Neural Computing and Applications
https://doi.org/10.1007/s00521-023-08432-0

ORIGINAL ARTICLE

Multi-objective chaos game optimization

Nima Khodadadi[1] · Laith Abualigah[2,7,8,9] · Qasem Al-Tashi[3,4] · Seyedali Mirjalili[5,6,10]

Received: 8 January 2022 / Accepted: 20 February 2023

Abstract
The Chaos Game Optimization (CGO) has only recently gained popularity, but its effective searching capabilities have a lot of potential for addressing single-objective optimization issues. Despite its advantages, this method can only tackle problems formulated with one objective. The multi-objective CGO proposed in this study is utilized to handle the problems with several objectives (MOCGO). In MOCGO, Pareto-optimal solutions are stored in a fixed-sized external archive. In addition, the leader selection functionality needed to carry out multi-objective optimization has been included in CGO. The technique is also applied to eight real-world engineering design challenges with multiple objectives. The MOCGO algorithm uses several mathematical models in chaos theory and fractals inherited from CGO. This algorithm's performance is evaluated using seventeen case studies, such as CEC-09, ZDT, and DTLZ. Six well-known multi-objective algorithms are compared with MOCGO using four different performance metrics. The results demonstrate that the suggested method is better than existing ones. These Pareto-optimal solutions show excellent convergence and coverage.

Keywords Algorithm · Bechmark · Artificial Intelligence · Multi-objective optimization · CEC benchmark · Chaos game optimization · Engineering problems · Optimization

1 Introduction

The term "optimization" is commonly used to refer to the process of determining which of several possible actions would yield the best results under specified constraints. Because of the interdependence and complexity of sophisticated engineering systems, one will need an analyst with a broad perspective to help one optimize their production, laboratory, retail, or service system. Furthermore, when studying a system, the subsets' interaction should be considered to preserve its integrity and optimality. Additionally, the system's components' specifications, and existing uncertainties, should be described and incorporated into the system's intended goals. Metaheuristic algorithms are search techniques that use a higher-level approach to find the optimal solution to a given problem. Genetic Algorithm (GA) [1], Particle Swarm Optimizer (PSO) [2], Ant colony Optimization (ACO) [3], Stochastic Paint Optimizer (SPO) [4] and Mountain Gazelle Optimizer (MGO) [5] are some of the well-known metaheuristic algorithms. Additionally, optimization is applied in a number of fields, such as control, medicine, image processing and structural engineering [6, 7].

✉ Seyedali Mirjalili
ali.mirjalili@torrens.edu.au

1 The Department of Civil, Architectural and Environmental Engineering, University of Miami, 1251 Memorial Drive, Coral Gables, FL 33146, USA

2 MEU Research Unit, Middle East University, Amman 11831, Jordan

3 Department of Imaging Physics, The University of Texas MD Anderson Cancer Center, Houston, TX, USA

4 University of Albaydha, Albaydha, Yemen

5 Centre for Artificial Intelligence Research and Optimization, Torrens University Australia, Sydney, Australia

6 Yonsei Frontier Lab, Yonsei University, Seoul, Korea

7 Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan

8 Computer Science Department, Prince Hussein Bin Abdullah Faculty for Information Technology, Al al-Bayt University, Mafraq 25113, Jordan

9 Applied Science Research Center, Applied Science Private University, Amman 11931, Jordan

10 University Research and Innovation Center, Obuda University, Budapest 1034, Hungary

Published online: 02 April 2023

Everyone desires to gain the most significant benefit at the cheapest cost [8]. This goal can be presented mathematically as an optimization problem. However, there are various optimization problems with many objectives in the real world and frequent inconsistencies among specific goals [9]. Therefore, it is often challenging to discover the optimal solution that settles all the objects simultaneously [10]. Accordingly, multi-objective problems frequently have multiple solutions rather than a single one, and multi-objective optimizers have gained the interest of researchers. Ordinarily, optimization problems with less than four specified objectives are designated multi-objective problems, while other problems with more than four are designated many-objective problems [11].

After a lengthy investigation, multi-objective problems are sufficiently advanced, and exciting consideration is given to addressing many-objective problems [12]. Generally speaking, techniques for tackling various optimization problems are subdivided into two kinds. The conventional optimizers are gradient search optimizers, Newton search optimizers, quasi-newton search optimizers, and conjugate gradient search optimizers. The other kind is heuristic search optimizers, which are stimulated the person's expertise in addressing remarkable problems or behavior of living in real life. Classical optimizers typically require calculating derivatives or differentials, so it is hard to utilize many complex real-world problems. Therefore, usually, when tackling multi-objective problems, heuristic optimizers are employed, such as Multi-Objective Genetic Algorithm (MOGA) [13], Multi-Objective Artificial Bee Colony Optimizer (MOABC) [14], Multi-Objective Artificial Hummingbird Algorithm (MOAHA) [15], Multi-Objective Seagull Optimization Algorithm (MOSOA) [16], Multi-Objective Particle Swarm Optimization (MOPSO) [17], Multi-Objective Firefly Algorithm (MOFA) [18], Multi-Objective Atomic Orbital Search (MOAOS) [19], Artificial Vultures Optimization Algorithm (MOAVOA) [20], Multi-Objective Bonobo Optimizer (MOBO) [21], Multi-Objective Stochastic Paint Optimizer (MOSPO) [22], Multi-Objective Moth-Flame Optimization (MMFO) [23], Archive-Based Multi-objective Harmony Search (AMHS) [24] and Multi-objective Non-dominated Advanced Butterfly Optimization Algorithm (MONSBOA) [25]. This paper proposed a novel optimization structure with a distinguished convergence and coverage as a new multi-objective optimizer. The proposed method is based on modifying the Chaos Game optimizer (CGO) [26] to produce dynamic control factors to decrease the time of finding the best solutions for addressing various multi-objective benchmark functions and industrial engineering problems.

Nevertheless, the number of non-dominated solutions is negligible at the beginning of the optimization rule. Therefore, they may use the population members in the wrong direction. Hence, the main idea is to generate a diverse number of solutions in the Pareto front that will encourage the candidate solutions to progress toward encouraging areas of the given search space in successive iterations. The multi-objective CGO approach that has been presented, referred to as MOCGO, makes use of a leader selection methodology to strengthen its capabilities and avoid the drawbacks of the original CGO method as well as an archive method to save non-dominated solutions. The proposed MOCGO is tested on a wide variety of problems, both constrained and unconstrained, from the fields of mathematics and industrial engineering optimization. The results of a series of comparisons between the proposed MOCGO method and other state-of-the-art multi-objective approaches using several common performance metrics, such as Inverted Generational Distance (IGD), Generational Distance (GD), Spacing(S), and Maximum Spread (MS), demonstrated the proposed MOCGO approach's superior ability to handle multiple complex problems.

This article continues as follows. Section 2 covers multi-objective related work. Section 3 suggests a single version and a multi-objective Chaos Game Optimizer (MOCGO). Section 4 tabulates and discusses the experimental outcomes. Section 5 then discusses the conclusion and future works.

2 Literature review

Most real-world optimization problems, including big data, data mining, design, optimization, scheduling, mathematics, control, etc., are essentially designated by multiple differing objectives. The variables are constantly indistinct when tackling specific problems because of uncontrollable circumstances, leading to more complex problem presentations [27]. Single-objective problems are distinct from multi-objective problems [28]. Only one best solution is achieved in the first type, whereas many solutions are accomplished in multi-objective problems, called Pareto-optimal solutions [29]. The objective function in single-objective problems is numerical, and it is sufficient to check the objective values to compare the quality of the candidate solutions. Typically, the best cases of minimization problems are the smaller objective values. But the objective values are a vector in multi-objective problems. Therefore, the theory of Pareto dominance is used to compare the quality of the candidate solutions with various objective values [30].

As an example, in [31], a multi-objective GA is proposed for optimizing the parameters of the Modular Neural Network, and this is only one of a number of new multi-objective techniques that have recently been introduced in

the literature. The advantages of the proposed multi-objective strategy are illustrated using face and ear datasets. Results from the granular strategy-using modular neural network were shown to be more trustworthy than those from the traditional method that did not involve optimization. A new optimization structure is expressed in [32] by connecting multi-objective and multicriteria decision-making ideas. The proposed optimization method combined multi-objective ABC, best–worst, and grey relational methods to address the optimization problem. The outcomes demonstrated the efficacy of the proposed approach for resolving problems with multiple objectives.

A new multi-objective hybrid forecasting method is proposed in [33] using Ant Lion optimizer, which includes four steps: data preprocessing, optimization, forecasting, and evaluation steps. The decomposing approach distributes the initial wind speed data into a finite collection of segments. The outcomes demonstrated that the suggested methodology produced lower average mean absolute errors. For the purpose of resolving multi-objective problems in rapidly changing environments, an innovative multi-objective evolutionary PSO has been developed in [34]. Furthermore, a new optimization structure of multi-swarm-based PSO is utilized to tackle the given issues in changing settings. The results showed that the proposed method got better outcomes for trading with these multi-objective problems in quickly changing settings.

In [35], it is suggested that a modified version of multi-objective FA, which consists of six single and multi-objective optimization problems, may be applied to big data situations. As seen in the findings, the proposed strategy outperformed the competitors. This paper introduced a multi-objective optimizer for addressing the flow shop scheduling problems considering the energy losses. The proposed optimizer is compared with other well-known optimizers by analyzing the results. A novel framework is introduced in [36] as a multi-objective evolutionary method. Several multi-objective methods are used in the proposed framework, which is used to address various problems. The proposed methodologies had good results, indicating that the design is feasible and practicable. New multi-objective feasibility PSO is presented in [37] to address constrained multi-objective problems. A comparison of the suggested method to the original multi-objective PSO and other popular methods revealed significant improvements for the latter.

Khodadadi et al. [38] have created a multi-objective version of the Crystal Structure Algorithm (Crystal), which draws its inspiration from crystal structure principles. Completions on Evolutionary Computation (CEC-09), real-world engineering, and mathematics multi-objective optimization benchmark problems are used to evaluate the effectiveness of the given method. If applied to multi-objective issues, the strategies presented can deliver outstanding results.

Pereira et al. [39] described the invention of the Multi-objective Lichtenberg Algorithm, a new metaheuristic inspired by the propagation of radial intra-cloud lightning and Lichtenberg figures that can handle multiple objectives. For each iteration, the algorithm uses a Lichtenberg figure to distribute points for evaluation in the objective function, which is shot in various sizes with varied rotations. This allows for a great deal of exploration and exploitation. As the first hybrid multi-objective metaheuristic, the Multi-objective Lichtenberg Algorithm (MOLA) has been tested against classic and current metaheuristics employing well-known and complicated test function groups as well as constrained complex engineering challenges. With expressive values of convergence and maximum spread, the Multi-Objective Lichtenberg Algorithm stands out as a potential multi-objective algorithm.

