus & Woodward, 2009), are compared for experimental performance. This research also uses the four performance metrics to test the efficiency of algorithms. It can be inferred from the experimental results that the *MOSOA* is a highly promising multi-objective metaheuristic strategy for finding solutions to real world problems.

The remainder section of this article is structured as follows. Section 2 presents the background for the *MOO* and related works. Section 3 introduces *SOA* algorithm's mathematical principles, accompanied by the proposed *MOSOA* technique in Section 4. Section 5 contains the findings of experiments and the discussions. In Section 6, the efficacy of the proposed *MOSOA* is assessed on six engineering design *MOO* problems. Finally, conclusions and future works are given in Section 7.

2 Background

2.1 Definitions of MOO

Multi-objective technique can be described as an optimization method that can have functions of a given problem with more than one objective (Coello Coello, 2009):

$$Minimize : F(\vec{z}) = [f_1(\vec{z}), f_2(\vec{z}), \ldots, f_n(\vec{z})] \tag{1}$$

Subject to:
$$g_i(\vec{z}) \geq 0,\ i = 1, 2, \ldots, m \tag{2}$$

$$h_i(\vec{z}) = 0,\ i = 1, 2, \ldots, p \tag{3}$$

where $\vec{z} = [z_1, z_2, \ldots, z_k]^T$ is the decision variables vector, g_i is the i^{th} in-equality constraint, h_i is the i^{th} equality constraint, m is the number of in-equality constraint, and p is the number of equality constraint.

2.2 Related works

Several *MOO* algorithms have recently been published in the literature. There are a number of issues related to multi-objective metaheuristic techniques, such as the variety of complex, infallible solutions (Deb, 2012) and distinctive optimum performance. The principle of knowledge moving between search space and agents should solve those problems. In the single iteration, optimal *Pareto* front should be obtained by using *MOO* algorithms. Non-dominated Sorting Genetic Algorithm 2 (*NSGA-II*) (Deb et al., 2002) is one of the most popular multi-objective metaheuristic technique. Non-dominated sorting method along with rigid and niching operator is used in this technique to get the best outputs. In *NSGA-II*, the random population is generated and individuals are clustered according to the technique of non-dominated sorting. Another random population is created to assist operators in selection, mutation, and recombination. A new population is created in each simulation and the sorting is done via non-dominated sorting. The viability of selecting a new one entirely depends on the degree of control of the final population. The whole cycle remains iterative before the correct findings are found. The most common *MOO* algorithm is the *MOPSO* (Coello Coello & Lechuga, 2002). It is an extension variant of a *PSO* algorithm with a single-objective. In the *MOPSO*, the archive concept is used to store and retrieve the *POSs*. Additionally, in the *MOPSO*, the mutation operator is often used to boost its efficiency.

Multi-objective Evolutionary Algorithm based on Decomposition (*MOEA/D*) (Q. Zhang & Li, 2007) is a well-known *MOO* algorithm. It is based on the parallel and distributed computing principle for the decomposition of a problem. All of the sub-problems and weighting vector are assigned to a single-objective function which results multi-sub-problem. The another method is commonly divided into two sub-processes: cooperation and completion. Cooperation as the name implies may help to solve the problem by mutual cooperation between neighboring members. If neighbors have better solutions then they can substitute the solutions to get the better solutions. This task of finding the best answer by challenging neighbors is called the process of competition. *MOEA/D* has minimal computational complexity and the convergence speed is agile than *NSGA-II*.

In 2009, Angus and Woodward (Angus & Woodward, 2009) created the Ant Colony Optimization (*ACO*) multi-objective method, called the *MOACO*. It uses concepts like the pheromone model, the design process, the estimation of the solution and the method of updating. Gong *et al.* (Gong, Jiao, Du, & Bo, 2008) has created a neighbor's Non-dominated Immune Algorithm (*NNIA*) for *MOO* problems. It depends on the ideas of non-dominated neighbor selection, immune operator motivated, heuristic search operators, and elitism. However, one of the *NNIA's* fundamental drawbacks is the lack of diversity. This problem has been resolved in the *NNIA's* updated version and the latest method is being developed called the *NNIA2*.

In addition to these *MOO* algorithms, several other algorithms are also proposed in literature, such as Multi-objective Cat Swarm Optimization (*MOCSO*) (Pradhan & Panda, 2012), Multi-objective Artificial Bee Colony Algorithm (*MOABC*) (Hancer, Xue, Zhang, Karaboga, & Akay, 2015), Multi-objective Flower Pollination Algorithm (*MOFPA*) (Yang et al., 2014), a self-adaptive Artificial Bee Colony algorithm (Xue, Jiang, Zhao, & Ma, 2018), hybrid Harmony Search and Artificial Bee Colony algorithm (Wu, Qian, Ni, & Fan, 2012a), Multi-objective Spotted Hyena Optimizer (Dhiman & Kumar, 2018b), Multi-objective Spotted Hyena and Emperor Penguin Optimizer (*MOSHEPO*) (Dhiman, 2020), and external archive guided Multi-objective Evolutionary Algorithm based on decomposition (Cai, Li, Fan, & Zhang, 2014).

While several optimization algorithms are mentioned in the literature, yet no algorithm is capable of solving these optimization problems. For example, in *MOPSO* and *MOACO*, when the region where the particles or ants explore happens to be of lower quality than the particles' previous best positions, the algorithm is in high risk of becoming trapped and being unable to improve any further. In this case, increasing the diversity of the population by making it larger, does not work because the larger the population, the stronger is the bias toward the centroid of the swarm. Chances may be that newly designed optimization algorithm can solve these issues and new problems that have not been previously solved. An advancement of the currently developed *SOA* is defined next to find the optimal solution of multi-objective problems.

3 Seagull optimization algorithm (SOA)

In this section, the inspiration and mathematical modeling of proposed algorithm is discussed in detail.

3.1 Biological paradigm

Seagulls, scientific named as Laridae, are sea birds which can be found all over the planet. There is a wide range of seagulls species with different masses and lengths. Seagulls are omnivorous and eat insects, fish, reptiles, amphibians, earthworms, and so on. Body of most seagulls is covered with white plumage. Seagulls are very intelligent birds. They use bread crumbs to attract fish and produce rain-like sound with their feet to attract earthworms hidden under the ground. Seagulls can drink both fresh and salt water. Most of animals are unable to do this. However, seagulls have a special pair of glands right above their eyes which is specifically designed to flush the salt from their systems through openings in the bill.

Generally, seagulls live in colonies. They use their intelligence to find and attack the prey. The most important thing about the seagulls is their migrating and attacking behaviors. Migration is defined as the seasonal movement of seagulls from one place to another to find the richest and most abundant food sources that will provide adequate energy (Hoyo, Elliott, & Sargatal, 1996). This behavior is described as follows:

- During migration, they travel in a group. The initial positions of seagulls are different to avoid the collisions between each other.
- In a group, seagulls can travel towards the direction of best survival fittest seagull, i.e., a seagull whose fitness value[1] is low as compared to others.
- Based on the fittest seagull, other seagulls can update their initial positions.

Seagulls frequently attack migrating birds over the sea (Macdonald & Mason, 1973) when they migrate from one place to another. They can make their spiral natural shape movement during attacking. A conceptual model of these behaviors is illustrated in Fig. 1. These behaviors can be formulated in such a way that it can be associated with the objective function to be optimized. This makes it possible to formulate a new optimization algorithm. This paper focuses two natural behaviors of seagulls.

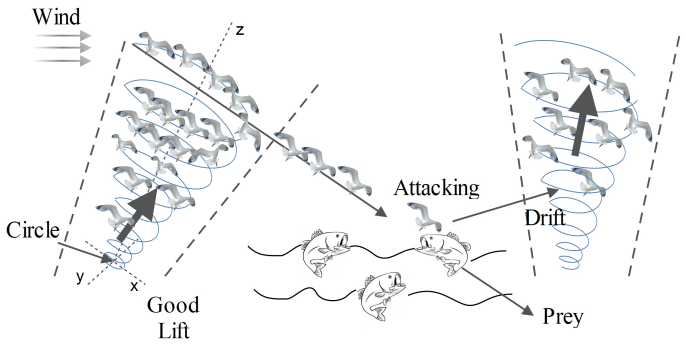

Fig. 1. Migration and attacking behaviors of seagulls.

3.2 Mathematical model

The mathematical models of migration and attacking the prey are discussed.

3.2.1 Migration (exploration). During migration, the algorithm simulates how the group of seagulls move towards one position to another. In this phase, a seagull should satisfy three conditions:

- **Avoiding the collisions:** To avoid the collision between neighbours (i.e., other seagulls), an additional variable A is employed for the calculation of new search agent position (see Fig. 2).

[1]The term fitness value is defined as a process which evaluates the population and gives a score or fitness. Whereas, the process is a function which measures the quality of the represented solution.

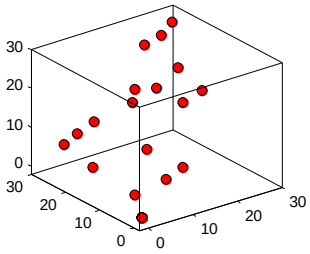

Fig. 2. Collision avoidance between search agents.

$$\vec{C_s} = A \times \vec{P_s}(x) \tag{4}$$

where $\vec{C_s}$ represents the position of search agent which does not collide with other search agent, $\vec{P_s}$ represents the current position of search agent, x indicates the current iteration, and A represents the movement behavior of search agent in a given search space.

$$\begin{aligned} &A = f_c - (x \times (f_c / Max_{iteration})) \\ &\text{where: } x = 0, 1, 2, \ldots, Max_{iteration} \end{aligned} \tag{5}$$

where f_c is introduced to control the frequency of employing variable A which is linearly decreased from f_c to 0. In this work, the value of f_c is set to 2.

- **Movement towards best neighbor's direction:** After avoiding the collision between neighbours, the search agents are move towards the direction of best neighbour (see Fig. 3).

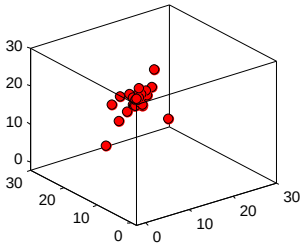

Fig. 3. Movement of search agents towards the best neighbour.

$$\vec{M_s} = B \times (\vec{P_{bs}}(x) - \vec{P_s}(x)) \tag{6}$$

where $\vec{M_s}$ represents the positions of search agent $\vec{P_s}$ towards the best fit search agent $\vec{P_{bs}}$ (i.e., fittest seagull). The behavior of B is randomized which is responsible for proper balancing between exploration and exploitation. B is calculated as:

$$B = 2 \times A^2 \times rd \tag{7}$$

where rd is a random number lies in the range of $[0,\ 1]$.

- **Remain close to the best search agent:** Lastly, the search agent can update its position with respect to best search agent which is shown in Fig. 4.

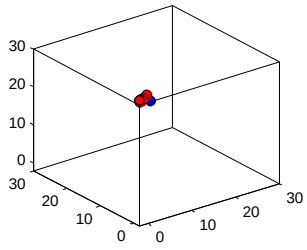

Fig. 4. Convergence towards the best search agent.

$$\vec{D_s} =\mid \vec{C_s} + \vec{M_s} \mid \tag{8}$$

where $\vec{D_s}$ represents the distance between the search agent and best fit search agent (i.e., best seagull whose fitness value is less).

3.2.2 Attacking (exploitation). The exploitation intends to exploit the history and experience of the search process. Seagulls can change the angle of attack continuously as well as speed during migration. They maintain their altitude using their wings and weight. While attacking the prey, the spiral movement behavior occurs in the air (see Fig. 5). This behavior in x, y, and z planes is described as follows.

$$x' = r \times cos(k) \tag{9}$$

$$y' = r \times sin(k) \tag{10}$$

$$z' = r \times k \tag{11}$$

$$r = u \times e^{kv} \tag{12}$$

where r is the radius of each turn of the spiral, k is a random number in range $[0 \leq k \leq 2\pi]$. u and v are constants to define the spiral shape, and e is the base of the natural logarithm. The updated position of search agent is calculated using Eqs. (8) - (12).

$$\vec{P_s}(x) = (\vec{D_s} \times x' \times y' \times z') + \vec{P_{bs}}(x) \tag{13}$$

where $\vec{P_s}(x)$ saves the best solution and updates the position of other search agents.

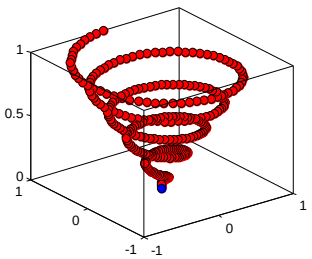

Fig. 5. Natural attacking behavior of seagull.

The proposed *SOA* starts with a random generated population. The search agents can update their positions with respect to best search agent during the iteration process. A is linearly decreased from f_c to 0. For smooth transition between exploration and exploitation, variable B is responsible. Hence, *SOA* is considered as a global optimizer (see Algorithm :) because of its better exploration and exploitation capability.

4 Proposed Multi-objective Seagull Optimization Algorithm (MOSOA)

4.1 Motivation

When resolving any metaheuristic problem of optimization, the proper balance between exploration and exploitation features empowers the optimization algorithm to find the best solutions. These algorithms are popular among researchers in their basic architecture, facility of implementation, and derivatives free mechanisms. However, the major disruption to these algorithms is that most algorithms can change control parameters. Another flaw is that these algorithms can not always converge with globally optimum, due to the stagnating situation with sub-optimal solutions throughout searching. The discovery process determines diversity by undertaking a global quest to produce new solutions in the quest and exploitation process by looking for neighborhood solutions (Chegini, Bagheri, & Najafi, 2018) to ensure optimum convergence. *MOSOA's* key principle of algorithm is based on *SOA* attacking and migration behaviors. The three components are used to build a *MOO* (Dhiman & Kumar, 2018a) version of the *SOA*, as shown in Fig. 6. This depicts the key integral being archive controller and grid, which reserves the optimal non-dominated *Pareto* solutions and the latter integral is a pioneer selection method for selecting the most effective mover from the archive with regard to prey orientation.

One of the important questions in working with textitMOSOA is why this algorithm needs to be created? The answer of this question can be given with the support of "No Free Lunch (NFL)" theorem (Wolpert & Macready, 1997), which states that the there is no metaheuristic to solve all type of optimization problems. This theorem is the basis for several developments in the field of metaheuristic and overall optimization.

4.2 Archive controller

All *POSs* that are best accessed are stored in a storage space, known as the archive. The controller determines whether to include a specific solution in the list. The Archive updation rules are given below:

- If the archive is found to be empty the current solution should be acknowledged.
- If an entity within the archive dominates some solution then the particular solution should be discarded.
- If the external population does not overpower the solution then the particular solution should be accepted and stored within the archive.
- If the new dimension dominates the solutions, they will be discarded from the list.