Zhong [40] suggested the multi-objective marine predator algorithm (MOMPA). This approach incorporates an external archive component storing previously discovered non-dominant Pareto-optimal solutions. The concept of elite selection serves as the foundation for a technique that is being developed for selecting top predators. Using the predator's foraging strategy as a model, this method selects the most powerful solutions from the repository to serve as top predators. Algorithm performance is evaluated using the CEC2019 multi-modal multi-objective benchmark functions and compared to nine current metaheuristics techniques. In addition, the proposed approach is tested using seven multi-objective engineering design problems. The findings show that the suggested MOMPA algorithm outperforms previous algorithms and gives very competitive outcomes.

Multi-objective thermal exchange optimization (MOTEO) is a physics-inspired metaheuristic approach suggested by Khodadadi et al. [40] to address problems of multi-objective optimization. The single version of TEO has used Newtonian cooling laws to solve single-objective optimization problems more effectively, and MOTEO is based on that principle. Different problems are used to assess MOTEO's efficacy in this research. In comparison with existing algorithms, the recommended method may provide accurate solution, consistency, and coverage for addressing multi-objective problems, resulting in high-quality Pareto Fronts.

Dhiman et al. [16] introduce the Multi-objective Seagull Optimization Algorithm (MOSOA). The non-dominated Pareto-optimal solutions are supposed to be able to be cached with the help of the dynamic archive, according to this method. By driving seagull migration and attacking behaviors, the roulette wheel selection approach is utilized in order to select the archive solutions that have the

greatest potential for success. In order to validate the suggested algorithm, it is subjected to validation with twenty-four benchmark test functions, and the performance of the proposed algorithm is evaluated alongside that of previously developed metaheuristic algorithms. In order to determine whether or not the proposed method is suitable for use in the process of finding solutions to problems that occur in the real world, it is tested on six constrained engineering design problems. Empirical analyses demonstrate the suggested method outperforms others. The suggested approach also considers those Pareto-optimal solutions with a high convergence rate.

This research is motivated to develop the multi-objective version of CGO limitations for the first time in the literature. In addition, several analyses have been carried out on the uses of MOO in various fields of study. A survey of some of the MOO settlement methods reveals that they employ a complicated mathematical problem and a complex method of solving. The fundamental contribution of this study is to suggest a MOO settlement approach that does not involve the use of sophisticated mathematical calculations to solve the problem. As the majority of extant optimizers are population-based, they can simultaneously handle a large number of candidate solutions, whereas other search methods employ the same procedure to iteratively duplicate their solutions. Recent novel optimizers have distinct optimization procedures to address different problems with various objectives. However, the well-known optimization theorem, No Free-Lunch (NFL) [41], reasonably explained that none of the existing search methods are approved to tackle all problems efficiently. This statement is true for both single- and multi-objective optimization approaches. As a result, it can be concluded that important problems can be solved by modifying existing, well-known techniques. Different methods are better adapted to tackle unconstrained issues than other constraints, which require careful operators or components.

CGO utilizes a multi-objective particle swarm optimization technique called an archive method in addition to a leader selection rule. Each of these methods is used to find the best solution. Heuristic algorithms can be used in various ways to discover and store Pareto's optimum solutions. In this work, Pareto-optimal solutions are stored in an archive. Evidently, the MOCGO algorithm's convergence originates from the CGO method. CGO can enhance the quality of a solution chosen from the repository.

Nevertheless, it is difficult to identify a set of Pareto-optimal solutions with an extensive range of variations. Chaos Game Optimization (CGO) [26] is a novel search algorithm that handles various optimization challenges. The CGO optimizer's concept is based on chaos theory.

3 Multi-objective chaos game optimization (MOCGO)

The CGO, with its inspiration and the mathematical model of the optimization technique, is described in the next part. Then, the multi-objective nature of this method is described and its features.

3.1 Chaos game optimization (CGO)

Talatahari and Azizi [26] devised the CGO, a population-based metaheuristic algorithm that replicates chaos theory's self-similar and self-organized dynamical systems. The majority of chaotic processes exhibit fractal graphical forms. The chaotic game generates fractals by starting with a polygon form and a randomly chosen beginning point. The goal is to build a series of points repeatedly in order to create a picture with a comparable form at various scales. The number of vertices dictates the primary form of the polygon. A Sierpinski triangle is formed by combining three vertices (see Fig. 1). As can be seen in Fig. 1, a triangle is repeatedly split into sub-triangles.

The CGO method takes into account various solution candidates that reflect certainly suitable seeds within a Sierpinski triangle. The beginning positions of eligible seeds in the search space are picked at random. Each iteration of the algorithm generates four new seeds (X_{new}) that are eligible for the following iteration based on the location of each seed. The new seeds are constructed utilizing three vertices in the search space: X_i, X_{Mean}, and X_{best}. X_i represents the location of the ith suitable seed,X_{Mean} represents the mean of a randomly selected collection of suitable seeds, and X_{best} represents the location of the finest seed. The temporary triangle is formed by these three vertices, and each of them is indicated by one of the colors red (MG_i), blue (X_i), and green (GB) colors mark each of the selected vertices. A dice is taken with two red faces, two blue faces, and two green faces. Figure 2 shows the temporary triangle.

It has been shown that there are four ways to control and change the CGO algorithm's exploration and exploitation rate by manipulating the movement constraints of the seeds. Following is a presentation of four distinct formulations for α_i [26]:

$$\alpha_i = \begin{cases} \text{rand} \\ 2 \times \text{rand} \\ (\delta \times \text{rand}) + 1 \\ (\varepsilon \times \text{rand}) + (\sim \varepsilon) \end{cases} \quad (1)$$

where *rand* denotes to random number in the interval of [0,1] with uniformly distributed, while δ and ε are random integers in the interval of [0,1]. As the dice are rolled, the ith seed in its position is moved toward the corresponding

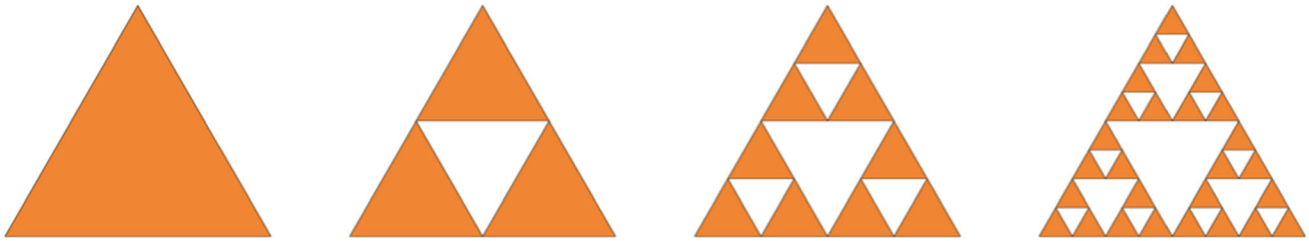

Fig. 1 The process of creating the Sierpinski triangle

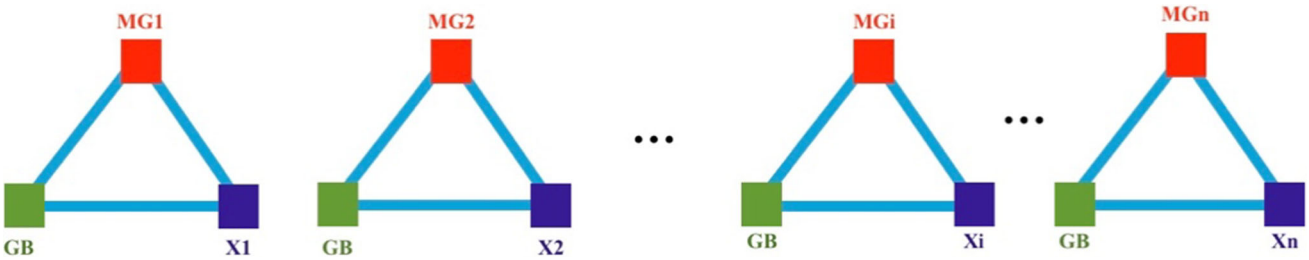

Fig. 2 The temporary triangles in the search space

vertex based on which color comes up. The dice are modeled using a combination of three random factors α_i, β_i, and γ_i. Each initial seed contributes to the production of four other seeds, which are based on the other vertices of the temporary triangles as follows [26]:

$$X_{new}^{1} = X_i + \alpha_i^{p(1)} \times (\beta_i \times GB - \gamma_i \times MG_i) \quad (2)$$

$$X_{new}^{2} = GB + \alpha_i^{p(2)} \times (\beta_i \times MG_i - \gamma_i \times X_i) \quad (3)$$

$$X_{new}^{3} = MG_i + \alpha_i^{p(3)} \times (\beta_i \times GB - \gamma_i \times X_i) \quad (4)$$

$$X_{new}^{4} = X_i\left(x_i^k = x_i^k + R\right) \quad (5)$$

where k is a uniformly distributed random integer in the range [1, d], d is the number of design variables, and R is a uniformly distributed random number [0, 1]. In addition, β_i and γ_i are two random integers of 1 or 2. The probability of rolling the dice is modelled using β_i andγ_i. It's also worth noting that α_i produces four unique random vectors. The

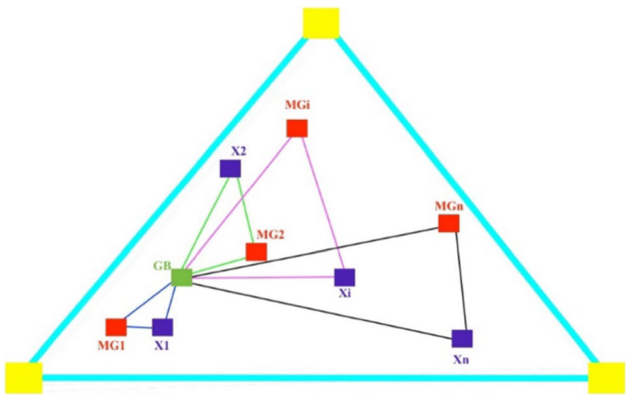

Fig. 3 The process of forming temporary triangles

exploration and exploitation rate of the CGO algorithm is controlled and adjusted by changing their order using a permutation between integers 1 to 4 as p. Until a termination requirement is satisfied, the process is carried out for each seed and repeated each iteration. A schematic representation of this procedure is shown in Fig. 3

3.2 Multi-objective chaos game optimization (MOCGO)

There is a wide variety of multi-objective algorithms and methods for solving complex challenges. Since no method or algorithm has ever been employed to solve a multi-objective problem with 100% efficiency, most academics are constantly looking for fresh ideas and methods with improved capabilities. In order to solve multi-objective issues, we proposed a multi-objective CGO method in this study. The findings section is where we expect to find a better function through comparison. Because it was designed to work with problems involving single-objective optimization, the CGO cannot be utilized directly for the purpose of resolving challenges involving multi-objective optimization. Therefore we introduce the multi-objective variation of CGO for addressing optimization problems in a way that simultaneously satisfies several requirements. The capability of CGO to carry out multi-objective optimization has recently been expanded with the addition of three new mechanisms. The mechanisms used are similar to those used by MOGWO [42], but the exploration and exploitation phases of MOCGO inherit from the CGO algorithm. Those mechanisms are discussed in detail as follows:

```
Initialize  the random values for initial particles of eligible seeds
Evaluate  the fitness function for given suitable seeds
Find  the GB
Get  the non-dominant responses and generate the archive
While  Iteration < Max-Iteration

    For each eligible seed
      Find α_i, β_i and γ_i by Equation (1)
      Find Leader
        Create a temporary triangle according to the leader
        Create new seeds by Equations (2)-(5)
          For each new seed
            Check the boundary limits for seeds
            Evaluate the fitness function associated with new seeds
          End for
    End for

Evaluate  the objective function for all seeds
Obtain  the non-dominated answers
Update  the archive in accordance with the discovered non-dominant responses

    If the archive is filled
      Use the grid technique for erasing existing archives
      Add the archive's new response
    Endif

    If any of the new solutions contributed to the repository are situated outside of the hypercubes
      Update the grids
    Endif

End while;
Return Archive
```

Fig. 4 Pseudo-code of the MOCGO algorithm

The Archive: A fixed-sized external archive is integrated into the CGO for saving non-dominated Pareto-optimal solutions obtained so far. The archive has its own special controller to decide which solutions are allowed in and which are not. The number of saved solutions is restricted in the archive. Archive outputs are measured against iteratively generated non-dominated solutions. Three possible scenarios:

1. It is not possible to add the new solution to the archive if there is at least one archive member that dominates the new solution.
2. If the newly proposed solution is superior to at least one of the existing solutions in the archive, then it may be considered to be included in the archive. In such a scenario, the repository will be able to include newly developed solutions.
3. If the new solution and archive solutions are not dominant, the new solution is added to the archive.

The grid mechanism: is the second effective mechanism integrated into CGO to enhance the non-dominated solutions in the archive. In the situation that the archive is already at full, the grid mechanism needs to be activated so that the segmentation of the objective space can be reorganized and a search can be conducted to identify the most congested area so that a solution may be removed from there. To improve diversity of the final approximated Pareto-optimal front, position the new solution in the least crowded area. If there are a greater number of potential solutions in the hypercube, there is a greater chance that one of those answers will be eliminated. If there is already a solution archive full, the most congested areas are found first, and a solution is intentionally deleted from one of them. A solution that lies outside of the hypercubes represents a unique circumstance. Each component of this scenario has been enhanced so that it can accommodate the most up-to-date solutions. As a consequence of this, components of various other solutions can also be altered.

The Leader Selection Mechanism: is the last machine included in CGO. This leader leads the other search agents toward areas of the search space that appear to have a good chance of providing a solution, with the end goal of obtaining a solution that is close to the global optimum. However, due to the Pareto optimality principles covered in the prior paragraph, it is difficult to compare the solutions in a multi-objective search space. To address this problem,

the leader selection process was created. As was already indicated, the best non-dominated solutions so far are archived. The leader selection component selects one of its non-dominated solutions and puts it in the search space's least congested regions. For each hypercube, the selection is performed by a roulette wheel with the following probability:

$$P_i = \frac{C}{N_i} \tag{6}$$

where N is the variety of acquired Pareto-optimal solutions in the i th section and C is a constant number greater than one.

According to Eq. (6), hypercubes with less congestion are more likely to employ new leaders. When there are fewer solutions available in a hypercube, that hypercube becomes a more likely candidate for leader selection. As the archive is optimized, its diversity is protected by the grid mechanism and the selection leader component. A low chance of selecting leaders from the most populated hypercubes is also provided by the leader selection component's use of a roulette wheel. This focuses on avoiding MOCGO at the local front.

Obviously, the MOCGO algorithm derives its convergence from the CGO method. If we pick one of the solutions from the archive, the MOCGO method will have an even higher level of consistency than it already possesses. On the other hand, it is difficult to determine which solutions are best according to the Pareto principle when there is a lot of variability. This issue was resolved by including the leader function collection and archive maintenance. The computational complexity of MOCGO is $O(mn^2)$, where n is the population size and m is the number of objectives to achieve. There is a significant improvement in computational complexity over traditional methods such as NSGA [43] and SPEA [44], which have $O(mn^3)$ complexity. MOCGO's pseudo-code is shown in Fig. 4.

4 Results and discussion

Performance measurements and case studies are used to figure out how well the methods in this section. These approaches include advanced multi-modal benchmark functions, real-world engineering design and mathematics problems. These problems are used to test how well multi-objective optimizers can handle non-convex and nonlinear constraints. Experiments are carried out using MATLAB software (R2021a) on a Macintosh (macOS Monterey) with a Core i9 processor and 16 GB of RAM.

4.1 Performance metrics

The algorithms' performance is evaluated using the following four metrics [45–47]:

Generational Distance is one of the measures that is utilized on a regular basis for the purpose of determining whether or not multi-objective metaheuristic optimization algorithms have converged. It measures the total distances between solution candidates obtained by different methods [48].

$$GD = \left(\frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} dis_i^2\right)^{\frac{1}{2}} \tag{7}$$

Solution candidates in separate sets achieved by various optimization techniques are called spacing (S) [49].

$$S = \left(\frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} \left(d_i - \overline{d}\right)^2\right)^{\frac{1}{2}} \quad \text{Where} \quad \overline{d} = \frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} d_i \tag{8}$$

The maximum spread (MS) in various solution sets refers to the spread of solution candidates in terms of the number of distinct optimal options and the number of possible solutions [50].

$$\text{MS} = \left[\frac{1}{m}\sum_{i=1}^{m}\left[\frac{\min\left(f_i^{\max}, F_i^{\max}\right) - \max\left(f_i^{\min}, F_i^{\min}\right)}{F_i^{\max} - F_i^{\min}}\right]^2\right]^{\frac{1}{2}} \tag{9}$$

The Inverted Generational Distance (IGD) is a statistic for comparing the Pareto front approximations obtained by various multi-objective algorithms [51].

$$\text{IGD} = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{10}$$

4.2 Experimental setting

This section compares the proposed multi-objective Chaos Game optimizer (MOCGO) to other well-known competitive approaches utilizing numerous benchmark problems. The comparisons were carried out in order to validate the suggested method's outcomes in terms of numerous standard performance measures such as IGD, MS, GD, and S. Several comparative methods have been used in the comparisons, including multi-objective Particle Swarm Optimizer (MOPSO) [17], multi-objective Gray Wolf Optimizer (MOGWO) [52], multi-objective Ant Lion Optimizer (MOALO) [53], multi-Objective Crystal Structure Algorithm (MOCryStAl) [38], multi-objective Harris Hawks Optimization (MOHHO) [54] and multi-objective Salp Swarm Algorithm (MSSA) [55]. The population size (number of tested solutions (N)) and the total number of

Table 1 Parameters of methods

Parameters	MOPSO	MOGWO	MOALO	MoCrStAl	MOHHO	MSSA	MOCGO
Mutation probability ($P_w, orpro$)	0.5	–	–	–	0.5	–	–
Population size (N_{pop})	100	100	100	100	100	100	100
Archive size (N_{rep}, orTM)	100	100	100	100	100	100	100
Number of adaptive grid (N_{grid})	30	30	30	30	30	30	30
Personal learning coefficient (C_1)	1	–	–	–	1	–	–
Global learning coefficient (C_2)	2	–	–	–	2	–	–
Inertia weight (w)	0.4	–	–	–	0.4	–	–
Beta	4	4	4	4	4	4	4
Gamma	2	2	2	2	2	2	2

tested iterations (T) of all tested algorithms are fixed as 50 and 1000, respectively. The parameter settings of the comparative methods are taken from the original paper, which is presented in Table 1. The used benchmark functions in the experiments are presented in Tables 2, 3, and 8.

4.2.1 Discussion of the CEC-09 test function

The outcomes of the comparison methods using the CEC-09 are provided in the following section.

Tables 2 and 3 include descriptions of the evaluated Bi-objective and Tri-objective CEC-09 benchmark functions. These problems are usually used to evaluate the performance of the multi-objective methods in the literature. The following section contains the findings of the comparison approaches.

Table 4 provides the statistical findings of CEC-09 benchmark functions in terms of IGD performance measures. The findings reveal clearly that the suggested MOCGO produced outstanding outcomes compared to previous methodologies. MOCGO got the best results in six out of ten test cases in several problems (i.e., UF2, UF3, UF4, UF8, UF9, and UF10), followed by MOGWO, which got the best results in some problems (i.e., UF5, UF6, and UF7), three out of ten test cases, and MOPSO got the best results in a problem (i.e., UF1), one out of ten test cases. The results shown in Table 4 show the strength of the proposed method in solving various complex problems with multiple objectives compared to other similar methods used in the literature. The proposed modifications to the new MOCGO method clearly helped improve the results and obtain substantial results in all comparisons, which confirms the ability of the proposed MOCGO method to solve such problems. These problems are usually hard to solve by the traditional method, and the method that gets excellent results can be considered an advanced search method to solve any complicated problem.

Table 5 analyzes CEC-09 benchmark functions using GD performance metrics. The findings clearly demonstrate that the proposed MOCGO outperformed previous comparing approaches. MOCGO consistently achieved the top outcomes in various challenges in six out of ten test cases, followed by MOGWO, which acquired the best results in some problems (i.e., UF1, UF5, and UF6), three out of ten test cases, and MOPSO got the best results in a problem (i.e., UF7), one out of ten test cases. The findings in Table 5 demonstrate the suggested method's resilience for handling various complicated situations with multiple objectives compared to other comparable techniques in the literature. The proposed new MOCGO method clearly improved the results and achieved substantial results in all measurements, confirming the strength of the proposed MOCGO method in solving such problems. The SD values showed that the proposed approach produced consistent results. We concluded from these results that the proposed multi-objective method is active and can solve complicated problems.