Algorithm : Seagull Optimization Algorithm

Input: Seagull population $\vec{P_s}$
Output: Optimal search agent $\vec{P_{bs}}$

procedure SOA
Initialize the parameters A, B, and $Max_{iteration}$
    Set $f_c \leftarrow 2$
    Set $u \leftarrow 1$
    Set $v \leftarrow 1$
    while $(x < Max_{iteration})$ **do**
        $\vec{P_{bs}} \leftarrow$**ComputeFitness(**$\vec{P_s}$**)** /* Calculate the fitness values of each search agent using **ComputeFitness** function*/
    /* Migration behavior */
        $rd \leftarrow Rand(0,\ 1)$ /* To generate the random number in range [0, 1] */
        $k \leftarrow Rand(0,\ 2\pi)$ /* To generate the random number in range [0, 2π] */
    /* Attacking behavior */
        $r \leftarrow u \times e^{kv}$ /* To generate the spiral behavior during migration */
        Calculate the distance $\vec{D_s}$ using Eq. (8)
        $P \leftarrow x' \times y' \times z'$ /* Compute x, y, z planes using Eqs. (9) - (12) */
        $\vec{P_s}(x) \leftarrow (\vec{D_s} \times P) + \vec{P_{bs}}$
        $x \leftarrow x + 1$
    end while
return $\vec{P_{bs}}$
end procedure

procedure ComputeFitness($\vec{P_s}$)
    for i ← 1 to n **do** /* Here, n represents the dimension of a given problem */
        $FIT_s[i] \leftarrow FitnessFunction(P_s(\vec{i},:))$ /* Calculate the fitness of each individual */
    end for
    $FIT_{s_{best}} \leftarrow$ **BEST**$(FIT_s[])$ /* Calculate the best fitness value using **BEST** function */
return $FIT_{s_{best}}$
end procedure

procedure BEST($FIT_s[]$)
    $Best \leftarrow FIT_s[0]$
    for i ← 1 to n **do**
        if$(FIT_s[i] < Best)$ **then**
            $Best \leftarrow FIT_s[i]$
        end if
    end for
return $Best$ /* Return the best fitness value */
end procedure

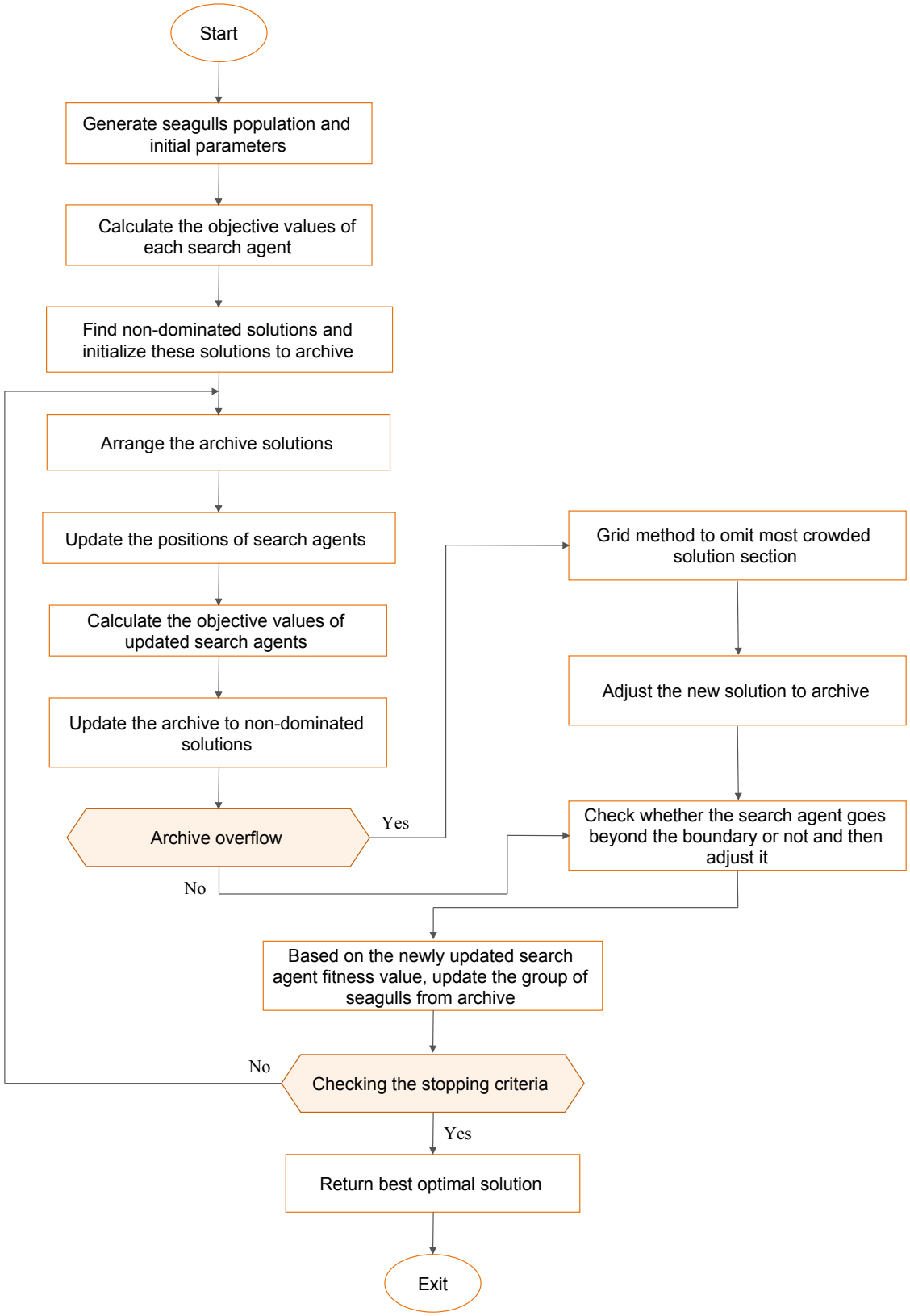

Fig. 6. Flowchart of the proposed MOSOA algorithm.

4.3 Grid

Adaptive grid approach (Knowles & Corne, 2000) produces distributions of the *Pareto* fronts. There are four regions for the objective function which is employed (see Figs. 7 - 8). The grid approach is used to compute individuals from the position of the developed population if it is outside the grid area. Given the uniform distribution of hypercubes, the grid space is created.

4.4 Leader selection mechanism

The key problem in multi-objective search space is to compare the new solutions in a given search space with existing solutions. Using a method for selecting the leader solves this problem. In this technique, the least crowded search space is filled by using the roulette-wheel selection method with one of the best solutions from the boundary of obtained optimum solutions (see Fig. 9). This approach is characterized in terms of:

$$U_k = \frac{g}{N_k} \tag{14}$$

such as g is a constant variable with value greater than 1 and N_k defines the count of *POSs* to kth segment. This method is a popularly used classical method which defines each individual's contribution using the proportion of the roulette wheel. The proposed *MOSOA* algorithm is an extension of the *SOA* algorithm, with multi-objectivity and search space distinction. *MOSOA* has an archive search space while *SOA* has to do the extra job of saving optimal solutions.

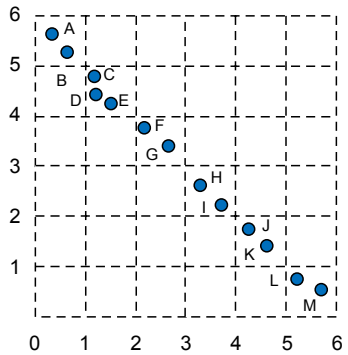

Fig. 7. Selection of individuals from grid.

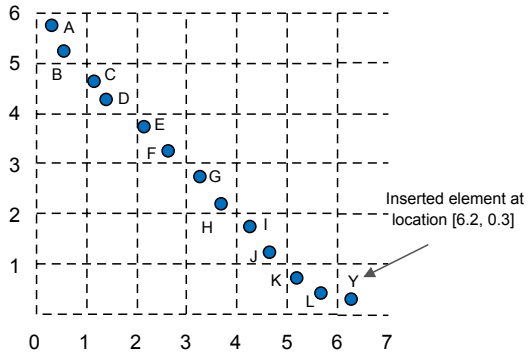

Fig. 8. Adjust and assign the individuals.

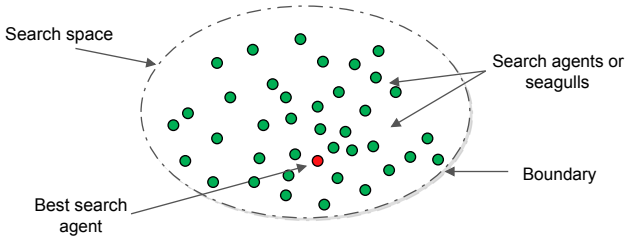

Fig. 9. Leader selection mechanism.

4.5 Computational complexity

In this section, we provide the computational efficiency of the proposed technique in detail.

4.5.1 Time complexity.

1. It takes $\mathcal{O}(n_o \times n_p)$ time to initialize the population, where n_p is the number of population size and n_o is the number of objectives
2. The fitness computation of search agents needs $\mathcal{O}(Max_{iteration} \times n_o \times n_p)$ time, where $Max_{iteration}$ represents the maximum number of iterations.
3. The proposed algorithm requires $\mathcal{O}(L)$ time for leader selection, where L indicates the computation time to find the leader.
4. It needs $\mathcal{O}(n_o \times (n_{ns} + n_p))$ time to update the archive of non-dominated solutions.
5. Steps 2 to 4 are repeated until the maximum iteration is reached.

Hence, the time complexity of *MOSOA* algorithm is $\mathcal{O}(Max_{iteration} \times n_o \times (n_p + n_{ns}) \times L)$.

Algorithm : Multi-objective Seagull Optimization Algorithm (MOSOA)

Input: Seagulls population
Output: Archive of non-dominated optimal solutions
procedure MOSOA
For each search agent, calculate their corresponding objective values
Find all the non-dominated solutions and initialize these solutions to archive
    while ($x < Max_{iteration}$) **do**
        for each search agent **do**
            Update the position of current search agent
        end for
        Compute the objective values of all search agents
        Find the non-dominated solutions from updated search agents
        Update the obtained non-dominated solutions to archive
        if archive is full **then**
            Grid method should be run to omit one of the most crowded archive members
            Add new solution to the archive
        end if
        Check if any search agent goes beyond the search space and then adjust it
        Compute the objective values of each search agent
        $x \leftarrow x + 1$
    end while
return archive
end procedure

4.5.2 Space complexity. The spatial complexity of the proposed *MOSOA* algorithm is determined as it initializes, i.e., when the population is generated in memory, which requires $\mathcal{O}(n_o \times n_p)$ time.

5 Experimental results and discussions

5.1 Experimental setup

The experimentation and algorithms are implemented in Matlab R2018a (8.3.0.532) version and run it in the environment of Microsoft Windows 8.1 with 64 bits on Core i-5 processor with 3.20 GHz and 8 GB memory.

5.2 Benchmark test functions

To demonstrate the efficiency of the proposed *MOSOA* algorithm, experiment on twenty-four well-known benchmark test functions is performed. The special session benchmark test suit, i.e., the *IEEE CEC-9* (Liu, Zou, Chen, & Wu, 2009), which includes ten unconstrained test functions, is used. These test functions and their characteristics are set out in Appendix A of *Supplementary Material*. The $UF1 - UF10$ is known as the unconstrained test functions in these test functions. These test functions have the potential to solve problems with the *MOO* bound constraint. The proposed algorithm is

also validated on $DTLZ$ and ZDT test functions, i.e., $DTLZ1 - DTLZ9$ (Deb et al., 2005) and $ZDT1 - ZDT6$ (Zitzler et al., 2000). Both these types of test functions, along with their comprehensive characteristics (see Table 1), are described in Appendices B and C of *Supplementary Material*, respectively.

5.3 Algorithms for comparison

The proposed algorithm is compared to six well-known optimization methods, such as: Multi-objective Particle Swarm Optimization (*MOPSO*) (Coello Coello & Lechuga, 2002), Non-dominated Sorting Genetic Algorithm 2 (*NSGA-II*) (Deb et al., 2002), Multi-objective Evolutionary Algorithm based on Decomposition (*MOEA/D*) (Q. Zhang & Li, 2007), Pareto Envelope based Selection Algorithm II (*PESA-II*) (Corne, Jerram, Knowles, & Oates, 2001), Multi-objective Spotted Hyena and Emperor Penguin Optimizer (*MOSHEPO*) (Dhiman, 2020), and Multi-objective Ant Colony Optimization (*MOACO*) (Angus & Woodward, 2009). According to their original versions, various parameters associated with these methods are same as mentioned in the literature. The following initial parameters for *MOPSO* algorithm are set as (Coello Coello & Lechuga, 2002):

- $\phi_a = \phi_b = 2.05$
- $\phi_f = \phi_a + \phi_b$
- Inertia weight: $w = \dfrac{2}{\phi_f - 2 + \sqrt{\phi_f^2 - 4\phi_f}}$
- Personal coefficient: $c_1 = \chi * \phi_a$
- Social coefficient: $c_2 = \chi * \phi_b$
- Grid inflation parameter: $\alpha = 0.1$
- Leader selection pressure parameter: $\beta = 4$
- Number of grids: $Grid_{number} = 10$

For *NSGA-II*, the following parameters are set as (Gong et al., 2008):

- Population size $(X) = 100$
- Cross over probability $P_c = 0.8$
- Mutation probability $P_m = 0.1$

For *MOEA/D*, the following parameters are chosen as (Q. Zhang & Li, 2007):

- Subproblems: $N = 100$
- Number of neighbours: $T = 0.1{*}N$
- Updated new child maximal copies: $M = 0.01{*}N$
- Probability of selecting parents: $P_p = 0.9$
- Mutation rates: $M_r = 0.5$
- Distribution index: $D_i = 30$

The following initial parameters are set for *PESA-II* (Gong et al., 2008):

- Cross over probability $P_c = 0.8$
- Distribution index for SBX = 15
- Mutation probability $P_m = 1/n$
- Polynomial mutation of distribution index = 20

For *MOACO*, the following initial parameters are set as (Ab Wahab, Nefti-Meziani, & Atyabi, 2015):

- Initial pheromone = 1.0E-06
- Pheromone update constant $Q = 20$
- Exploration constant $q_0 = 1$
- Global pheromone decay rate = 0.9
- local pheromone decay rate = 0.5
- $\alpha = 0.5$
- $\beta = 2.5$

5.4 Performance metrics

The four separate performance metrics have been selected to measure the efficiency of the proposed algorithm, as: *Hypervolume (HV)* (Zitzler & Thiele, 1999; Coello Coello, Dhaenens, & Jourdan, 2010), Δ_p ($p = 1$) (Rudolph, Schütze, Grimme, Domínguez-Medina, & Trautmann, 2016; Schütze, Esquivel, Lara, & Coello, 2012; Schütze, Laumanns, Tantar, Coello, & Talbi, 2010), *Spread* (Li & Zheng, 2009; Coello Coello et al., 2010), and *Epsilon (ε)* (Zitzler, Thiele, Laumanns, Fonseca, & da Fonseca, 2003; Coello Coello et al., 2010).

5.5 Performance evaluation

The results of the *MOSOA* algorithm are explained next on adopted benchmark test suites, such as: *IEEE CEC-9*, *ZDT*, and *DTLZ*.

5.5.1 Results based on the IEEE CEC-9 ($UF1 - UF10$) test functions. Comparison of the efficiency of the proposed methodology is performed in Table 2 using the *IEEE CEC-9* benchmark test problems with well-known approaches. Using the *MOSOA*, optimal solutions are calculated in terms of the $UF1$, $UF2$, $UF4$, $UF5$, and $UF6$. The *MOPSO* obtains strong spread performance in terms of the $UF3$ test method when compared with other techniques. By considering the Δ_p and *Epsilon* as efficiency metrics, the *MOEA/D* obtains promising results compared to competing algorithms. The *MOSOA* offers best performance in terms of the test functions of $UF7$, $UF9$, and $UF10$. For $UF7$ on Δ_p, *MOEA/D* outperforms than others. Compared with current techniques, the *NSGA-II* predicts the best value of Δ_p and *Epsilon* for the $UF8$ test function. For $UF9$, on Hypervolume and Δ_p performance metrics, $MOEA/D$ and $MOSHEPO$ obtain better results. For $UF10$, $NSGA-II$ and $PESA-II$ results are superior on *Hypervolume* and Δ_p performance metrics, respectively. Fig. 10 portrays the *MOSOA* non-dominated *Pareto* optimal solutions. It can be seen from this figure that the proposed *MOSOA* algorithm non-dominated *Pareto* solutions are very much similar to *Pareto* solutions of *IEEE CEC-9* benchmark test functions.

5.5.2 Results based on the ZDT ($ZDT1 - ZDT6$) test functions. Table 3 demonstrates the *MOSOA's* results using benchmark test problems of ZDT. For the ZDT test functions, for most cases the *MOSOA* outperforms other algorithms by considering the Δ_p, Spread, and Epsilon as efficiency metrics. For $ZDT1$, *MOPSO* efficiency is superior than others on Δ_p performance metric. Whereas, *MOSOA* obtains better results on rest of the metrics. For $ZDT2$ on *Spread* metric, *MOPSO* has good performance. For $ZDT3$ on *Hypervolume* metric, *PESA-II* obtains optimal results than others. For $ZDT4$

and $ZDT6$, on *Hypervolume* and Δ_p performance metrics, *MOPSO* and *MOEA/D* non-dominated solutions are better than competitor approaches, respectively. Fig. 11 shows the acquired *POSs* using current techniques. The *NSGA-II*, *MOEA/D*, and *MOSHEPO* achievement degrades for the benchmark function of $ZDT2$. The *MOSOA* offers better results for the $ZDT2$, $ZDT3$, and $ZDT6$ benchmark test functions.

5.5.3 Results based on the DTLZ ($DTLZ1 - DTLZ9$) test functions. The study results of *MOPSO*, *NSGA-II*, *MOEA/D*, *PESA-II*, *MOSHEPO*, and *MOACO* using the $DTLZ$ benchmark problem are shown in Table 4. The *MOSOA* offers promising statistically significant results in terms of the $DTLZ1$, $DTLZ2$, $DTLZ3$, $DTLZ5$, $DTLZ6$, and $DTLZ9$ benchmark functions as opposed to other current techniques. In terms of Δ_p and *Epsilon*, the *MOEA/D* and the *NSGA-II* outperform other methods as performance metrics for the $DTLZ4$ benchmark function. The *PESA-II* offers maximal *Hypervolume* and *Spread* values. For $DTLZ7$ and $DTLZ8$, *NSGA-II* and *MOPSO* algorithms are superior than others in terms of almost all performance metrics. The results explicitly state that this proposed algorithm successfully converges the $DTLZ$ benchmark test functions. Fig. 12 provides a comparison of the *MOSOA's* expected and actual optimal *Pareto* front.

Table 1
Characteristics of ZDT and DTLZ benchmark test functions.

Problems	Properties
ZDT1	Convex
ZDT4	Convex
ZDT2	Concave
ZDT6	Concave
DTLZ2	Concave
DTLZ3	Concave
DTLZ4	Concave
ZDT3	Disconnected
DTLZ7	Disconnected
DTLZ1	Linear
DTLZ5	-
DTLZ6	-

Table 2
The obtained results using proposed and competitor approaches on IEEE CEC-9 benchmark test functions.

F	Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
	Hypervolume	**3.17E+02**	3.70E-01	3.89E-01	6.50E-01	3.71E-01	1.63E-01	1.34E-01
		1.20E-04	1.90E-02	3.24E-03	2.70E-03	4.85E-02	1.10E-01	6.69E-02
UF1	Δ_p	**2.95E-10**	3.11E-04	2.30E-03	4.55E-04	5.30E-03	1.49E-02	2.55E-03
		6.99E-10	2.99E-04	6.22E-03	1.90E-04	1.60E-03	1.56E-03	1.80E-03
	Spread	**1.03E-02**	7.42E-01	2.13E+00	3.01E-01	1.30E+00	1.91E+00	1.00E+00
		1.11E-03	2.30E-01	1.12E-01	1.73E-01	1.51E-01	1.93E-01	1.91E-01
	Epsilon	**4.80E-04**	1.01E-01	2.13E-01	1.07E-02	1.64E-01	5.11E-01	1.71E-01
		1.09E-04	3.11E-02	4.40E-02	1.05E-02	1.12E-01	1.57E-01	1.10E-01
	Hypervolume	**1.95E+01**	5.00E-01	5.00E-01	5.41E-01	4.82E-01	4.15E-01	3.04E-01
		1.21E-04	1.03E-03	6.00E-03	2.77E-03	1.12E-02	1.95E-01	1.16E-01
UF2	Δ_p	**2.12E-07**	1.50E-03	4.77E-03	3.41E-04	2.62E-03	2.20E-03	2.11E-03
		1.10E-07	2.85E-05	4.62E-04	1.74E-04	4.85E-04	1.10E-03	1.97E-03
	Spread	**2.46E-03**	3.97E-01	5.03E-01	3.21E-01	6.38E-01	2.02E+00	2.15E-01
		2.92E-03	1.22E-01	3.71E-02	4.50E-02	7.11E-02	2.22E-01	1.48E-01
	Epsilon	**2.04E-03**	8.22E-02	2.30E-01	4.81E-02	1.78E-01	2.34E-01	3.70E-02
		4.75E-05	8.72E-03	2.33E-02	1.26E-02	4.60E-02	1.51E-01	2.42E-03
	Hypervolume	3.18E-01	1.75E-00	1.13E-00	4.03E-00	1.44E-00	**1.01E+01**	2.63E-00
		2.71E-01	2.50E-02	2.58E-02	2.21E-02	2.35E-02	2.74E-02	**1.20E-02**
UF3	Δ_p	1.62E-02	4.87E-03	7.95E-03	**1.05E-03**	2.53E-02	2.47E-02	1.87E-02
		1.60E-02	**3.30E-04**	1.61E-03	4.00E-04	1.88E-03	1.98E-03	1.41E-03
	Spread	7.01E-00	**4.23E-01**	2.31E+00	4.37E-01	2.00E+00	1.08E+00	2.47E+00
		5.98E-01	3.18E-01	4.46E-02	2.04E-01	1.02E-01	5.38E-02	**1.75E-02**
	Epsilon	2.40E-00	2.83E-01	2.83E-01	**1.30E-01**	3.77E-01	4.54E-01	2.95E-01
		2.11E-01	**2.43E-03**	3.53E-02	4.10E-02	4.30E-02	6.47E-02	3.47E-02
	Hypervolume	**1.01E+01**	1.41E-01	1.45E-01	1.36E-01	1.41E-01	1.51E-01	1.40E-01
		1.31E-04	6.70E-03	2.18E-03	1.00E-02	2.12E-03	5.51E-02	1.96E-02
UF4	Δ_p	**3.14E-05**	1.65E-03	1.30E-03	2.88E-03	1.38E-03	4.61E-03	3.98E-03
		1.68E-06	1.03E-04	3.44E-05	4.68E-04	1.31E-04	2.63E-03	1.80E-04
	Spread	**1.01E-02**	3.27E-01	3.12E-01	3.02E-01	7.23E-01	2.03E+00	2.75E-01
		2.12E-03	5.82E-02	3.13E-02	8.45E-02	4.31E-02	3.22E-01	3.23E-02
	Epsilon	**1.11E-03**	6.47E-02	4.98E-02	5.80E-02	7.34E-02	1.61E-01	2.71E-02
		2.32E-04	4.28E-02	7.14E-03	8.13E-03	7.60E-03	2.11E-01	5.84E-03
	Hypervolume	**1.61E+01**	2.50E-03	1.71E-02	3.71E-02	5.21E-02	2.95E-02	2.49E-02
		1.09E-03	1.56E-02	3.05E-02	3.61E-02	5.00E-02	3.68E-01	1.48E-02
UF5	Δ_p	**1.00E-02**	1.62E-01	1.88E-01	2.04E-01	1.99E-01	3.26E-00	2.41E-01
		1.07E-03	1.51E-01	2.88E-02	1.90E-02	1.37E-02	2.76E-01	1.98E-02
	Spread	**1.74E-02**	5.56E-01	1.25E+00	1.21E+00	1.13E+00	3.82E+00	1.20E+00
		3.77E-03	7.01E-02	8.94E-02	4.27E-02	1.33E-01	6.38E-02	3.72E-02
	Epsilon	**2.17E-02**	2.26E+00	6.91E-01	6.41E-01	7.80E-01	9.71E-01	4.96E-01
		1.06E-02	4.27E-01	1.58E-01	1.42E-01	1.75E-01	1.59E-01	2.46E-01
	Hypervolume	**2.41E+00**	6.02E-02	2.95E-01	3.07E-01	2.09E-01	4.14E-01	3.88E-01
		1.37E-03	2.47E-02	5.98E-02	9.23E-02	4.39E-02	7.60E-02	3.20E-02
UF6	Δ_p	**3.30E-04**	8.55E-03	4.67E-03	2.13E-03	2.92E-02	5.22E-02	7.79E-02
		1.30E-04	3.55E-03	1.50E-03	1.66E-03	2.78E-03	1.44E-03	7.28E-03
	Spread	**6.08E-02**	8.78E-01	1.18E+00	1.13E+00	1.26E+00	1.11E+00	1.10E+00
		3.41E-03	5.73E-02	1.23E-01	8.46E-02	2.69E-01	8.27E-02	4.40E-02
	Epsilon	**1.01E-02**	4.94E-01	4.38E-01	2.54E-01	4.47E-01	7.71E-01	4.38E-01
		1.01E-03	1.81E-01	1.48E-01	1.47E-01	1.90E-01	2.66E-01	1.08E-01
	Hypervolume	1.32E-01	2.94E-01	2.28E-01	3.90E-01	1.48E-01	**1.21E+01**	1.85E-01
		5.35E-02	2.56E-03	7.84E-03	**2.33E-05**	7.81E-05	7.81E-03	2.98E-03
UF7	Δ_p	2.60E-02	5.37E-03	6.77E-03	**1.50E-04**	5.48E-02	2.17E-02	3.36E-02
		1.44E-02	1.66E-03	1.05E-03	**1.43E-05**	1.70E-03	2.31E-03	1.74E-03
	Spread	**1.80E-01**	7.51E-01	1.06E+00	2.48E-01	1.17E+00	1.00E+00	4.60E-00
		1.16E-03	5.62E-02	1.10E-01	1.74E-01	7.76E-02	1.57E-02	2.57E-02
	Epsilon	**1.04E-03**	1.81E-01	3.27E-01	3.41E-02	5.07E-01	7.00E-01	2.61E-01
		2.56E-04	1.06E-01	1.41E-01	2.16E-02	1.18E-01	1.36E-01	1.80E-01
	Hypervolume	**2.58E+01**	2.04E-02	2.26E-01	1.80E-01	1.92E-02	1.28E-01	1.01E-01
		2.57E-04	1.68E-02	3.06E-02	1.00E-01	2.31E-02	7.95E-02	1.78E-02
UF8	Δ_p	2.33E-02	1.91E-03	**1.80E-03**	2.15E-01	4.10E-03	3.86E-03	4.94E-03
		5.10E-03	1.00E-04	**3.44E-05**	4.16E-03	3.85E-04	8.08E-04	1.85E-03
	Spread	5.06E-00	8.15E-01	6.76E-01	**3.68E-01**	6.06E-01	7.61E-01	4.78E-01
		4.12E-00	7.21E-02	**7.04E-02**	6.91E-01	1.10E-01	1.28E-01	1.57E-01
	Epsilon	7.02E-00	6.68E-01	**5.23E-01**	6.00E-01	8.71E-01	7.04E-01	5.96E-01
		2.13E-00	7.50E-02	1.10E-01	8.53E-01	**4.21E-02**	1.30E-01	2.65E-01
	Hypervolume	3.08E-01	5.88E-02	1.53E-01	**1.75E+02**	8.42E-02	3.56E-01	2.38E-01
		5.32E-00	4.71E-02	5.16E-02	3.46E-01	2.50E-02	3.77E-02	**1.10E-02**
UF9	Δ_p	3.38E-02	1.67E-03	4.34E-03	5.91E-03	3.77E-03	**1.21E-03**	2.22E-03
		5.11E-03	2.26E-04	1.50E-04	3.47E-03	2.30E-04	**1.03E-04**	1.75E-03
	Spread	**1.00E-02**	7.20E-01	7.51E-01	5.62E-01	6.54E-01	6.35E-01	3.58E-01
		6.36E-01	4.04E-02	5.82E-02	5.40E-02	6.44E-02	5.11E-02	**2.45E-02**
	Epsilon	**2.02E-02**	7.20E-01	4.13E-01	5.55E-01	7.94E-01	3.80E-01	5.84E-01
		4.14E-01	1.41E-01	5.45E-02	4.76E-02	3.15E-02	**1.91E-02**	4.67E-02
	Hypervolume	3.61E-01	1.00E+00	**3.52E+01**	5.91E-02	3.04E-02	2.37E-02	3.91E-02
		1.30E-01	0.00E+00	**1.15E-02**	4.10E-01	1.96E-02	1.85E-02	2.01E-02
UF10	Δ_p	5.77E-02	2.78E-02	2.12E-03	3.00E-03	**1.82E-03**	5.51E-03	3.21E-03
		3.90E-02	5.72E-03	2.01E-03	1.80E-03	3.17E-04	**2.46E-04**	1.45E-03
	Spread	**1.61E-02**	5.52E-01	6.65E-01	4.20E-01	8.17E-01	1.23E+00	3.52E-01
		4.81E-01	**4.06E-02**	5.27E-02	7.61E-01	8.12E-02	1.22E-01	1.85E-01
	Epsilon	**3.60E-02**	1.81E+00	1.04E+00	4.32E-01	7.85E-01	7.67E-01	5.01E-01
		4.52E-00	1.44E-01	1.54E-01	6.16E-01	1.06E-01	**1.02E-01**	1.84E-01

Table 3
The obtained optimal results using proposed and competitor approaches on ZDT benchmark test functions.