Table 6 gives the statistical outcomes of CEC-09 benchmark functions in terms of MS performance metrics. The results show that the suggested MOCGO produced better than other comparative methods. MOCGO obtained the best results in several test problems in eight out of ten test cases. MOPSO got the best results in a few other problems (i.e., UF3 and UF8), two out of ten test cases. The results presented in Table 6 confirm the quality of the obtained results produced by the proposed MOCGO method for tackling different complex problems with multiple objectives compared to similar methods employed in the literature. The proposed novel MOCGO method developed the results. It produced better results in all mentioned measurements, proving the robustness of the proposed MOCGO method to address such problems. Moreover, according to these results, the proposed method

Table 2 Bi-objective CEC-09 benchmark functions

Function	Mathematical formulation	D	Range
UF1	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}\left[\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)\right]^2, \quad f_2 = 1 - \sqrt{x}$ $+ \frac{2}{\lvert J_2\rvert}\sum\left[\left(x_j - \sin^{2}_{j\in J_2}(6\pi x_1 + \frac{j\pi}{n})\right)\right] J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, \qquad J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$	30	$x_1 \in [\,0,1]$ $x_i \in [-1,1]$ $i = 1,\ldots,D$
UF2	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} y_j^2, f_2 = 1 - \sqrt{x} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} y_j^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}, J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$ $y_j = \begin{cases} x_j - \left[\left(0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\cos\left(6\pi x_1 + \frac{j\pi}{n}\right) if\ j \in J_1 \\ x_j - \left[\left(0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\cos(6\pi x_1 + \frac{j\pi}{n}) if\ j \in J_2 \end{cases}$	30	$x_1 \in [\,0,1]$ $x_i \in [-1,1]$ $i = 1,\ldots,D$
UF3	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1} cos(\frac{20y_j\pi}{\sqrt{j}}) + 2) f_2 = \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2} cos(\frac{20y_j\pi}{\sqrt{j}}) + 2)$ J_1 and J_2 are the same as those of UF1, $y_j = x_j - x_1^{0.5\left(1.0+\frac{3(j-2)}{n-2}\right)}, j = 2,3,\ldots,n$	30	$x_i \in [\,0,1]$
UF4	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} h(y_j), f_2 = 1 - x_2 + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} h(y_j)$ J_1 and J_2 are the same as those of UF1, $y_j = x_j - sin(6\pi x_1 + \frac{j\pi}{n}), j = 2,3,\ldots,n$ $h(t) = \frac{\lvert t\rvert}{1+e^{2\lvert t\rvert}}$	30	$x_1 \in [\,0,1]$ $x_i \in [-2,2]$ $i = 1,\ldots,D$
UF5	$f_1 = x_1 + (\frac{1}{2N} + \varepsilon)\lvert sin(2N\pi x_1)\rvert + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} h(y_j) f_1 = 1 - x_1 + (\frac{1}{2N} + \varepsilon)\lvert sin(2N\pi x_1)\rvert + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} h(y_j)$ J_1 and J_2 are identical to those of UF1, $\varepsilon > 0, y_j = x_j - sin(6\pi x_1 + \frac{j\pi}{n}), j = 2,3,\ldots,n$ $h(t) = 2t^2 - cos(4\pi t) + 1$	30	$x_1 \in [\,0,1]$ $x_i \in [-1,1]$ $i = 1,\ldots,D$
UF6	$f_1 = x_1 + \max\{0, 2(\frac{1}{2N} + \varepsilon)\lvert \sin(2N\pi x_1)\rvert\} + \frac{2}{\lvert J_1\rvert}\left(\left(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 1\right)\right)$ $f_2 = 1 - x_1 + \max\{0, 2(\frac{1}{2N} + \varepsilon)\lvert \sin(2N\pi x_1)\rvert\} \frac{2}{\lvert J_1\rvert}\left(\left(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos(\frac{20y_j\pi}{\sqrt{j}} + 1\right)\right)$ J_1 and J_2 are identical to those of UF1, $\varepsilon > 0, y_j = x_j - sin(6\pi x_1 + \frac{j\pi}{n}), j = 2,3,\ldots,n$	30	$x_1 \in [\,0,1]$ $x_i \in [-1,1]$ $i = 1,\ldots,D$
UF7	$f_1 = \sqrt[5]{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_1} y_j^2, f_2 = 1 - \sqrt[5]{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} y_j^2$ J_1 and J_2 are identical to those of UF1, $\varepsilon > 0, y_j = x_j - sin(6\pi x_1 + \frac{j\pi}{n}), j = 2,3,\ldots,n$	30	$[x_1 \in [\,0,1]$ $x_i \in [-1,1]$ $i = 1,\ldots,D$

Table 3 Tri-objective CEC-09 benchmark functions

Function	Mathematical formulation	D	Range
UF8	$f_1 = cos(0.5x_1\pi)cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $f_2 = cos(0.5x_1\pi)sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $f_3 = sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $J_1 = \{j\lvert 3 \le j \le n, and\ j-1\ is\ a\ multiplication\ of\ 3\}$ $J_2 = \{j\lvert 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$ $J_3 = \{j\lvert 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3\}$	30	$x_1 \in [\,0,1]$ $x_2 \in [\,0,1]$ $x_i \in [-\,2,2]$ $i = 1,\ldots,D$
UF9	$f_1 = 0.5[max\{0,(1+\varepsilon)(1-4(2x_1-1)^2)\} + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $f_2 = 0.5[max\{0,(1+\varepsilon)(1-4(2x_1-1)^2)\} + 2x_1]x_2 + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $f_3 = 1 - x_2 + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2 sin(2\pi x_1 + \frac{j\pi}{n})^2)$ $J_1 = \{j\lvert 3 \le j \le n, and\ j\,1\ is\ a\ multiplication\ of\ 3\}$ $J_2 = \{j\lvert 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$ $J_3 = \{j\lvert 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3\}, \varepsilon = 0.1$	30	$x_1 \in [\,0,1]$ $x_2 \in [\,0,1]$ $x_i \in [-\,2,2]$ $i = 1,\ldots,D$
UF10	$f_1 = cos(0.5x_1\pi)cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}[4y_j^2 - cos(8\pi y_j) + 1]$ $f_2 = cos(0.5x_1\pi)sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_1}[4y_j^2 - cos(8\pi y_j) + 1]$ $f_3 = sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}[4y_j^2 - cos(8\pi y_j) + 1]$ $J_1 = \{j\lvert 3 \le j \le n, and\ j-1\ is\ a\ multiplication\ of\ 3\}$ $J_2 = \{j\lvert 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$ $J_3 = \{j\lvert 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3$	30	$x_1 \in [\,0,1]$ $x_2 \in [\,0,1]$ $x_i \in [-\,2,2]$ $i = 1,\ldots,D$

Table 4 Statistical analysis of the CEC-09 benchmark function to determine IGD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
UF1	Ave	**5.3291E-03**	6.2723E-03	6.1666E-03	6.7547E-03
	SD	2.9590E-03	2.3573E-03	6.0132E-04	**3.3961E-04**
UF2	Ave	4.1586E-03	3.1171E-03	5.9619E-03	**3.0229E-03**
	SD	4.8128E-04	1.4449E-04	4.6549E-04	**1.2127E-04**
UF3	Ave	1.7231E-02	1.1759E-02	1.2781E-02	**1.1582E-02**
	SD	1.0806E-03	1.0876E-03	1.6226E-03	**9.2291E-04**
UF4	Ave	2.8277E-03	2.7729E-03	3.5940E-03	**2.4541E-03**
	SD	2.5695E-04	4.2623E-04	1.0043E-03	**1.0026E-04**
UF5	Ave	3.4235E-01	**3.1605E-01**	3.5831E-01	5.3137E-01
	SD	1.5317E-01	1.0235E-01	9.7301E-02	**2.8499E-02**
UF6	Ave	2.9255E-02	**1.6384E-02**	2.6411E-02	3.8886E-02
	SD	9.1394E-03	**1.4457E-03**	6.2357E-03	3.4148E-03
UF7	Ave	4.3417E-03	**4.2653E-03**	7.8064E-03	1.0480E-02
	SD	2.4234E-03	2.0537E-03	3.6188E-03	**1.5310E-03**
UF8	Ave	9.2776E-03	4.4972E-03	6.8487E-03	**3.3240E-03**
	SD	1.6147E-03	9.9923E-04	1.4396E-03	**2.9643E-04**
UF9	Ave	1.2300E-02	4.1086E-03	6.7235E-03	**4.0121E-03**
	SD	2.9460E-03	9.5194E-04	1.1738E-03	**4.1190E-04**
UF10	Ave	6.3301E-02	3.4391E-02	4.0477E-02	**1.1692E-02**
	SD	1.3765E-02	9.0787E-03	1.2864E-02	**3.1689E-03**

The bold number is the best result among other methods

got more Pareto-optimal solutions than the other comparative algorithms in the decision space.

Table 7 summarizes the statistical findings for CEC-09 benchmark functions in terms of S performance metrics. The conclusions demonstrated indisputably that the suggested MOCGO approach outperformed previous comparable methodologies. (MOPSO, MOGWO, and MOALO). MOCGO achieved the most reliable results in various test problems (i.e., UF2, UF3, UF5, UF8, UF9, and UF10), in six out of ten test cases. MOALO produced the best results in other few problems (i.e., UF1, UF4, UF6, and UF7), in four out of ten test cases. The results shown in Table 7 verify the quality of the acquired results presented by the proposed MOCGO method for addressing several complex problems with multiple objectives compared to other comparable methods used in the literature. The proposed MOCGO method obviously improved the results. It yielded clearly better results in terms of all considered measurements, demonstrating the robustness of the suggested MOCGO method to address such optimization problems. The SD values also showed that the proposed strategy consistently produced similar outcomes independent of evaluation measures.

Figures 5 and 6 represent the best PF obtained on CEC-09 problems by MOPSO, MOGWO, MOALO, and the

Table 5 Statistical analysis of the CEC-09 benchmark function to determine GD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
UF1	Ave	2.7430E-02	3.4966E-02	**1.5372E-02**	7.7021E-02
	SD	2.5390E-02	3.1994E-02	**4.1474E-03**	3.7672E-02
UF2	Ave	2.0656E-02	1.1215E-02	1.9644E-02	**9.3716E-03**
	SD	4.6831E-03	2.1213E-03	4.0405E-03	**2.1178E-03**
UF3	Ave	9.6010E-02	7.2062E-02	3.9887E-02	**2.4564E-02**
	SD	2.3482E-02	1.0267E-02	6.2737E-03	**1.1926E-03**
UF4	Ave	9.2957E-03	1.0169E-02	5.7701E-03	**4.1831E-03**
	SD	9.2862E-04	3.4790E-03	7.4405E-04	**3.2217E-04**
UF5	Ave	4.2733E-01	2.8272E-01	**1.7752E-01**	7.4549E-01
	SD	2.9837E-01	1.5805E-01	**5.7170E-02**	1.0797E-01
UF6	Ave	2.0278E-01	1.1532E-01	**8.6731E-02**	4.6623E-01
	SD	1.6569E-01	5.0468E-02	**3.0128E-02**	1.8945E-01
UF7	Ave	1.7938E-02	**9.1316E-03**	1.3805E-02	8.3221E-02
	SD	1.3813E-02	1.2322E-02	**3.3199E-03**	3.3881E-02
UF8	Ave	2.6587E-01	3.3914E-02	4.5825E-02	**9.4878E-03**
	SD	8.2205E-02	1.3307E-02	2.9935E-02	**3.1263E-03**
UF9	Ave	4.0311E-01	5.1960E-02	4.5643E-02	**1.4653E-02**
	SD	7.9920E-02	2.1763E-02	1.1497E-02	**9.3088E-03**
UF10	Ave	1.3828E+00	6.9388E-01	4.5870E-01	**2.7876E-01**
	SD	2.1017E-01	2.0777E-01	1.6020E-01	**9.0385E-02**

The bold number is the best result among other methods

Table 6 Statistical analysis of the CEC-09 benchmark function to determine MS performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
UF1	Ave	8.7689E-01	8.9878E-01	9.2888E-01	**1.5057E+00**
	SD	4.5456E-01	**2.0878E-01**	2.3480E-01	7.2921E-01
UF2	Ave	1.1989E+00	1.1436E+00	1.1170E+00	**1.2489E+00**
	SD	2.3245E-01	6.4565E-02	1.6472E-01	**5.1575E-02**
UF3	Ave	**1.4879E+00**	1.3245E+00	8.3938E-01	9.6572E-01
	SD	8.9801E-01	8.6759E-02	2.4240E-01	**7.2458E-02**
UF4	Ave	1.0034E+00	1.0148E+00	8.3622E-01	**1.0304E+00**
	SD	3.3432E-02	7.7681E-03	8.1759E-02	**1.2536E-03**
UF5	Ave	7.7667E-01	1.7603E+00	1.7252E+00	**2.3944E+00**
	SD	6.4309E-01	7.7653E-01	4.8531E-01	**3.9402E-01**
UF6	Ave	9.7689E-01	1.4378E+00	9.8110E-01	**2.5381E+00**
	SD	1.3260E+00	**4.3259E-02**	3.9658E-01	2.0853E +00
UF7	Ave	1.2435E+00	1.1324E+00	8.8819E-01	**1.6926E+00**
	SD	2.5467E-01	**5.6578E-02**	3.0321E-01	3.5168E-01
UF8	Ave	**5.4325E +00**	8.9965E-01	5.7743E-01	1.1129E +00
	SD	2.9098E+00	3.3248E-01	4.9527E-01	**7.6427E-02**
UF9	Ave	6.4379E+00	2.3432E+00	4.1219E-01	**8.0200E+00**
	SD	1.2321E+00	3.2332E-01	2.6948E-01	**1.4182E-01**
UF10	Ave	5.9867E+00	5.9987E+00	3.1496E+00	**6.9151E+00**
	SD	3.0934E+00	1.2143E+00	1.1075E+00	**1.0023E+00**

The bold number is the best result among other methods

Table 7 Statistical analysis of the CEC-09 benchmark function to determine S performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
UF1	Ave	1.5649E-02	5.8711E-02	**3.8318E-03**	5.3728E-02
	SD	9.9871E-03	4.3083E-02	**7.2845E-03**	2.7475E-02
UF2	Ave	1.8565E-02	1.4426E-02	1.8319E-02	**1.3254E-02**
	SD	3.1440E-03	4.0894E-03	1.4069E-02	**2.5215E-03**
UF3	Ave	2.6957E-02	7.3997E-02	1.1187E-02	**1.0556E-02**
	SD	1.5713E-02	2.5622E-02	5.2701E-03	**1.8936E-03**
UF4	Ave	1.0182E-02	2.0138E-02	**3.4636E-03**	1.1082E-02
	SD	1.7583E-03	1.0792E-02	3.0832E-03	**1.6234E-03**
UF5	Ave	2.5756E-02	8.9494E-02	1.4277E-02	**1.0392E-02**
	SD	3.3090E-02	7.0701E-02	9.0182E-03	**8.7434E-03**
UF6	Ave	4.3768E-02	1.3105E-01	**4.5527E-03**	1.9256E-01
	SD	7.4660E-02	1.8372E-01	**3.6726E-03**	1.2531E-01
UF7	Ave	1.6883E-02	2.6653E-02	**6.1348E-03**	6.0614E-02
	SD	8.2967E-03	4.5707E-02	**5.6661E-03**	2.1989E-02
UF8	Ave	2.7757E-01	4.4312E-02	2.3043E-02	**2.2881E-02**
	SD	7.9069E-02	1.1641E-02	1.6247E-02	**1.0886E-02**
UF9	Ave	3.7998E-01	6.6954E-02	1.7566E-02	**1.6721E-02**
	SD	8.9375E-02	2.2886E-02	1.1348E-02	**1.0324E-02**
UF10	Ave	1.0998E+00	4.1590E-01	2.2214E-01	**2.0060E-01**
	SD	1.9502E-01	1.3872E-01	9.2452E-02	**8.5406E-02**

The bold number is the best result among other methods

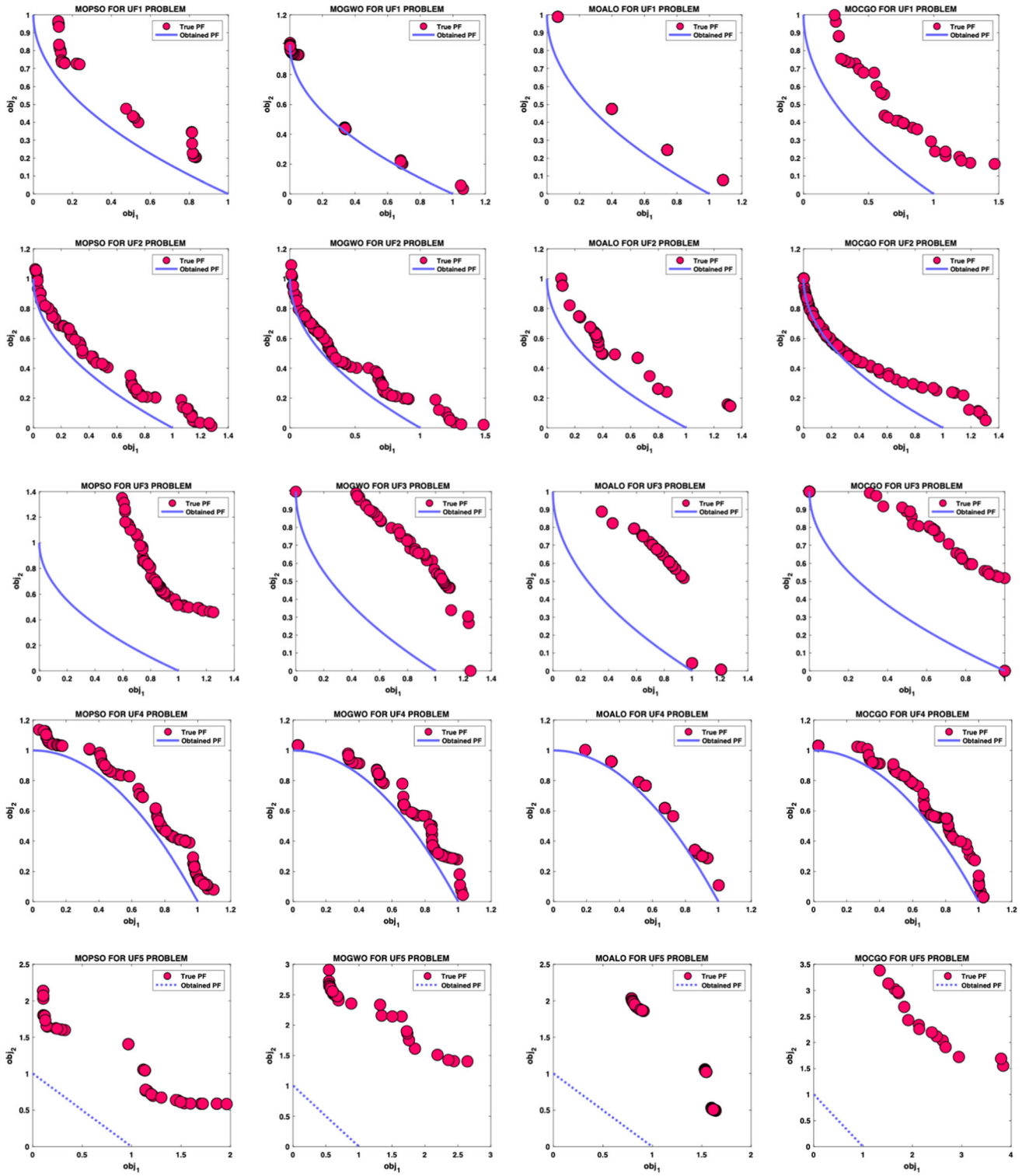

Fig. 5 The CEC-09 problems (UF1-5) True and obtained Pareto front results

proposed MOCGO algorithms. Figure 5 depicts the outcomes of the comparative methodologies on UF1-UF5, and Fig. 6 illustrates the results on UF6-UF10. Based on these figures, it can be shown that the proposed MOCGO displays a perfect convergence as it gets closer and closer to all of the true Pareto-optimal fronts. Moreover, the MOPSO, MOGWO, and MOALO methods explain the worst convergence, corresponding with the obtained results. The suggested approach is compared to other well-known comparison methods on the map-based problem to

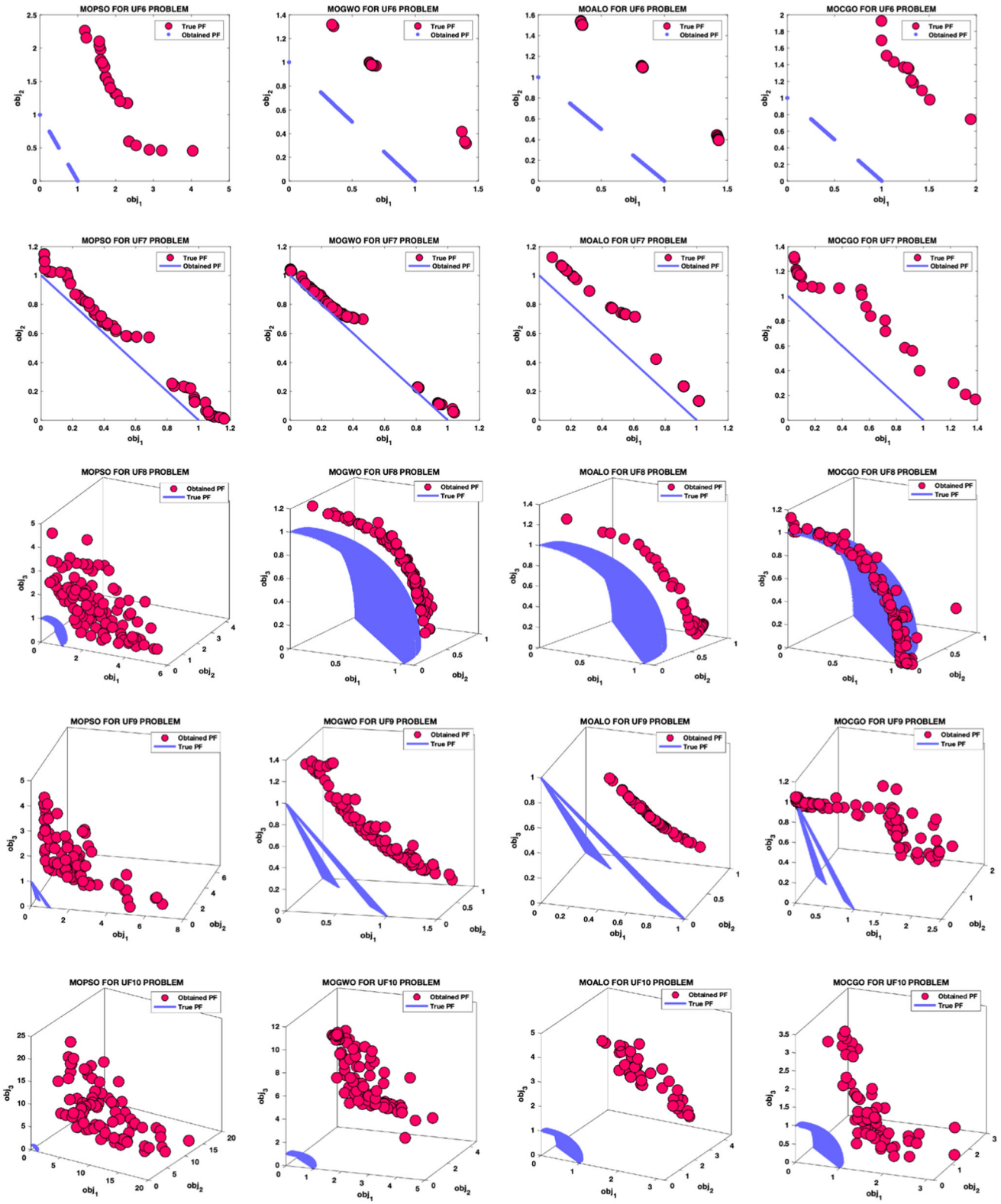

Fig. 6 The CEC-09 problems (UF6-10) True and obtained Pareto front results

illustrate its usefulness. MOCGO can cover all Pareto areas, although the optimum regions reported by others in the literature are partial, as demonstrated in Figs. 5 and 6. This demonstrates MOCGO's excellent performance and demonstrates its efficacy.