F	Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
	Hypervolume	**5.55E+01**	5.61E-01	5.61E-01	5.63E-01	5.55E-01	5.61E-01	5.61E-01
		2.81E-03	**1.41E-04**	1.61E-04	2.55E-04	7.57E-04	2.04E-04	1.45E-04
ZDT1	Δ_p	2.33E-03	1.32E-04	1.17E-04	**5.31E-05**	3.02E-04	1.10E-04	3.65E-04
		3.02E-06	**4.01E-07**	2.36E-06	1.43E-06	1.43E-04	2.20E-06	1.01E-06
	Spread	**1.31E-03**	5.86E-02	2.76E-01	1.84E-01	5.48E-01	1.42E-01	1.30E-01
		1.10E-05	1.03E-02	1.57E-02	1.81E-03	2.77E-02	1.61E-02	1.21E-02
	Epsilon	4.48E-03	4.86E-03	1.33E-02	**2.02E-03**	1.30E-02	7.91E-03	6.07E-02
		4.72E-03	**1.57E-04**	1.01E-03	2.81E-04	1.17E-02	5.67E-04	2.40E-04
	Hypervolume	2.28E+01	2.27E-01	2.25E-01	2.58E-01	**2.12E-02**	2.25E-01	2.26E-01
		2.70E-06	7.08E-05	2.10E-04	1.57E-04	5.53E-04	5.13E-04	1.08E-04
ZDT2	Δ_p	**1.02E-06**	1.44E-04	1.36E-04	3.97E-05	2.51E-04	1.21E-04	1.31E-04
		1.33E-07	1.56E-06	2.40E-05	5.01E-06	1.34E-05	3.20E-06	1.75E-06
	Spread	1.01E-00	**5.67E-02**	3.60E-01	1.83E-01	5.70E-01	1.48E-01	1.55E-01
		8.11E-03	**6.84E-03**	2.51E-02	1.49E-02	3.21E-02	1.45E-02	2.86E-02
	Epsilon	**1.21E-04**	4.45E-03	1.82E-02	1.67E-03	2.28E-02	7.48E-03	2.06E-03
		1.78E-03	**1.35E-04**	1.31E-03	2.61E-04	1.87E-02	5.15E-04	7.72E-04
	Hypervolume	4.11E-00	4.13E-01	4.14E-01	4.15E-01	**4.10E+01**	4.13E-01	4.15E-01
		1.21E-03	2.86E-04	2.49E-04	**2.08E-05**	1.81E-03	1.30E-04	4.81E-04
ZDT3	Δ_p	**3.12E-06**	1.56E-04	1.32E-04	3.13E-05	1.01E-03	2.20E-04	1.01E-04
		2.30E-05	**1.07E-06**	3.48E-06	4.54E-06	4.28E-04	1.87E-05	3.73E-05
	Spread	**1.00E-02**	6.04E-01	6.41E-01	2.13E+00	8.01E-01	7.37E-01	5.05E-01
		1.16E-04	2.37E-03	1.31E-02	2.37E-03	1.61E-02	3.73E-03	1.52E-03
	Epsilon	**2.90E-04**	4.70E-03	7.86E-03	3.10E-03	2.42E-01	8.61E-03	4.93E-03
		5.73E-03	4.48E-04	1.55E-03	**1.24E-05**	1.53E-01	1.13E-03	1.47E-03
	Hypervolume	5.61E-02	**1.00E+01**	5.47E-01	5.64E-01	5.51E-01	5.48E-01	5.44E-01
		3.43E-04	**1.00E-07**	2.52E-03	3.94E-05	3.63E-03	7.61E-03	2.45E-02
ZDT4	Δ_p	3.21E-04	2.40E-01	2.77E-04	**3.10E-05**	2.11E-04	3.70E-03	1.68E-04
		1.81E-03	1.37E-02	4.98E-05	**2.03E-05**	1.11E-04	1.21E-03	1.01E-04
	Spread	**1.01E-02**	7.85E-01	2.58E-01	1.29E-01	8.11E-01	2.05E-01	1.07E-01
		1.97E-04	6.17E-02	2.14E-02	2.01E-03	1.87E-01	1.35E-01	1.06E-02
	Epsilon	**1.08E-04**	4.13E+00	1.75E-02	1.93E-03	1.86E-02	5.36E-02	2.27E-02
		1.00E-05	1.90E+00	7.11E-03	1.02E-04	1.05E-02	4.28E-02	1.72E-02
	Hypervolume	**2.18E+00**	4.56E-02	3.81E-01	3.04E-02	2.90E-01	2.77E-02	2.25E-01
		3.32E-06	4.41E-05	1.45E-03	**5.10E-08**	1.36E-06	1.71E-03	1.38E-03
ZDT6	Δ_p	1.71E-03	1.36E-03	3.80E-04	**2.60E-05**	2.06E-04	3.94E-04	2.26E-04
		3.10E-07	1.08E-06	6.06E-05	**5.40E-08**	2.81E-05	2.36E-05	1.31E-05
	Spread	**1.10E-02**	7.00E-01	2.65E-01	1.69E-01	6.58E-01	1.37E-01	2.21E-01
		1.23E-05	3.78E-01	2.98E-02	1.62E-04	1.92E-01	2.01E-02	1.62E-02
	Epsilon	**1.21E-04**	3.85E-03	1.50E-02	1.63E-03	1.36E-02	1.55E-02	2.00E-02
		2.12E-03	4.15E-04	2.16E-03	1.42E-01	2.65E-03	**4.45E-05**	1.16E-04

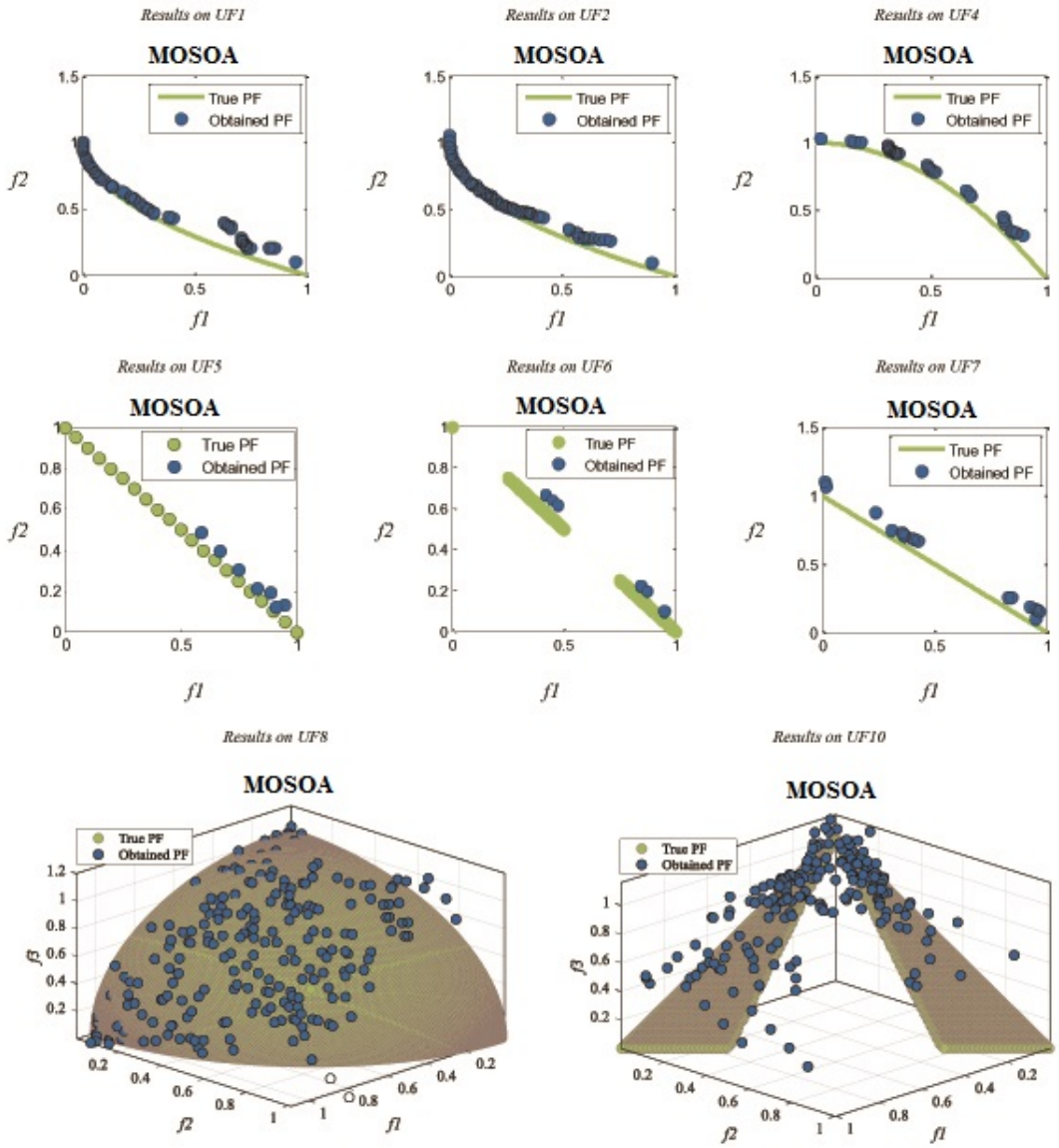

Fig. 10. The obtained Pareto solutions by the MOSOA technique on the IEEE CEC-9 test functions.

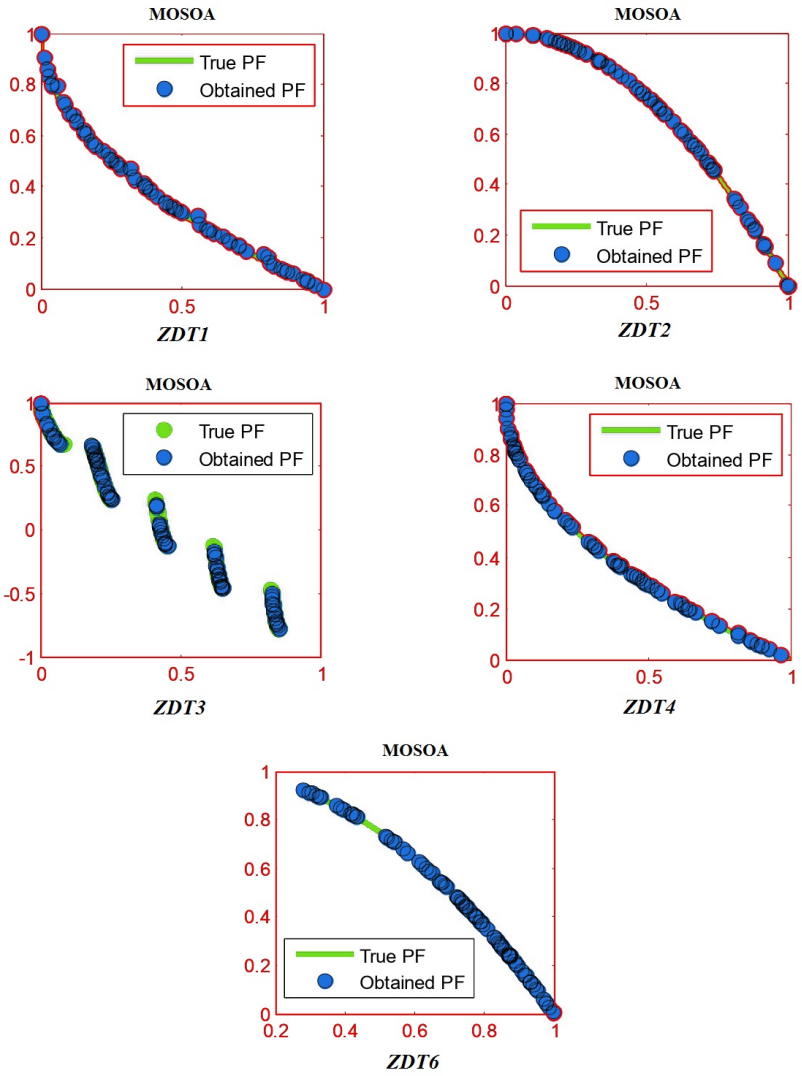

Fig. 11. The obtained Pareto solutions by the MOSOA technique on the ZDT test functions.

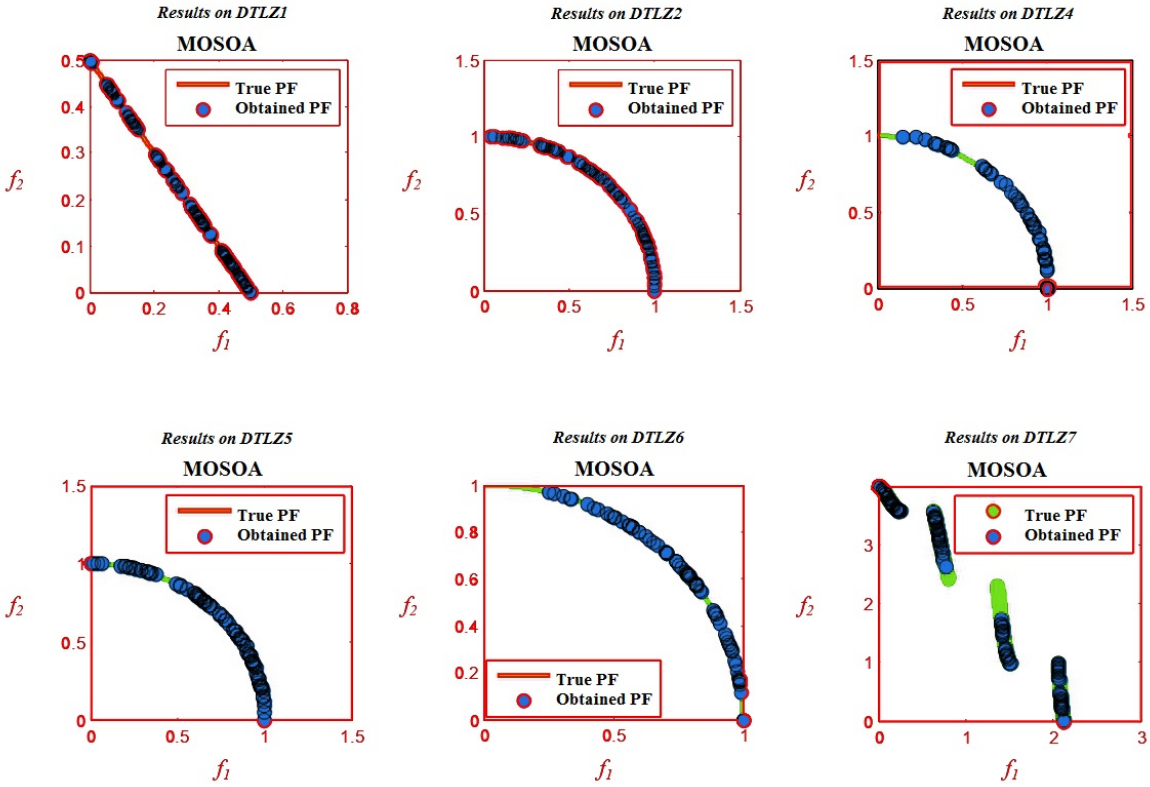

Fig. 12. The obtained Pareto solutions by the MOSOA technique on the DTLZ test functions.