4.2.2 Discussion of the ZDT and DTLZ test function

The advanced multi-modal benchmark functions with fixed-dimension, including ZDT (i.e., ZDT1-ZDT6) and DTLZ (DTLZ2 and DTLZ4), are tested to validate further the performance of the proposed MOCGO algorithm in the following section. The findings that were achieved using the proposed approach are compared with the results acquired using other comparison methods that are well-known (i.e., MOPSO, MOGWO, and MOCGO). The descriptions of the tested Multi-modal benchmark functions with fixed-dimension are presented in Table 8.

Benchmark functions ZDT and DTLZ, which measure GD performance, are statistically compared in Table 9. When compared to previous approaches, the suggested MOCGO performed exceptionally well. MOCGO got the best results in several problems (i.e., ZDT1, ZDT2, ZDT3 and ZDT4,), in five out of seven test cases. Followed by MOPSO, it got the best results in some problems (i.e., ZDT6, DTLZ2), two out of seven test cases, and MOGWO got the best results in a problem (i.e., DTLZ4), one out of seven test cases. The results shown in Table 9 compare the proposed method to similar approaches that have been used to solve advanced difficult issues with multiple objectives. According to the findings, the approach that was suggested is superior to others in this regard. In addition, the standard deviation values demonstrated that the suggested method is capable of producing results that are consistent across multiple instances.

Table 10 summarizes the statistical outcomes for the ZDT and DTLZ benchmark functions using IGD. Compared to other comparison algorithms, the results show that MOCGO performed quite well. MOCGO achieved the best results in five problems (i.e., ZDT1, ZDT2, ZDT3, ZDT4, ZDT6). MOPSO finished in second place, achieving the best possible scores in two of the seven tests (DTLZ2 and DTLZ4). In contrast to other comparable approaches utilized in the literature, Table 10 demonstrates the power of the suggested method in addressing various advanced complicated problems with multiple objectives. In addition, the SD values demonstrated that the suggested method is capable of producing results that are consistent across a range of different applications.

Tables 11 and 12 show the ZDT and DTLZ benchmark functions' MS and S performance. The findings indicate that the suggested MOCGO produced outstanding results compared to previous comparison algorithms. According to the MS measure results in Table 11, out of seven test instances, MOCGO achieved the best results in six problems (i.e., ZDT2, ZDT3, ZDT6, DTLZ2, and DTLZ4). MOGWO finished in second place, achieving the highest marks in one of the seven different tests (ZDT4). The results of the S measure are presented in Table 12, and it can be seen that out of a total of seven different test cases, MOCGO got the best results in three of them (i.e., ZDT3, ZDT6, and DTLZ4). MOPSO came in second, with the best results in two of the seven test cases (i.e., ZDT2 and ZDT4). MOGWO got the best results in one case (i.e.,

Table 8 Multi-modal benchmark functions with fixed-dimension

Function	Mathematical formulation	D	Range
ZDT1	$F_1 = x_1, F_2 = g\left(1 - \sqrt{\frac{F_1}{g}}\right), g = 1 + \frac{9}{d-1}\sum_{i=2}^{d} x_i$	30	$x_i \in [\,0,1]$
ZDT2	$F_1 = x_1, F_2 = g\left(1 - \left(\frac{F_1}{g}\right)^2\right), g = 1 + \frac{9}{d-1}\sum_{i=2}^{d} x_i$	30	$x_i \in [\,0,1]$
ZDT3	$F_1 = x_1, F_2 = g\left(1 - \sqrt{F_1 g} - {F_1}/{g}(10\pi F_1)\right), g = 1 + \frac{9}{d-1}\sum_{i=2}^{d} x_i$	30	$x_i \in [\,0,1]$
ZDT4	$F_1 = x_1, \quad F_2 = g\left(1 - \sqrt{{F_1}/{g}}\right), \quad g = 1 + 10(d-1) + \sum_{i=2}^{d}(x_i^2 - 10\cos(4\pi x_i))$	10	$x_1 \in [\,0,1]$ $x_i \in [-\,5,5]$ $i = 1, \ldots, D$
ZDT6	$F_1 = 1 - \exp(-4x_1)\sin^6(6\pi x_1), \quad F_2 = g\left(1 - \left({F_1}/{g}\right)^2\right), \quad g = 1 + 9\left(\frac{\sum_{i=2}^{d} x_i}{d-1}\right)^{0.25}$	10	$x_i \in [\,0,1]$
DTLZ2	$F_1 = (1+g)\cos\left(x_1\left(\frac{\pi}{2}\right)\right)\cos\left(x_2\left(\frac{\pi}{2}\right)\right), \mathrm{F}_2 = (1+\mathrm{g})\cos\left(x_1\left(\frac{\pi}{2}\right)\right)\sin\left(x_2\left(\frac{\pi}{2}\right)\right),$ $F_3 = (1+g)\sin\left(x_1\left(\frac{\pi}{2}\right)\right), \mathrm{g} = \sum_{i=3}^{d}(x_i - 0.5)^2$	12	$x_i \in [\,0,1]$
DTLZ4	$F_1 = (1+g)\cos\left(x_1^{\pi}\left(\frac{\pi}{2}\right)\right)\cos\left(x_2^{\pi}\left(\frac{\pi}{2}\right)\right), \mathrm{F}_2 = (1+\mathrm{g})\cos\left(x_1^{\pi}\left(\frac{\pi}{2}\right)\right)\sin\left(x_2^{\pi}\left(\frac{\pi}{2}\right)\right),$ $F_3 = (1+g)\sin\left(x_1^{\pi}\left(\frac{\pi}{2}\right)\right), \mathrm{g} = \sum_{i=3}^{d}(x_i - 0.5)^2$	12	$x_i \in [\,0,1]$

Table 9 Statistical analysis of the mathematical functions to determine GD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
ZDT1	Ave	1.7932E-03	5.8093E-05	1.3644E-04	**5.0355E-05**
	SD	5.0496E-03	4.2240E-05	1.7358E-04	**1.1710E-05**
ZDT2	Ave	1.4749E-01	3.7956E-05	4.0621E-05	**3.5936E-05**
	SD	6.5211E-02	2.0061E-05	1.9757E-05	**3.1526E-06**
ZDT3	Ave	2.0489E-04	2.0332E-04	1.7253E-03	**1.3825E-04**
	SD	**2.7322E-05**	9.0168E-05	1.1636E-03	2.0852E-05
ZDT4	Ave	4.1299E+00	6.8746E+00	2.0045E+00	**1.2290E+00**
	SD	3.9519E+00	7.4640E+00	**9.6098E-01**	2.5919E+00
ZDT6	Ave	**2.5805E-02**	1.3278E-01	3.3034E-02	7.7697E-02
	SD	5.8008E-02	3.4317E-01	2.0337E-02	**2.0003E-02**
DTLZ2	Ave	**6.8600E-03**	7.2624E-03	2.1134E-02	3.0501E-02
	SD	9.0878E-04	**5.2125E-04**	1.0861E-02	8.8083E-03
DTLZ4	Ave	1.0097E-02	**1.8707E-03**	3.6328E-02	3.3886E-02
	SD	2.4570E-03	**3.8729E-04**	1.9865E-02	1.6069E-02

The bold number is the best result among other methods

Table 10 Statistical analysis of the mathematical functions to determine IGD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
ZDT1	Ave	8.0879E-04	1.2199E-03	1.5144E-02	**3.3448E-04**
	SD	1.4764E-03	3.4912E-04	2.1178E-03	**7.0516E-05**
ZDT2	Ave	5.2023E-02	6.4188E-03	2.1623E-02	**3.1708E-04**
	SD	9.5007E-03	8.6889E-03	1.7638E-03	**4.3667E-05**
ZDT3	Ave	2.6241E-04	4.8548E-04	3.7118E-03	**2.5402E-04**
	SD	3.9884E-05	6.8091E-05	1.6638E-03	**3.8345E-05**
ZDT4	Ave	7.0747E-01	3.6983E-01	6.1546E-01	**6.3264E-02**
	SD	3.1784E-01	3.3799E-01	2.4280E-01	**1.3281E-01**
ZDT6	Ave	8.2281E-03	3.5919E-03	3.4649E-03	**5.4121E-04**
	SD	2.4862E-02	8.5585E-03	1.8983E-03	**1.1214E-04**
DTLZ2	Ave	**4.6940E-04**	3.0120E-03	3.2518E-03	1.0827E-03
	SD	**2.6098E-05**	3.9605E-04	4.6848E-04	1.0522E-04
DTLZ4	Ave	**1.6840E-03**	7.3987E-03	1.1027E-02	3.6037E-03
	SD	**9.4057E-05**	3.7323E-04	2.9491E-03	2.0291E-04

The bold number is the best result among other methods

DTLZ2), and MOALO got the best results in one case (i.e., ZDT1). In contrast to other similar methods utilized in the literature, these results demonstrated the power of the suggested approach in addressing various advanced complicated problems with multiple objectives. The SD values confirmed the suggested technique's capability to provide consistent results.