5.6 Sensitivity analysis

5.6.1 Effect of archive on the MOSOA algorithm. Table 5 portrays repeated iterations of the archive assessment. The $ZDT1$, $DTLZ6$, and $ZDT3$ are used as test functions for demonstrating the archive's effect on the proposed algorithm. The $ZDT1$, $DTLZ6$, and $ZDT3$ test functions have concave, convex, and consecutively disconnected properties. The archive size is deemed as 10. Fig. 13 reflects the *MOSOA's* comprehensive converging behavior. It is observed that optimal values for the various test functions are obtained using the *MOSOA*.

5.6.2 Effect of selection mechanism on the MOSOA algorithm. The output of the proposed *MOSOA* algorithm is evaluated using the approaches to pick the roulette wheel and the tournament. The test functions of $ZDT1$, $ZDT3$, and $ZDT6$ are used to obtain prediction over the output of the proposed *MOSOA*. In Fig. 14, roulette wheel and tournament selection strategies converging behavior, are demonstrated. From this figure it can be concluded that the performance of the approach to roulette wheel selection is better than the approach to tournament selection as regards convergence towards the optimal solution.

Table 4

The obtained optimal results using proposed and competitor approaches on DTLZ benchmark test functions.

F	Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
	Hypervolume	3.61E-02	2.47E-01	5.36E-01	**2.55E+01**	4.58E-01	6.55E-01	3.10E-01
		7.32E-03	4.55E-02	2.24E-01	**4.93E-03**	2.72E-02	1.96E-02	1.01E-02
DTLZ1	Δ_p	**2.44E-05**	1.52E-03	2.77E-03	3.23E-04	4.01E-03	3.45E-04	4.78E-04
		1.10E-04	1.48E-04	2.80E-04	4.41E-05	1.01E-04	**6.20E-06**	3.10E-04
	Spread	**1.32E-02**	4.94E-01	1.01E+00	1.84E-01	2.71E+00	6.48E-01	3.96E-01
		2.16E-00	2.91E-00	**2.01E-01**	3.23E-01	2.88E-01	2.57E-00	2.77E-01
	Epsilon	2.61E-02	2.95E-02	1.46E-03	3.34E-02	**1.03E-03**	4.25E-03	1.06E-02
		3.71E-01	2.92E-02	4.22E-01	1.61E-02	1.66E-01	**1.48E-02**	1.01E-02
	Hypervolume	**1.08E+01**	1.93E-01	1.62E-02	2.03E-01	4.26E-02	3.03E-01	2.96E-01
		1.06E-04	1.62E-03	2.95E-03	4.20E-02	1.11E-02	1.86E-03	1.14E-03
DTLZ2	Δ_p	**2.53E-05**	2.47E-03	2.66E-04	5.91E-04	2.18E-02	3.72E-04	3.94E-04
		2.10E-06	7.05E-04	3.00E-05	5.41E-05	4.38E-03	3.10E-05	3.02E-05
	Spread	**2.31E-02**	5.45E-01	3.82E-01	6.41E-01	6.55E-01	4.45E-01	3.82E-01
		4.77E-04	1.72E-01	6.53E-03	1.40E-02	4.72E-01	2.04E-02	5.27E-02
	Epsilon	**2.68E-03**	5.61E-02	1.36E-01	3.03E-02	2.72E-01	7.40E-02	5.04E-02
		5.15E-01	**7.21E-03**	5.36E-02	2.56E-01	3.72E-01	1.12E-02	3.73E-02
	Hypervolume	1.08E-01	3.71E-01	5.23E-02	3.62E-01	4.22E-01	**1.00E+00**	3.02E-01
		7.36E-02	1.62E-02	4.24E-04	1.73E-02	1.34E-03	**0.00E+00**	4.56E-03
DTLZ3	Δ_p	**2.46E-05**	4.21E-03	4.04E-04	1.48E-03	1.21E-01	1.01E-01	2.03E-02
		1.11E-05	2.13E-03	2.32E-04	5.40E-04	2.91E-02	3.58E-02	1.40E-02
	Spread	**1.38E-04**	3.61E-02	1.31E+00	3.62E-02	1.42E-03	1.21E+00	1.91E-02
		3.43E-01	2.72E-01	6.25E-02	**1.62E-02**	4.21E-01	1.08E-01	1.62E-01
	Epsilon	**2.17E-03**	2.73E-01	4.72E-01	2.83E-02	2.21E-01	5.36E+00	2.21E-01
		1.43E-03	2.26E-02	4.76E-02	5.62E-02	7.73E-02	1.71E+00	3.82E-02
	Hypervolume	1.61E-01	1.73E-01	2.32E-02	1.73E-01	**2.41E+01**	2.11E-01	3.95E-02
		1.30E-01	**2.72E-03**	5.88E-02	2.62E-02	4.22E-03	8.65E-03	4.26E-03
DTLZ4	Δ_p	1.32E-03	3.60E-02	2.68E-02	**1.13E-04**	3.78E-02	3.34E-03	2.15E-02
		3.10E-06	1.01E-04	4.30E-03	1.08E-04	5.48E-03	3.42E-03	3.91E-03
	Spread	1.77E-01	2.07E-01	4.72E-02	5.63E-01	**4.13E-03**	2.56E-01	4.04E-02
		4.10E-01	2.52E-02	5.62E-01	**2.20E-02**	4.54E-02	1.28E-01	3.81E-02
	Epsilon	8.04E-03	3.01E-02	**5.72E-03**	2.62E-02	1.41E-01	2.22E-01	3.61E-02
		7.91E-03	4.51E-02	8.13E-03	7.62E-02	**7.32E-03**	2.74E-01	2.93E-02
	Hypervolume	**1.48E+01**	1.62E-03	2.12E-02	4.23E-01	4.92E-02	1.95E-01	2.32E-02
		1.31E-05	1.82E-04	4.05E-02	2.82E-02	4.22E-02	1.46E-02	1.98E-02
DTLZ5	Δ_p	**1.34E-04**	1.34E-01	2.20E-01	3.52E-03	2.61E-02	2.76E-01	1.91E-02
		2.71E-05	6.35E-02	2.61E-02	5.87E-04	2.25E-02	1.30E-02	1.03E-03
	Spread	2.80E-01	4.62E-01	1.35E+00	4.25E-01	1.03E+00	**1.42E-02**	6.21E-01
		4.01E-03	4.61E-02	6.95E-03	4.21E-02	**3.13E-04**	4.46E-02	3.28E-02
	Epsilon	**2.13E-03**	2.72E+00	3.10E-01	5.93E-01	3.62E-01	6.74E-01	3.05E-01
		2.45E-00	3.16E-01	1.68E-01	3.89E-01	**1.01E-01**	2.21E-01	1.94E-01
	Hypervolume	**1.62E+00**	2.01E-03	1.91E-01	3.12E-02	1.17E-01	1.04E-01	1.52E-02
		1.10E-04	1.55E-02	6.82E-02	1.24E-03	3.42E-02	4.51E-02	1.01E-02
DTLZ6	Δ_p	**2.01E-03**	1.32E-02	2.51E-02	4.20E-03	3.26E-02	2.10E-02	1.11E-02
		1.66E-04	2.65E-03	4.11E-03	2.61E-03	5.35E-03	2.11E-03	2.84E-03
	Spread	2.44E-00	4.17E-01	2.41E+00	3.11E-01	**1.25E-01**	1.11E+00	1.41E-01
		5.76E-01	**1.20E-02**	2.82E-01	4.06E-01	1.62E-01	6.17E-02	3.74E-02
	Epsilon	**1.46E-04**	5.94E-01	3.62E-03	2.64E-02	2.42E-01	6.71E-01	3.33E-02
		6.15E-02	1.81E-03	1.51E-01	**1.36E-04**	5.35E-01	1.72E-01	5.81E-03
	Hypervolume	1.08E-03	2.52E-01	4.17E-03	3.10E-01	**2.48E+00**	1.50E-01	3.11E-02
		1.31E-01	2.62E-02	**2.08E-04**	2.82E-03	4.81E-01	6.25E-02	6.71E-03
DTLZ7	Δ_p	3.11E-02	4.51E-03	5.97E-03	**2.37E-03**	3.90E-02	2.67E-02	5.41E-03
		2.76E-03	1.12E-03	**2.47E-04**	2.26E-03	5.93E-03	1.80E-03	2.91E-03
	Spread	3.66E-01	4.41E-01	1.82E+00	2.23E-01	**2.72E-03**	1.02E-01	1.27E-02
		5.24E-03	7.03E-02	1.52E-03	2.36E-01	**1.14E-04**	2.90E-02	4.41E-03
	Epsilon	6.77E-01	7.62E-01	**1.16E-02**	2.62E-02	2.07E-01	3.03E-01	1.27E-01
		4.17E-03	1.06E-01	**1.62E-04**	2.34E-02	2.32E-02	1.92E-01	3.31E-02
	Hypervolume	6.61E-02	**3.51E+01**	1.83E-01	3.87E-03	1.02E-02	1.97E-01	1.38E-01
		1.46E-03	**1.68E-04**	2.93E-02	1.00E-02	3.22E-02	5.92E-02	1.75E-03
DTLZ8	Δ_p	5.45E-02	**2.17E-04**	4.02E-03	3.74E-02	3.68E-03	2.10E-03	3.52E-03
		8.80E-03	1.01E-03	1.67E-03	1.31E-03	**5.67E-04**	1.46E-03	7.94E-04
	Spread	2.86E-01	2.26E-01	**1.36E-03**	2.38E-01	4.06E-02	4.01E-02	1.36E-02
		6.76E-01	2.21E-02	7.52E-03	2.91E-01	2.13E-02	1.98E-02	**3.04E-03**
	Epsilon	1.00E-01	2.68E-01	**5.24E-03**	2.00E-01	2.23E-02	5.08E-01	1.22E-02
		5.18E-01	2.50E-02	**1.23E-03**	3.22E-01	1.21E-02	1.81E-01	5.96E-03
	Hypervolume	**1.61E+02**	5.88E-02	2.34E-03	2.75E-02	4.42E-02	2.82E-01	1.97E-01
		1.35E-03	4.71E-02	5.16E-02	3.46E-01	2.50E-02	1.93E-02	7.75E-02
DTLZ9	Δ_p	**1.46E-04**	4.46E-03	5.76E-02	2.23E-03	5.88E-03	4.46E-03	6.03E-03
		2.61E-05	2.57E-03	3.32E-03	1.86E-03	3.31E-04	2.90E-04	8.42E-04
	Spread	8.86E-01	3.20E-01	**3.51E-03**	5.75E-01	2.54E-02	3.15E-01	4.08E-02
		2.77E-01	7.87E-02	2.82E-01	5.30E-02	**7.44E-03**	3.93E-01	8.80E-03
	Epsilon	8.57E-01	3.20E-01	**6.72E-03**	5.05E-01	6.66E-02	2.40E-02	3.37E-02
		3.98E-01	7.41E-01	**5.92E-03**	6.76E-02	5.95E-01	1.43E-02	2.78E-02

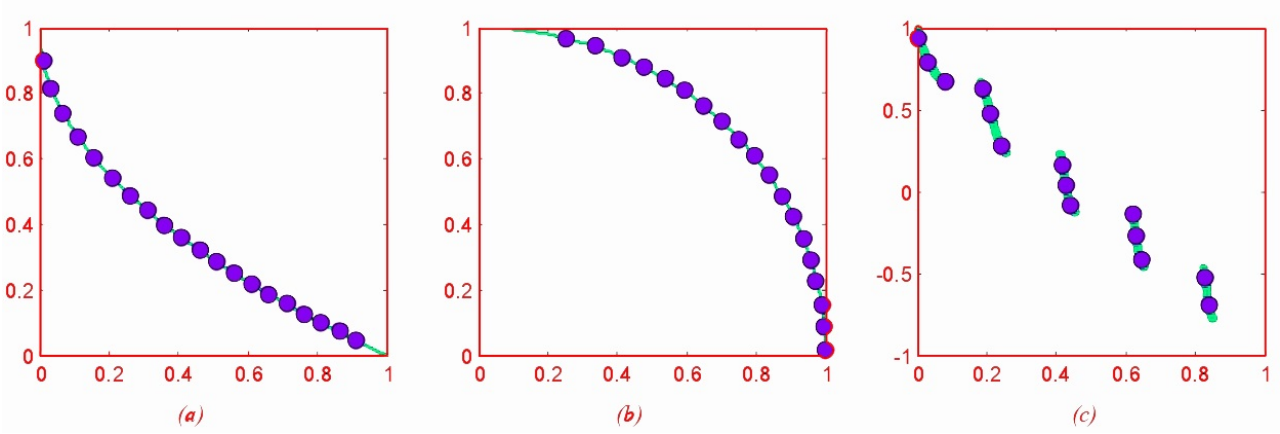

Fig. 13. Archive measurement of convergence on different Pareto fronts, i.e., (a) ZDT1, (b) ZDT6, and (c) ZDT3.

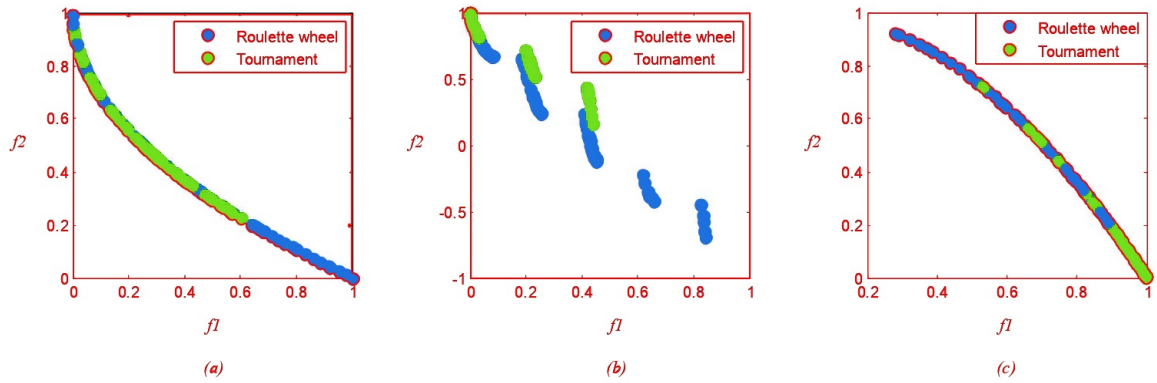

Fig. 14. The convergence of selection approaches on (a) ZDT1, (b) ZDT3, and (c) ZDT6.