Figures 7 and 8 show the best PF produced by MOPSO, MOGWO, MOALO, and the proposed MOCGO algorithms on ZDT and DTLZ problems. The results of the comparison techniques on ZDT (i.e., ZDT1-ZDT6) are shown in Fig. 7, and the results of the comparative methods on DTLZ (i.e., DTLZ2 and DTLZ4) are shown in Fig. 8. These diagrams demonstrate that the proposed

Table 11 Statistical analysis of the mathematical functions to determine MS performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
ZDT1	Ave	9.9870E-01	8.2895E-01	3.0049E-01	**9.9964E-01**
	SD	**2.3779E-02**	7.4545E-02	5.1645E-02	1.4642E-01
ZDT2	Ave	7.4311E-02	6.3944E-01	3.6133E-02	**1.0000E +00**
	SD	2.3499E-01	3.4118E-01	4.3001E-02	**0.0000E +00**
ZDT3	Ave	9.9865E-01	9.7302E-01	7.1037E-01	**9.9971E-01**
	SD	**1.7888E-03**	2.1153E-02	1.1021E-01	5.9020E-03
ZDT4	Ave	4.6863E-01	**6.2129E +00**	2.3408E+00	1.1529E+00
	SD	1.4819E+00	6.9292E+00	4.8105E+00	**3.5499E-01**
ZDT6	Ave	1.1170E+00	2.1639E+00	1.4199E+00	**5.3501E+00**
	SD	**5.3983E-01**	1.6010E+00	5.8815E-01	1.2843E+00
DTLZ2	Ave	1.0976E+00	8.9087E-02	1.5348E-01	**1.7522E+00**
	SD	9.0869E-02	**3.3330E-02**	6.5189E-02	5.4922E-01
DTLZ4	Ave	1.2437E+00	2.1199E-01	3.3046E-01	**1.7613E+0**
	SD	1.4187E-01	**6.5198E-02**	1.8953E-01	5.9416E-01

The bold number is the best result among other methods

Table 12 Statistical analysis of the mathematical functions to determine S performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
ZDT1	Ave	1.1319E-02	1.4894E-02	**5.1688E-03**	1.0769E-02
	SD	1.0762E-03	6.1897E-03	1.9499E-03	**1.0710E-03**
ZDT2	Ave	**7.3295E-04**	1.1600E-02	8.6826E-04	1.0949E-02
	SD	2.3178E-03	7.3141E-03	1.1868E-03	**1.0931E-03**
ZDT3	Ave	1.3270E-02	1.5123E-02	1.4655E-02	**1.2072E-02**
	SD	2.3561E-03	6.5176E-03	9.2716E-03	**2.0699E-03**
ZDT4	Ave	**4.3804E-03**	1.3446E+00	1.4977E-02	1.6506E-01
	SD	**1.3852E-02**	1.6767E +00	2.5415E-02	3.2301E-01
ZDT6	Ave	1.8109E-02	5.7668E-02	2.4910E-02	**1.7675E-02**
	SD	3.0071E-02	9.3879E-02	1.2571E-02	**1.0986E-02**
DTLZ2	Ave	5.7741E-02	**4.7074E-03**	1.1726E-02	1.0259E-01
	SD	4.9507E-03	**1.2812E-03**	5.3284E-03	4.6703E-02
DTLZ4	Ave	6.1316E-02	1.1737E-02	1.8314E-02	**1.0162E-02**
	SD	5.3260E-03	3.3199E-03	1.0509E-02	**1.2177E-03**

The bold number is the best result among other methods

MOCGO approaches all true Pareto-optimal fronts with almost complete convergence. Furthermore, the MOPSO, MOGWO, and MOALO approaches explain the poorest convergence.

4.2.3 Discussion of engineering problems

This section tests the proposed MOCGO on eight multi-objective engineering problems (see Appendix) of which some are discussed as follows:

4.2.3.1 The 4-bar truss In the well-known issue of structural optimization shown in Fig. 9, [56], the goal is to reduce both the volume (f_1) and the displacement (f_2) of a 4-bar truss to their smallest possible values. The following equations link four design variables ($x_1 - x_4$) to the cross-sectional area of members 1 to 4.

$$\text{Minimize} : f_1(x) = 200 \times (2 \times x_1 + \text{sqrt}(2 \times x_2) + \text{sqrt}(x_3) + x_4) \quad (11)$$

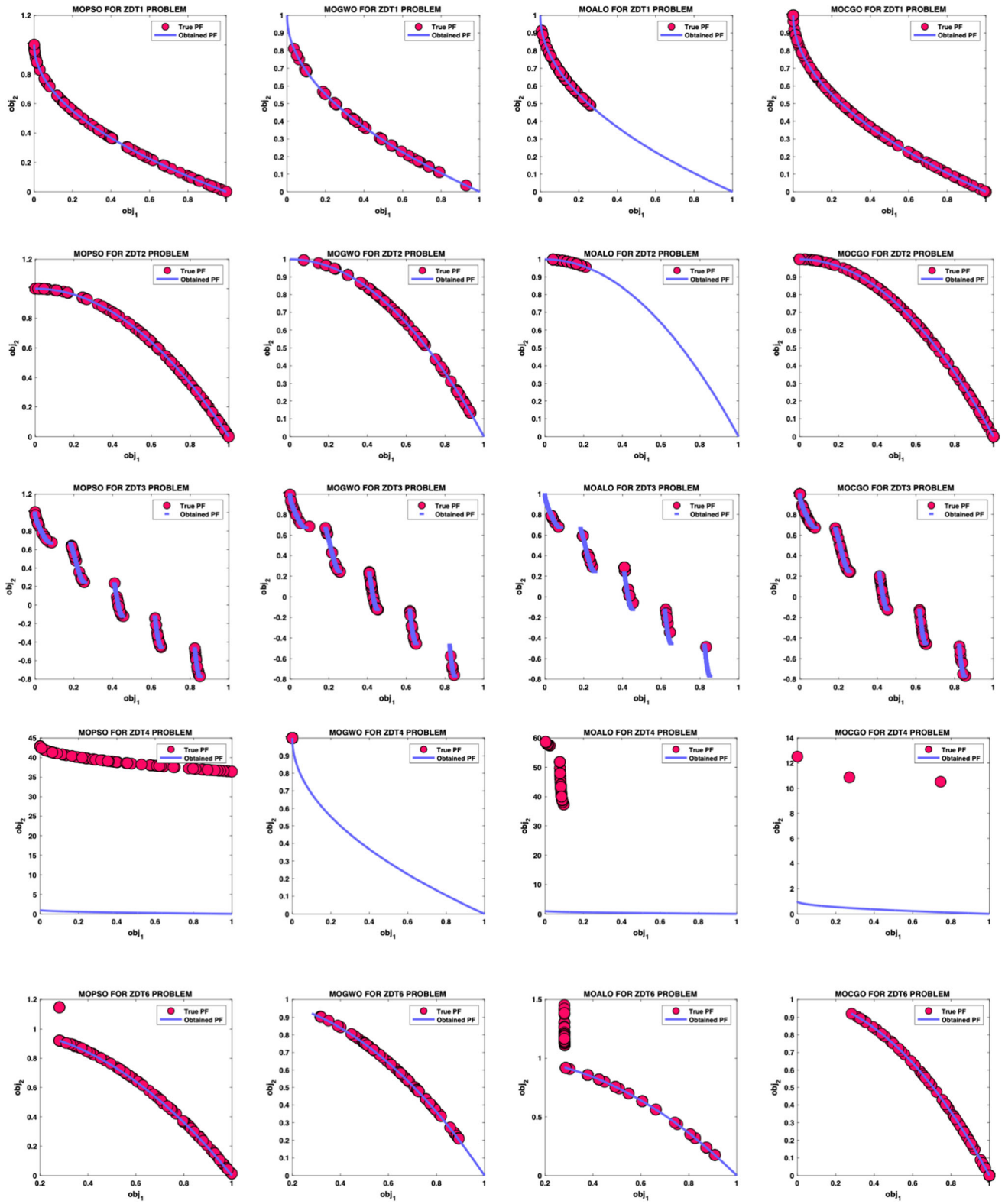

Fig. 7 True and obtained Pareto front for ZDT problems

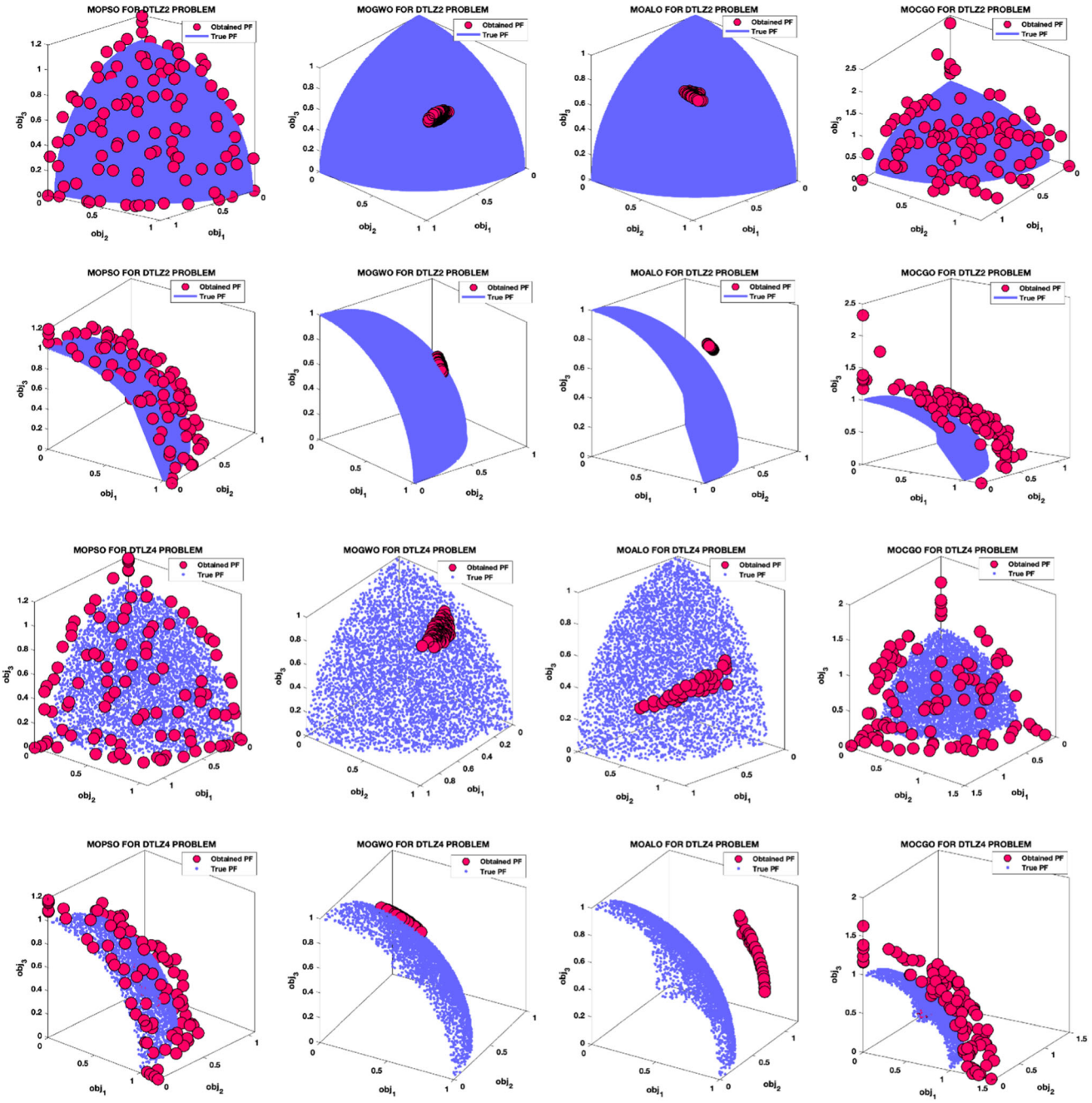

Fig. 8 The DTLZ problems True and obtained Pareto front results

$$\begin{aligned}\text{Minimize}: f_2(x) = 0.01 \times \left(\frac{2}{x_1}\right) + \left(\frac{2 \times \text{sqrt}(2)}{x_2}\right) \\ - \left(\frac{2 \times \text{sqrt}(2)}{x_3}\right) + (2/x_1)1 \leq x_1 \\ \leq 3, 1.4142 \leq x_2 \leq 3\ 1.4142 \leq x_3 \\ \leq 3, 1 \leq x_4 \leq 3\end{aligned} \tag{12}$$

4.2.3.2 The welded beam Ray and Liew [57] proposed four design restrictions for welded beams. Figure 10 illustrates this scenario in further detail. The welded beam is shown schematically in Fig. 10. The manufacturing cost (f_1) and beam deflection (f_2) of a welded beam should be kept to a minimum in this issue. The four design variables are the weld thickness (x_1), the clamped bar's length (x_2), the clamped bar's height (x_3) and the clamped bar's thickness (x_4).