5.7 Wilcoxon signed-rank test

In the literature (Roy, Islam, Murase, & Yao, 2015), it can be seen that the output indicators do not provide any guarantee for better convergence and diversity, since sometimes the solutions obtained are not similar to the optimal front of *Pareto*. The *Wilcoxon signed-rank* test (Richardson, 2010) is performed on the average value of the output measures *Hypervolume*, Δ_p, *Spread*, and *Epsilon*. For each question the difference between each pair of average results is determined. These differences are sorted in an ascending order and assigned a rank ranging from the smallest to the largest number. If the proposed algorithm is better than the competitor algorithms with respect to a particular performance criterion, the positive rank is given. Otherwise, it is assigned negative rank. A value amount is set to 0.10 for contrast, and summarizes both the positive and negative rank (Richardson, 2010). The findings of the *Wilcoxon* test are shown in Table 6, where +, -, and = indicate that *MOSOA's* output is superior, inferior, and equal to competitor algorithms. From Table 6 it is observed that *MOSOA* surpasses all the competitor algorithms except

Table 5
The archive values obtained by MOSOA algorithm.

Iterations	ZDT1 (*Concave*)				DTLZ6 (*Convex*)				ZDT3 (*Disconnected*)			
	Archive		Objective value		Archive		Objective value		Archive		Objective value	
	x	y	f_1	f_2	x	y	f_1	f_2	x	y	f_1	f_2
1	0.981	0.013	0.658	0.729	0.055	3.409	0.441	0.179	0.788	-0.189	0.248	-0.161
	0.782	-0.051			-0.199	5.779			0.776	-0.035		
	0.744	0.087			-0.197	5.778			0.742	0.087		
	0.711	0.218			-0.178	5.128			0.695	0.217		
	0.621	0.338			-0.176	5.128			0.633	0.329		
	0.547	0.460			-0.198	5.782			0.546	0.446		
	0.437	0.555			-0.114	3.906			0.436	0.547		
	0.326	0.638			-0.175	5.132			0.231	0.638		
	0.200	0.710			0.044	3.408			0.200	0.696		
	0.057	0.760			-0.158	5.131			0.069	0.754		
50	0.637	-0.189	0.560	0.548	-0.075	1.822	0.384	0.031	0.616	-0.19	0.100	-0.161
	0.657	-0.063			-0.179	3.031			0.552	0.054		
	0.602	-0.087			-0.175	3.820			0.676	0.040		
	0.640	0.141			-0.169	4.029			0.548	0.172		
	0.53	0.211			-0.173	2.804			0.531	0.137		
	0.426	0.315			-0.12	4.257			0.484	0.323		
	0.347	0.230			-0.079	2.888			0.551	0.216		
	0.238	0.542			-0.178	2.822			0.133	0.571		
	0.202	0.626			0.011	2.017			0.193	0.636		
	0.067	0.558			-0.162	4.212			0.017	0.540		
100	-0.174	0.701	0.569	0.356	0.72	-0.053	0.344	0.761	-0.181	0.724	0.059	0.71
	-0.089	0.464			0.773	-0.055			-0.161	0.588		
	0.028	0.336			0.748	0.091			-0.17	0.500		
	0.129	0.224			0.711	0.211			-0.123	0.460		
	0.248	0.147			0.630	0.340			0.000	0.267		
	0.273	0.081			0.528	0.458			0.036	0.073		
	0.404	0.012			0.383	0.587			0.231	-0.027		
	0.486	-0.034			0.323	0.641			0.217	-0.292		
	0.604	-0.095			0.262	0.714			0.417	-0.457		
	0.699	-0.135			0.050	0.767			0.614	-0.711		

NSGA-II which finds superior on measure *Hypervolume*.

6 Engineering design problems

To evaluate the efficacy of the proposed *MOSOA* algorithm, its efficiency is evaluated on six specific engineering design problems. These multi-objective constrained engineering problems employed the death penalty (Coello, 2002). Though, the role of death penalty is used to discard the infeasible solutions and does not use the knowledge of those solutions that are useful in solving the controlled inviolable regions. *MOSOA* algorithm is fitted with death penalty feature to handle the multiple constraints due to low computational cost and its simplicity.

6.1 Welded beam design problem

This issue has centered primarily on reducing the cost of production and at the same time decreases the vertical deflection (Ragsdell & Phillips, 1976) (see Fig. 15). The

Table 6
Wilcoxon signed-rank test.

Algorithms	*Hypervolume*	Δ_p	*Spread*	*Epsilon*
MOSOA	=	+	+	+
MOPSO	+	+	=	+
NSGA-II	-	+	+	=
MOEA/D	+	+	+	+
PESA-II	+	+	=	+
MOSHEPO	+	+	+	+
MOACO	+	+	+	+

problem of welded beam design requires four variables for optimization, as shown in Fig. 15. Appendix D of *Supplementary Material* sets out the scientific notation of this problem. Table 7 illustrates the comprehensive study of various approaches in order to obtain the best outcome for the problem. The optimal solution for finding the near-optimal solution is accomplished using *MOSOA*. Proposed solution outperforms indicative of longevity of the algorithm for solving restricted engineering problems.

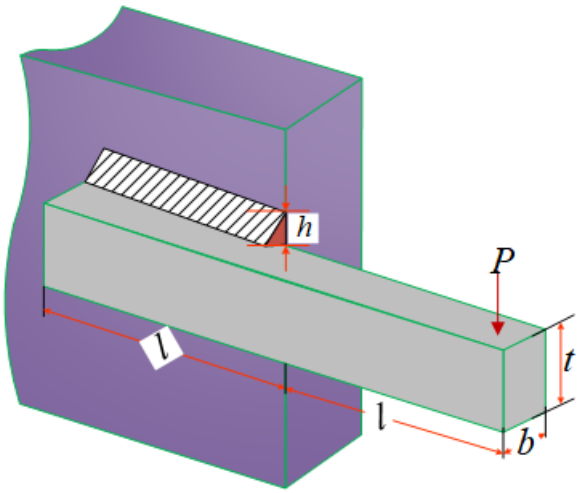

Fig. 15. Welded beam design problem.

Fig. 16 demonstrates that *MOSOA* is having the optimal solution. From this figure it is observed that in case of solving the problem of welded beam design, the proposed *MOSOA* algorithm has a high durability.

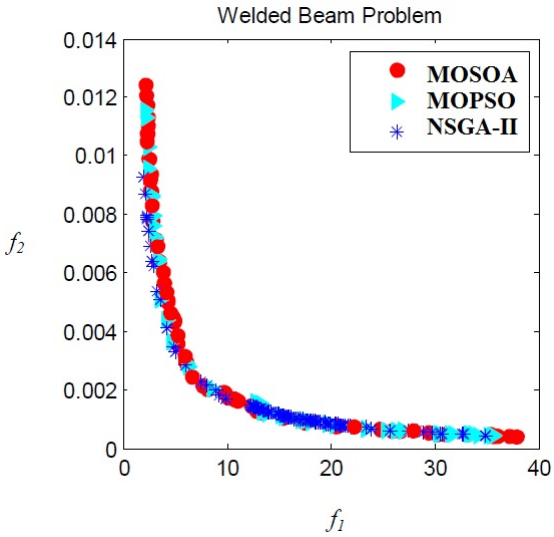

Fig. 16. The obtained Pareto solutions by MOSOA and competitive techniques on welded beam problem.

6.2 Multiple-disk clutch brake design problem

This problem focuses on lowering the stop time (f_1) and lowering the brake system mass (f_2) (Rao & Waghmare, 2017) (see Fig. 17 and Appendix D of *Supplementary Material*). The question consists of five criteria for judgment. Table 8 demonstrates the statistical importance of the strategy suggested against other methods. *MOSOA* is used to obtain the best output of decision variables. If evaluated in terms of Δ_p, *Spread*, *Epsilon*, and *Hypervolume* it outperforms other algorithms. *MOSOA*, *MOPSO*, and *NSGA-II* managed to achieve optimum *Pareto* fronts. It can be observed by Fig. 18 that *MOSOA*, *MOPSO*, and *NSGA-II* results are comparable.

Table 7
The comparison between different approaches for welded beam problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**4.17E+01**	7.08E-01	6.16E-01	6.93E-01	7.81E-01	6.41E-01	7.58E-01
	2.34E-02	5.92E-01	3.21E-01	5.96E-01	7.95E-01	3.86E-01	4.71E-01
Δ_p	**2.70E-03**	2.35E-01	1.68E-01	2.35E-02	5.83E-01	6.91E-02	3.33E-02
	4.04E-05	5.50E-02	1.82E-02	1.31E-02	5.86E-02	4.52E-02	2.49E-02
Spread	**1.36E-02**	1.47E-01	8.91E-01	1.77E+00	2.71E-01	4.47E-01	2.65E-01
	4.95E-07	7.28E-02	1.77E-01	2.11E+00	5.21E-02	1.17E-01	1.96E-01
Epsilon	**8.63E-04**	1.03E-01	8.63E-02	3.21E-02	2.23E-01	2.62E-02	3.67E-02
	1.11E-06	7.52E-02	1.35E-02	8.74E-03	8.62E-02	4.64E-03	2.01E-03

Table 8
The comparison between different approaches for multiple-disk clutch brake problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**3.52E+00**	6.85E-01	5.04E-01	7.52E-01	8.01E-01	8.62E-01	5.41E-01
	1.42E-02	4.31E-01	3.51E-01	6.62E-01	7.13E-01	6.95E-01	2.10E-01
Δ_p	**1.01E-03**	4.14E-02	1.12E-01	6.42E-02	1.20E-01	8.71E-02	4.12E-02
	1.43E-07	4.95E-03	1.51E-02	5.02E-02	7.81E-02	3.12E-02	3.50E-02
Spread	**1.11E-02**	1.35E-01	7.62E-01	2.41E+00	7.42E-01	7.31E-01	2.30E-01
	1.15E-02	1.91E-01	3.12E-01	1.34E+00	4.20E-01	3.41E-01	1.37E-01
Epsilon	**1.64E-03**	1.13E-01	5.02E-02	1.38E-01	1.11E-01	1.92E-02	2.97E-02
	4.16E-04	8.83E-02	1.43E-02	7.31E-02	6.31E-02	1.30E-02	1.57E-02

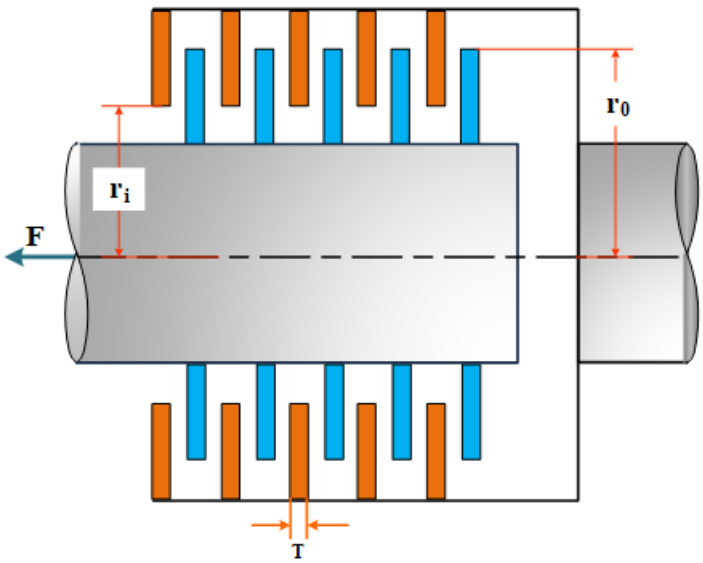

Fig. 17. Multiple-disk clutch brake design problem.

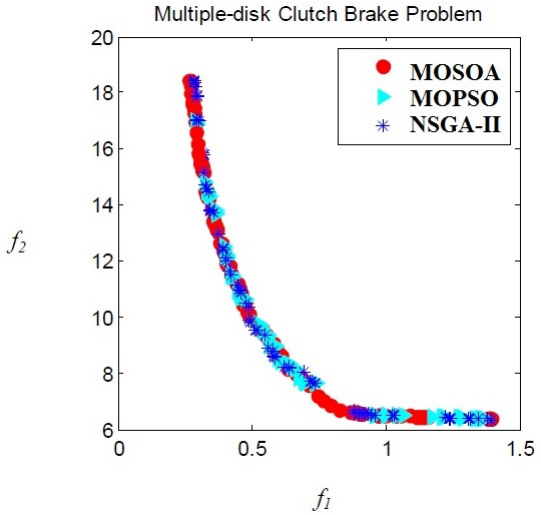

Fig. 18. The obtained Pareto solutions by MOSOA and competitive techniques on multiple-disk clutch brake problem.

6.3 Pressure vessel design problem

This problem was proposed by Kannan and Kramer (Kannan & Kramer, 1994) to minimize total cost (f_1)and optimize storage space (f_2),as shown in Fig. 19.

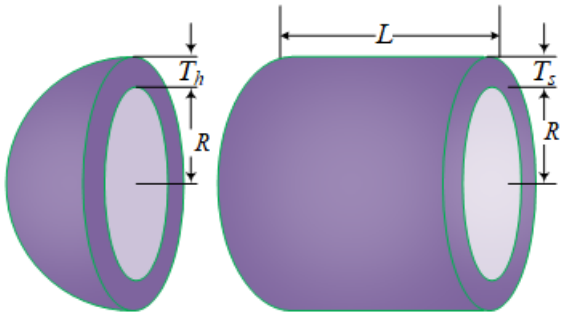

Fig. 19. Pressure vessel design problem.

This problem is composed of four variables of nature. Mathematical notes comprising this particular problem are summarized in Appendix D of *Supplementary Material*. Table 9 displays the *MOSOA's* comparative analysis with other recorded algorithms. Using the *MOSOA* minimum cost and optimum capacity goals are achieved. The *MOSOA*, *MOPSO*, and *NSGA-II* generate optimal *Pareto* fronts which are shown in Fig. 20. The *MOSOA*, when evaluated against the *NSGA-II* and *MOPSO*, is also able to provide comparable results.

Table 9
The comparison between different approaches for pressure vessel problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**2.02E+00**	6.20E-01	5.24E-01	7.94E-01	6.24E-01	6.50E-01	2.10E-01
	1.01E-02	2.10E-01	2.41E-01	5.01E-01	6.00E-01	6.83E-01	1.67E-01
Δ_p	**1.16E-06**	5.15E-02	3.30E-02	1.58E-02	2.10E-03	2.66E-02	1.27E-03
	2.12E-04	1.18E-02	3.44E-03	4.97E-02	2.21E-02	8.91E-02	1.62E-02
Spread	**1.65E-02**	6.00E-01	3.31E-01	2.32E+00	7.44E-01	8.62E-01	3.17E-01
	1.10E-02	1.55E-01	2.61E-01	1.03E+00	5.37E-01	5.34E-01	1.88E-01
Epsilon	**1.18E-03**	2.48E-01	8.51E-02	2.46E-01	1.41E-01	4.29E-02	1.57E-02
	3.20E-04	7.42E-02	6.83E-02	1.34E-01	2.12E-01	2.81E-02	2.17E-02

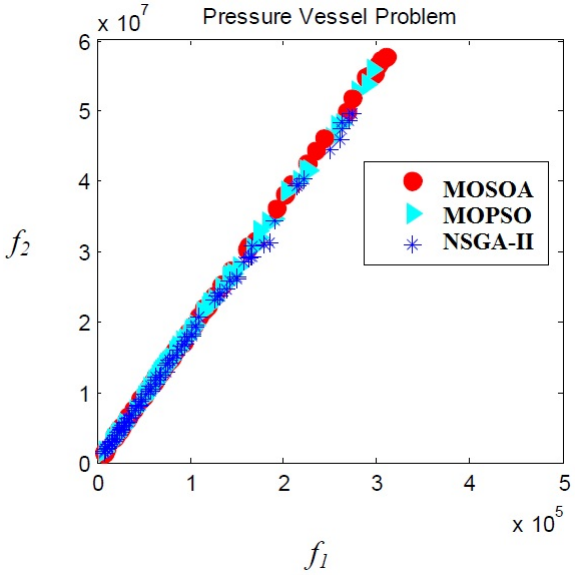

Fig. 20. The obtained Pareto solutions by MOSOA and competitive techniques on pressure vessel problem.

6.4 Speed reducer design problem

This selected problem comprises of two parameters that need optimization, such as: stress (f_2) and weight (f_1). There are eleven constraints (Rao, Savsani, & Vakharia, 2011) to this problem (see Fig. 21). It includes seven variables for optimization of this problem ($x_1 - x_7$). Appendix D of *Supplementary Material* summarizes mathematical notations

that comprise this problem.

Table 10 displays the comparative study of optimal results obtained using six optimization algorithms, and the *MOSOA*. In comparison with other equivalents, output gain can be easily seen from Table 10 using the *MOSOA*. Small weight and tension are the two targets which the *MOSOA* has achieved. Fig. 22 presents the best results obtained from algorithms *MOSOA*, *MOPSO*, and *NSGA-II*. However, for this particular problem the *MOSOA* produces the best optimal *Pareto* solution. *Pareto* solutions collection verifies the *MOSOA's* outstanding results.

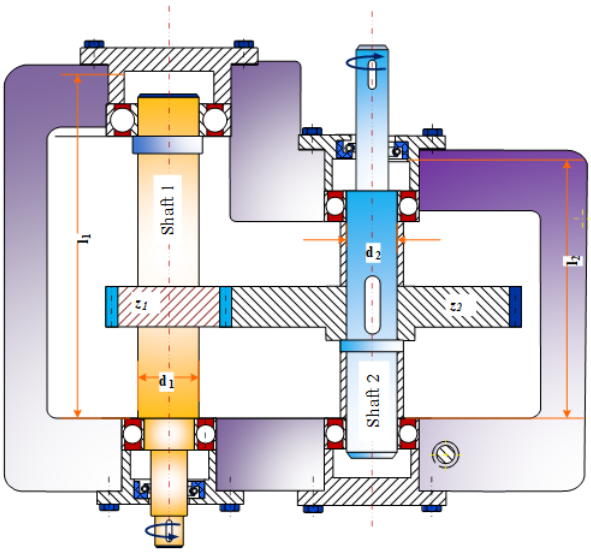

Fig. 21. Speed reducer design problem.

Table 10
The comparison between different approaches for speed reducer problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**2.21E+00**	4.85E-01	5.01E-01	6.51E-01	8.94E-01	7.63E-01	4.10E-01
	1.21E-02	3.34E-01	4.61E-01	5.68E-01	7.02E-01	5.13E-01	3.01E-01
Δ_p	**3.12E-04**	4.06E-02	1.25E-02	5.43E-02	4.01E-01	5.41E-03	4.98E-02
	1.56E-04	3.82E-03	2.02E-02	2.62E-02	7.92E-02	3.65E-03	3.48E-03
Spread	**1.72E-02**	2.35E-01	5.41E-01	1.96E+00	3.51E-01	5.94E-01	2.10E-01
	1.14E-02	2.36E-01	3.31E-01	1.44E+00	5.31E-01	3.43E-01	1.51E-01
Epsilon	**1.17E-03**	1.08E-01	2.27E-01	1.95E-01	5.52E-01	1.33E-01	1.74E-01
	4.08E-04	2.58E-01	6.52E-02	6.62E-02	4.17E-02	1.95E-01	3.53E-02

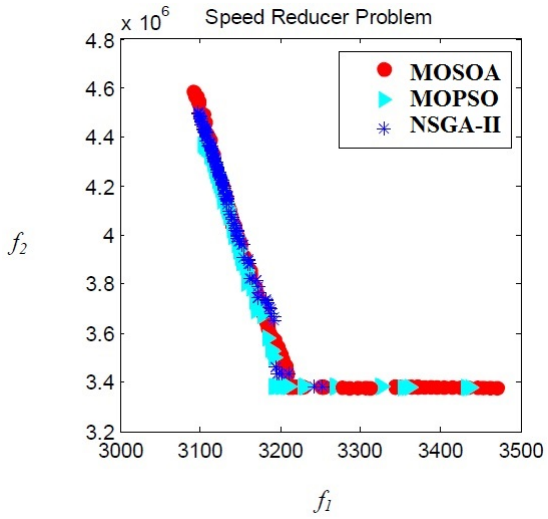

Fig. 22. The obtained Pareto solutions by MOSOA and competitive techniques on speed reducer problem.

6.5 Gear train design problem

In this problem, there are requirements to determine the number of teeth and size for every four gears by which the error (Prayoonrat & Walton, 1988) between obtained and required gear ratio, as shown in Fig. 23. This problem consists of four parameters for an integer judgment. Appendix D of *Supplementary Material* summarizes mathematical notations that comprise this problem.

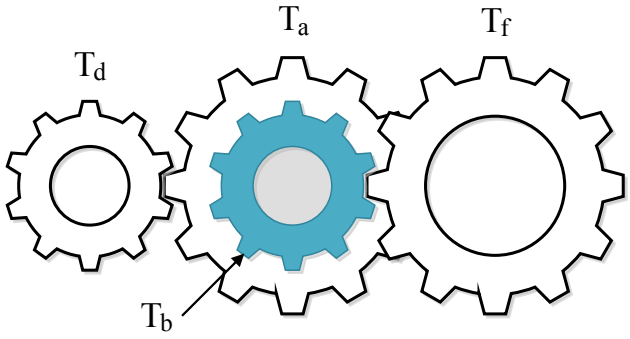

Fig. 23. Gear train design problem.

Table 11 compares the *MOSOA's* proposed performance evaluation with other competing algorithms. Using *MOSOA* the best possible results are obtained. Fig. 24 shows the best *MOSOA*, *MOPSO*, and *NSGA-II* outcomes in terms of the optimal *Pareto* fronts.

Table 11
The comparison between different approaches for gear train problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**2.63E+02**	8.52E-01	5.10E-01	5.38E-01	7.61E-01	8.85E-01	5.47E-01
	1.85E-03	5.95E-01	4.61E-01	6.96E-01	6.13E-01	7.90E-01	3.33E-01
Δ_p	**1.61E-03**	4.41E-01	3.57E-02	5.78E-01	3.44E-01	6.01E-02	3.51E-02
	1.40E-02	1.88E-02	**4.64E-04**	1.32E-02	3.07E-02	3.65E-02	2.31E-02
Spread	**1.25E-02**	1.83E-01	4.38E-01	1.24E+00	6.92E-01	8.63E-01	4.21E-01
	6.62E-03	1.02E-01	2.11E-01	1.01E+00	4.01E-01	4.31E-01	2.46E-01
Epsilon	**2.44E-04**	1.10E-01	1.31E-01	1.21E-01	8.45E-02	1.26E-01	3.61E-02
	1.50E-04	6.61E-02	1.03E-01	8.64E-02	2.47E-02	1.24E-01	1.91E-02

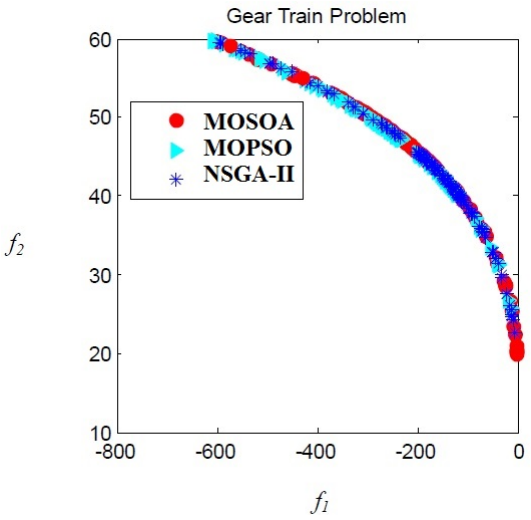

Fig. 24. The obtained Pareto solutions by MOSOA and competitive techniques on gear train problem.

6.6 25-bar truss design problem

To test the output of the proposed algorithm this problem (Dhiman & Kaur, 2018, In pressa, 2018, In pressb) is selected. This architecture consists of ten static nodes, and twenty-five bar cross-sectional members (see Fig. 26).

The 25-bar truss dominant position is shown in Table 12. Compared to other algorithms, as is shown in Table 12, the *MOSOA* obtains the best optimal consequence. The statistical findings obtained in terms of average and standard deviation also indicate that the *MOSOA* is outperforming the rival algorithms. From Fig. 25, it is also clear that the *MOSOA* converges very quickly in order to achieve the optimum solution.

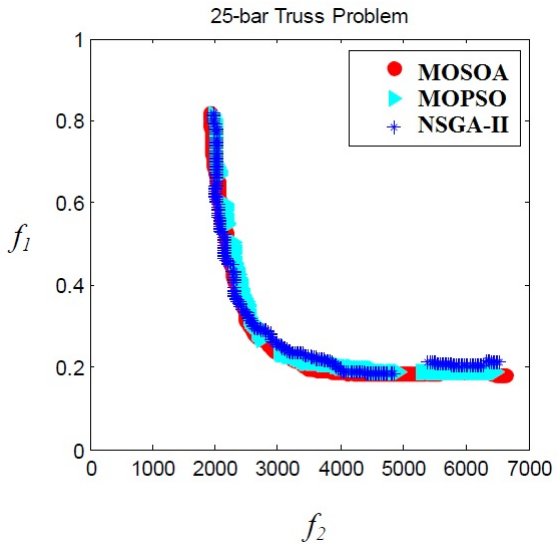

Fig. 25. The obtained Pareto solutions by MOSOA and competitive techniques on 25-bar truss problem.

7 Conclusions and Future Works

This article introduced a new *MOO* algorithm, named Multi-objective Seagull Optimization Algorithm (*MOSOA*). This algorithm mimics the searching and attacking behaviors of seagulls in nature. Three new components are integrated into this study to solve *MOO* problems. The first part includes an archive, the key function of which is to identify and compile the best non-dominated solutions. The second part provides the grid mechanism to omit the most crowded part of the non-dominated solutions. And, the third aspect provides a criterion for leader selection to choose optimal solutions from the archive. This algorithm is checked on twenty-four well-known benchmark test functions. Finally, six real-life *MOO* problems have validated it. Various experimental findings in this

Table 12
The comparison between different approaches for 25-bar truss problem.

Performance Metrics	MOSOA	MOPSO	NSGA-II	MOEA/D	PESA-II	MOSHEPO	MOACO
Hypervolume	**1.13E-00**	7.58E-01	3.34E-01	6.45E-01	3.04E-01	5.75E-01	2.03E-01
	1.15E-02	4.78E-01	1.41E-01	4.01E-01	1.70E-01	4.84E-01	1.30E-01
Δ_p	**2.21E-03**	1.55E-01	2.78E-02	1.01E-01	1.67E-01	2.26E-02	4.37E-02
	1.41E-03	1.71E-01	1.46E-02	1.33E-01	3.01E-02	1.85E-02	2.58E-02
Spread	**1.16E-02**	2.58E-01	3.31E-01	1.48E-01	4.06E-01	4.37E-01	2.33E-01
	3.32E-03	2.42E-01	2.26E-01	1.17E-01	3.03E-01	1.85E-01	1.01E-01
Epsilon	**1.87E-03**	1.90E-01	1.81E-01	1.58E-01	2.86E-02	1.41E-01	2.65E-01
	1.71E-03	2.01E-02	1.43E-01	1.80E-02	1.75E-02	1.01E-01	1.57E-01

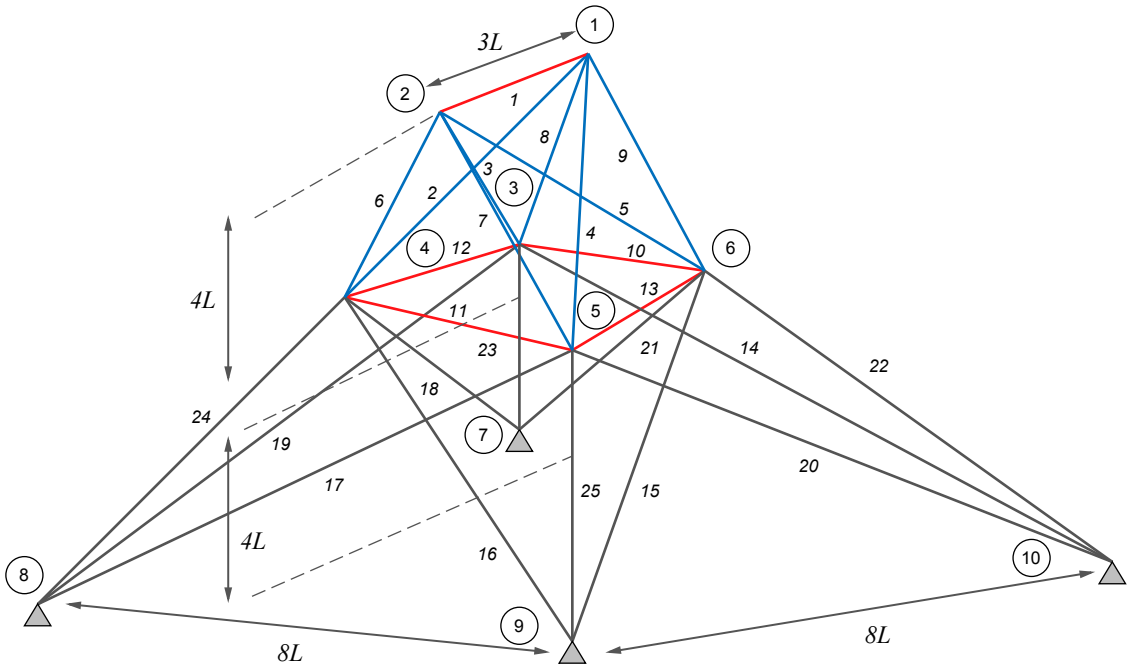

Fig. 26. 25-bar truss design problem.

regard represent that the *MOSOA* provides the best results in terms of computational costs compared with existing competing algorithms. The binary version of *MOSOA* algorithm will be designed for future research to solve various difficult real-life complex problems. Also, the many-objective version of the proposed algorithm can be seen as the future contribution.

*Acknowledgement The first and corresponding author **Dr. Gaurav Dhiman** would like to thanks to his parents and god for their divine blessings on him.

8 References

Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. (2015, 05). A comprehensive review of swarm optimization algorithms. *PLOS ONE*, *10*(5), 1-36. Retrieved from `https://doi.org/10.1371/journal.pone.0122827` doi: 10.1371/journal.pone.0122827

Angus, D., & Woodward, C. (2009). Multiple objective ant colony optimisation. *Swarm Intelligence*, *3*(1), 69–85. Retrieved from `http://dx.doi.org/10.1007/s11721-008-0022-4` doi: 10.1007/s11721-008-0022-4

Baykasoğlu, A., & Akpinar, Ş. (2015). Weighted superposition attraction (wsa): A swarm intelligence algorithm for optimization problems–part 2: Constrained optimization. *Applied Soft Computing*, *37*, 396–415.

Baykasoğlu, A., & Akpinar, Ş. (2017). Weighted superposition attraction (wsa): A swarm intelligence algorithm for optimization problems–part 1: Unconstrained optimization. *Applied Soft Computing*, *56*, 520–540.

Branke, J., Deb, K., Dierolf, H., & Osswald, M. (2004). Finding knees in multi-objective optimization. In X. Yao et al. (Eds.), *Parallel problem solving from nature - ppsn viii: 8th international conference, birmingham, uk, september 18-22, 2004. proceed-*

ings (pp. 722–731). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://dx.doi.org/10.1007/978-3-540-30217-9$_7$3 doi: 10.1007/978-3-540-30217-9$_7$3

Cai, X., Li, Y., Fan, Z., & Zhang, Q. (2014). An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, *19*(4), 508–523.

Chegini, S. N., Bagheri, A., & Najafi, F. (2018). Psoscalf: A new hybrid pso based on sine cosine algorithm and levy flight for solving optimization problems. *Applied Soft Computing*, *73*, 697–726.

Chen, M., & Hammami, O. (2015). A system engineering conception of multi-objective optimization for multi-physics system. In M. Haddar et al. (Eds.), *Multiphysics modelling and simulation for systems design and monitoring: Proceedings of the multiphysics modelling and simulation for systems design conference, mmssd 2014, 17-19 december, sousse, tunisia* (pp. 299–306). Cham: Springer International Publishing. Retrieved from http://dx.doi.org/10.1007/978-3-319-14532-7$_3$1 doi: 10.1007/978-3-319-14532-7$_3$1

Cheraghalipour, A., Hajiaghaei-Keshteli, M., & Paydar, M. M. (2018). Tree growth algorithm (tga): A novel approach for solving optimization problems. *Engineering Applications of Artificial Intelligence*, *72*, 393–414.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, *191*(11-12), 1245 - 1287.

Coello, C. A. C. (2006, Feb). Evolutionary multi-objective optimization: a historical view of the field. *IEEE Computational Intelligence Magazine*, *1*(1), 28-36. doi: 10.1109/MCI.2006.1597059

Coello Coello, C. A. (2009, Mar 01). Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China*, *3*(1), 18–30. Retrieved from https://doi.org/10.1007/s11704-009-0005-7 doi: 10.1007/s11704-009-0005-7

Coello Coello, C. A., Dhaenens, C., & Jourdan, L. (2010). Multi-objective combinatorial optimization: Problematic and context. In C. A. Coello Coello, C. Dhaenens, & L. Jourdan (Eds.), *Advances in multi-objective nature inspired computing* (pp. 1–21). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-11218-8$_1$ doi: 10.1007/978-3-642-11218-8$_1$

Coello Coello, C. A., & Lechuga, M. S. (2002). Mopso: A proposal for multiple objective particle swarm optimization. In *Proceedings of the evolutionary computation on 2002. cec '02. proceedings of the 2002 congress - volume 02* (pp. 1051–1056). Washington, DC, USA: IEEE Computer Society. Retrieved from http://dl.acm.org/citation.cfm?id=1251972.1252327

Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. (2001). Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the 3rd annual conference on genetic and evolutionary computation* (pp. 283–290).

Corne, D. W., Jerram, N. R., Knowles, J. D., Oates, M. J., & J, M. (2001). Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the genetic and evolutionary computation conference (gecco'2001* (pp. 283–290). Morgan Kaufmann Publishers.

Deb, K. (2012). Advances in evolutionary multi-objective optimization. In G. Fraser & J. Teixeira de Souza (Eds.), *Search based software engineering: 4th international symposium, ssbse 2012, riva del garda, italy, september 28-30, 2012. proceedings* (pp. 1–26). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://dx.doi.org/10.1007/978-3-642-33119-0$_1$ doi: 10.1007/978-3-642-33119-0$_1$

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002, Apr). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197. doi: 10.1109/4235.996017

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2005). Scalable test problems for evolutionary multiobjective optimization. In A. Abraham, L. Jain, & R. Goldberg (Eds.), *Evolution-*

ary multiobjective optimization: Theoretical advances and applications (pp. 105–145). London: Springer London. Retrieved from http://dx.doi.org/10.1007/1-84628-137-7$_6$ doi: 10.1007/1-84628-137-7$_6$

Dhiman, G. (2020). Moshepo: a hybrid multi-objective approach to solve economic load dispatch and micro grid problems. *Applied Intelligence*, *50*(1), 119–137.

Dhiman, G., & Kaur, A. (2018, In pressa). A hybrid algorithm based on particle swarm and spotted hyena optimizer for global optimization. In *Advances in intelligent systems and computing.*

Dhiman, G., & Kaur, A. (2018, In pressb). Spotted hyena optimizer for solving engineering design problems. In *International conference on machine learning and data science.* IEEE.

Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, *114*, 48 - 70. Retrieved from http://www.sciencedirect.com/science/article/pii/S0965997816305567 doi: https://doi.org/10.1016/j.advengsoft.2017.05.014

Dhiman, G., & Kumar, V. (2018a). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowledge-Based Systems*, *159*, 20–50.

Dhiman, G., & Kumar, V. (2018b). Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. *Knowledge-Based Systems*, *150*, 175–197.

Dhiman, G., & Kumar, V. (2019a). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, *165*, 169–196.

Dhiman, G., & Kumar, V. (2019b). Spotted hyena optimizer for solving complex and non-linear constrained engineering problems. In *Harmony search and nature inspired optimization algorithms* (pp. 857–867). Springer.

Gong, M., Jiao, L., Du, H., & Bo, L. (2008, June). Multiobjective immune algorithm with nondominated neighbor-based selection. *Evol. Comput.*, *16*(2), 225–255. Retrieved from http://dx.doi.org/10.1162/evco.2008.16.2.225 doi: 10.1162/evco.2008.16.2.225

Hajiaghaei-Keshteli, M., & Aminnayeri, M. (2013). Keshtel algorithm (ka); a new optimization algorithm inspired by keshtels' feeding. In *Proceeding in ieee conference on industrial engineering and management systems* (pp. 2249–2253).

Hancer, E., Xue, B., Zhang, M., Karaboga, D., & Akay, B. (2015, May). A multi-objective artificial bee colony approach to feature selection using fuzzy mutual information. In *2015 ieee congress on evolutionary computation (cec)* (p. 2420-2427). doi: 10.1109/CEC.2015.7257185

Handl, J., Kell, D. B., & Knowles, J. (2007, April). Multiobjective optimization in bioinformatics and computational biology. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, *4*(2), 279–292. Retrieved from http://dx.doi.org/10.1109/TCBB.2007.070203 doi: 10.1109/TCBB.2007.070203

Hoyo, J., Elliott, A., & Sargatal, J. (1996). Handbook of the birds of the world. *Lynx Edicions*, *3*, 572-599.

Kannan, B., & Kramer, S. N. (1994). An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of mechanical design*, *116*(2), 405–411.

Kaur, A., & Dhiman, G. (2019). A review on search-based tools and techniques to identify bad code smells in object-oriented systems. In *Harmony search and nature inspired optimization algorithms* (pp. 909–921). Springer.

Kipouros, T., Jaeggi, D. M., Dawes, W. N., Parks, G. T., Savill, A. M., & Clarkson, P. J. (2008, Mar 01). Biobjective design optimization for axial compressors using tabu search. *AIAA Journal*, *46*(3), 701-711. Retrieved from http://dx.doi.org/10.2514/1.32794 doi: 10.2514/1.32794

Knowles, J. D., & Corne, D. W. (2000, June). Approximating the nondominated front using the pareto archived evolution strategy. *Evol. Comput.*, *8*(2), 149–172. Retrieved from http://dx.doi.org/10.1162/106365600568167 doi: 10.1162/106365600568167

Li, M., & Zheng, J. (2009). Spread assessment for evolutionary multi-objective optimization. In M. Ehrgott, C. M. Fonseca, X. Gandibleux, J.-K. Hao, & M. Sevaux (Eds.), *Evolutionary multi-criterion optimization: 5th international conference, emo 2009, nantes, france, april 7-10, 2009. proceedings* (pp. 216–230). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://dx.doi.org/10.1007/978-3-642-01020-$0_2$0 doi: 10.1007/978-3-642-01020-$0_2$0

Liu, M., Zou, X., Chen, Y., & Wu, Z. (2009). Performance assessment of dmoea-dd with cec 2009 moea competition test instances. In *2009 ieee congress on evolutionary computation* (pp. 2913–2918).

Luh, G.-C., & Chueh, C.-H. (2004). Multi-objective optimal design of truss structure with immune algorithm. *Computers and Structures*, *82*(11-12), 829 - 844. Retrieved from http://www.sciencedirect.com/science/article/pii/S0045794904000677 doi: http://dx.doi.org/10.1016/j.compstruc.2004.03.003

Macdonald, S. M., & Mason, C. (1973). Predation of migrant birds by gulls. *Brit. Birds*, *66*, 361-363.

Madavan, N. K. (2002). Multiobjective optimization using a pareto differential evolution approach. In *Evolutionary computation, 2002. cec '02. proceedings of the 2002 congress on* (Vol. 2, p. 1145-1150). doi: 10.1109/CEC.2002.1004404

Marler, R., & Arora, J. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, *26*(6), 369–395. Retrieved from http://dx.doi.org/10.1007/s00158-003-0368-6 doi: 10.1007/s00158-003-0368-6

Marler, R. T., & Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, *26*(6), 369–395.

Pradhan, P. M., & Panda, G. (2012). Solving multiobjective problems using cat swarm optimization. *Expert Systems with Applications*, *39*(3), 2956 - 2964. Retrieved from http://www.sciencedirect.com/science/article/pii/S0957417411012887 doi: http://dx.doi.org/10.1016/j.eswa.2011.08.157

Prayoonrat, S., & Walton, D. (1988). Practical approach to optimum gear train design. *Computer-Aided Design*, *20*(2), 83–92.

Ragsdell, K., & Phillips, D. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Engineering for Industry*, *98*(3), 1021–1025.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2011). Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303–315.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2012). Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information sciences*, *183*(1), 1–15.

Rao, R. V., & Waghmare, G. (2017). A new optimization algorithm for solving complex constrained design optimization problems. *Engineering Optimization*, *49*(1), 60–83.

Richardson, A. (2010). Nonparametric statistics for non-statisticians: A step-by-step approach by gregory w. corder, dale i. foreman. *International Statistical Review*, *78*(3), 451–452.

Roy, P. C., Islam, M. M., Murase, K., & Yao, X. (2015). Evolutionary path control strategy for solving many-objective optimization problem. *IEEE transactions on cybernetics*, *45*(4), 702–715.

Rudolph, G., Schütze, O., Grimme, C., Domínguez-Medina, C., & Trautmann, H. (2016, Jun 01). Optimal averaged hausdorff archives for bi-objective problems: theoretical and numerical results. *Computational Optimization and Applications*, *64*(2), 589–618. Retrieved from https://doi.org/10.1007/s10589-015-9815-8 doi: 10.1007/s10589-015-9815-8

Schütze, O., Esquivel, X., Lara, A., & Coello, C. A. C. (2012, Aug). Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, *16*(4), 504-522. doi: 10.1109/TEVC.2011.2161872

Schütze, O., Laumanns, M., Tantar, E., Coello, C. A. C., & Talbi, E. G. (2010, March). Computing gap free pareto front approximations with stochastic search algorithms. *Evolutionary Computation*, *18*(1), 65-96. doi: 10.1162/evco.2010.18.1.18103

Wolpert, D. H., & Macready, W. G. (1997, April). No free lunch theorems for optimization. *Trans. Evol. Comp*, *1*(1), 67–82. Retrieved from `http://dx.doi.org/10.1109/4235.585893` doi: 10.1109/4235.585893

Wu, B., Qian, C., Ni, W., & Fan, S. (2012a). Hybrid harmony search and artificial bee colony algorithm for global optimization problems. *Computers & Mathematics with Applications*, *64*(8), 2621–2634.

Wu, B., Qian, C., Ni, W., & Fan, S. (2012b). The improvement of glowworm swarm optimization for continuous optimization problems. *Expert Systems with Applications*, *39*(7), 6335–6342.

Xue, Y., Jiang, J., Zhao, B., & Ma, T. (2018). A self-adaptive artificial bee colony algorithm based on global best for global optimization. *Soft Computing*, *22*(9), 2935–2952.

Yang, X.-S. (2010). *Engineering optimization: an introduction with metaheuristic applications*. John Wiley & Sons.

Yang, X.-S., Karamanoglu, M., & He, X. (2014). Flower pollination algorithm: A novel approach for multiobjective optimization. *Engineering Optimization*, *46*(9), 1222-1237.

Zhang, Q., & Li, H. (2007, Dec). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712-731. doi: 10.1109/TEVC.2007.892759

Zhang, Q., Zhou, A., Zhao, S., Suganthan, P. N., Liu, W., & Tiwari, S. (2008). Multiobjective optimization test instances for the cec 2009 special session and competition. *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, *264*.

Zhang, Y., Gong, D.-w., & Ding, Z.-h. (2011). Handling multi-objective optimization problems with a multi-swarm cooperative particle swarm optimizer. *Expert Systems with Applications*, *38*(11), 13933–13941.

Zitzler, E., Deb, K., & Thiele, L. (2000, June). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.*, *8*(2), 173–195. Retrieved from `http://dx.doi.org/10.1162/106365600568202` doi: 10.1162/106365600568202

Zitzler, E., & Thiele, L. (1999, Nov). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257-271. doi: 10.1109/4235.797969

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & da Fonseca, V. G. (2003, April). Performance assessment of multiobjective optimizers: An analysis and review. *Trans. Evol. Comp*, *7*(2), 117–132. Retrieved from `http://dx.doi.org/10.1109/TEVC.2003.810758` doi: 10.1109/TEVC.2003.810758