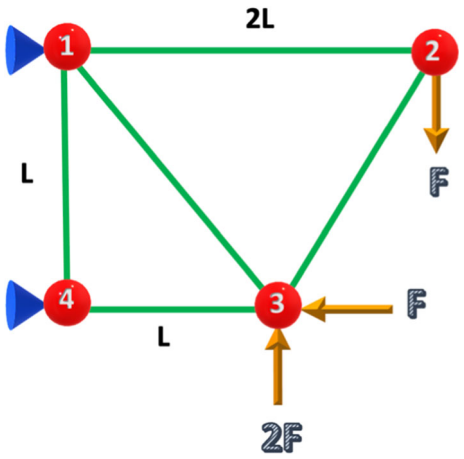

Fig. 9 The schematic view of the four-bar truss

$$\text{Minimize} : f_1(x) = 1.10471 \times x_1^2 \times x_2 + 0.04811 \times x_3 \times x_4 \times (14 + x_2) \tag{13}$$

$$\text{Minimize} : f_2(x) = 65856000/\left(30 \times 10^6 \times x_4 \times x_3^3\right) \tag{14}$$

$$\text{where} : g_1(x) = \tau - 13600 \tag{15}$$

$$g_2(x) = \sigma - 30000 \tag{16}$$

$$g_3(x) = x_1 - x_4 \tag{17}$$

$$g_4(x) = 6000 - P \tag{18}$$

$$0.125 \le x_1 \le 5, 0.1 \le x_2 \le 10$$

$$0.1 \le x_3 \le 10, 0.125 \le x_4 \le 5$$

$$\text{where} : q = 6000 * \left(14 + \frac{x_1}{2}\right); D = \text{sqrt}\left(\frac{x_2^2}{4} + \frac{(x_1 + x_3)^2}{4}\right) \tag{19}$$

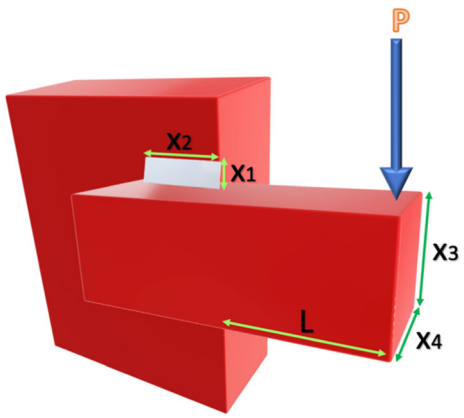

Fig. 10 The welded beam

$$J = 2 * \left(x_1 * x_2 * \text{sqrt}(2) * \left(\frac{x_2^2}{12} + \frac{(x_1 + x_3)^2}{4}\right)\right) \tag{20}$$

$$\alpha = \frac{6000}{\text{sqrt}(2) * x_1 * x_2} \tag{21}$$

$$\beta = Q * \frac{D}{J} \tag{22}$$

4.2.3.3 Disk brake According to Ray and Liew [56], there are many limitations to consider while designing a disc brake. Two goals need to be attained: reducing stopping time (f_1) and reducing brake mass (f_2). Figure 11 shows a schematic representation of the disc brake. Disc's inner radius (x_1), outer radius (x_2), engaging force (x_3), the number of friction surfaces (x_4), and five constraints are given below as equations.

$$\text{Minimize} : f_1(x) = 4.9 \times (10)^{(-5)} \times \left(x_2^{(2)} - x_1^{(2)}\right) \times (x_4 - 1) \tag{23}$$

$$\text{Minimize} : f_2(x) = \left(9.82 \times (10)^{(6)}\right) \times \left(x_2^{(2)} - x_1^{(2)}\right) / \left(\left(x_2\right)^{(3)} - x_1^{(3)}\right) \times x_4 \times x_3 \tag{24}$$

$$g_1(x) = 20 + x_1 - x_2 \tag{25}$$

$$g_2(x) = 2.5 + (x_4 + 1) - 30 \tag{26}$$

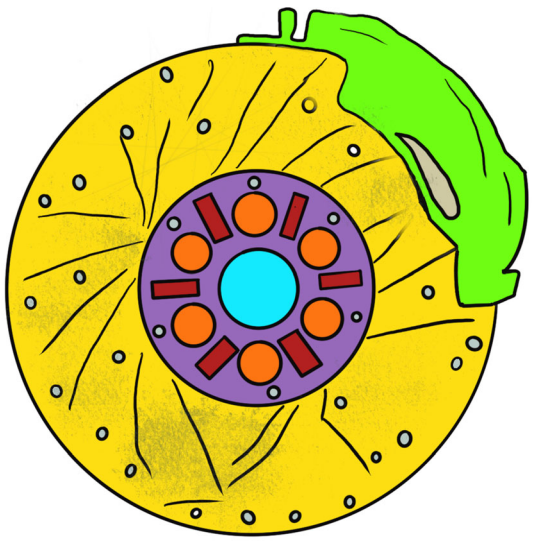

Fig. 11 The disk brakes

$$g_3(x) = (x_3)/\left(3.14 \times \left(x_2^{(2)} - x_1^{(2)}\right)^2\right) - 0.4 \quad (27)$$

$$g_4(x) = \left(2.22 \times (10)^{(-3)} \times x_3 \times \left((x_2)^{(3)} - x_1^{(3)}\right)\right)/\left(\left(x_2^{(2)} - x_1^{(2)}\right)^2\right) - 1 \quad (28)$$

$$g_5(x) = 900 - \left(2.66 \times (10)^{(-2)} \times x_3 \times x_4 \times \left((x_2)^{(3)} - x_1^{(3)}\right)\right)/\left(\left(x_2^{(2)} - x_1^{(2)}\right)^2\right) \quad (29)$$

$$55 \le x_1 \le 80, 75 \le x_2 \le 110$$

$$1000 \le x_3 \le 3000, 2 \le x_4 \le 20$$

4.2.3.4 Speed reducer It is well knowledge in mechanical engineering that the design of a speed reducer must minimize the component's mass (f_1) and stress (f_2) (see Fig. 12). The details of this example with seven variables and eleven constraints can be found in [56, 58].

$$\begin{aligned} \text{Minimize}: f_1(x) &= 0.7854 \times x_1 \times x_2^2 \\ &\times \left(3.3333 \times x_3^2 + 14.9334 + x_3\right) - 43.0934\ldots \\ &- 1.508 \times x_1 \times \left(x_6^2 + x_7^2\right) \end{aligned} \quad (30)$$

$$\begin{aligned} \dot{\eta}_i &= R_i(\psi_i)\upsilon_i \qquad i = 1, 2, \ldots, n \\ \dot{\upsilon}_i &= M_i^{-1}(-C_i(\upsilon_i) - D_i(\upsilon_i) - \tau_{wi} + \tau_i) \end{aligned} \quad (31)$$

$$\text{where}: g_1(x) = 27/\left(x_1 \times x_2^2 \times x_3\right) - 1 \quad (32)$$

$$g_2(x) = 397.5/\left(x_1 \times x_2^2 \times x_3^2\right) - 1 \quad (33)$$

$$g_3(x) = (1.93 \times x_4^3/\left(x_2 \times x_3 \times x_6^4\right) - 1 \quad (34)$$

$$g_4(x) = (1.93 \times x_5^3/\left(x_2 \times x_3 \times x_7^4\right) - 1 \quad (35)$$

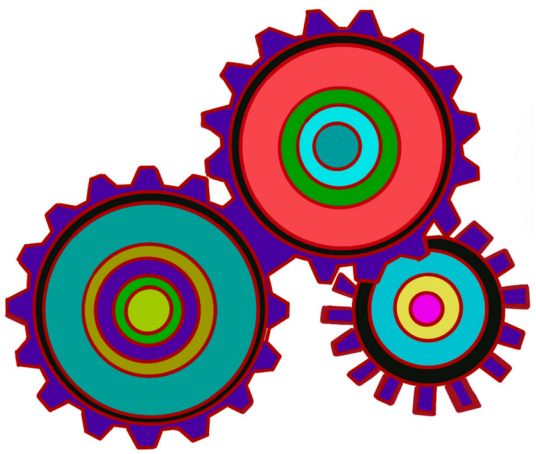

Fig. 12 The speed reducer

$$g_5(x) = \left(\text{sqrt}\left((745 \times x_4)/x_2 \times \left((x_3))^2 + 16.9e6\right)\right)/(110 \times x_6^3)\right) - 1 \quad (36)$$

$$g_6(x) = \left(\text{sqrt}\left((745 \times x_4)/x_2 \times \left((x_3))^2 + 157.9e6\right)\right)/(85 \times x_7^3)\right) - 1 \quad (37)$$

$$g_7(x) = ((x_2 \times x_3)/40) - 1 \quad (38)$$

$$\tau = \text{sqrt}\left(\alpha^2 + 2 \times \alpha \times \beta \times \frac{x_2}{2 \times D} + \beta^2\right) \quad (39)$$

$$\sigma = \frac{504000}{x_4 \times x_3^2} \quad (40)$$

$$\text{tmpf} = 4.013 \times \frac{30 \times 10^6}{196} \quad (41)$$

$$P = \text{tmpf} \times \text{sqrt}\left(x_3^2 \times \frac{x(4)^6}{36}\right) \times \left(1 - x_3 \times \frac{\text{sqrt}\left(\frac{30}{48}\right)}{28}\right) \quad (42)$$

4.2.3.5 Comparison of MOCGO with MOPSO, MOALO, MOGWO In this subsection, a comparison is made between MOCGO and the algorithms MOPSO, MOALO, and MOGWO for solving engineering problems based on the criteria of Ave and SDT. The outcomes of the comparison methodologies used in GD, IGD, MS, and S are presented in Tables 13, 14, 15 and 16, respectively. Table 13 demonstrates that the proposed strategy achieved promising outcomes in almost all of the situations that were put to the test using the GD measure. In comparison with MOGWO, MOPSO and MOALO achieved some of the best results; nevertheless, MOGWO did not get any of the best scores in that table. The findings of the comparison approaches for all of the problems that were examined are presented in Table 14, which summarizes IGD. The proposed method also proved its ability to solve real-world engineering problems effectively, which is also harmonized with the results in terms of MS and S, as shown in Tables 15 and 16. It can be concluded that the proposed method can solve complex problems with proven results using many tested problems. It can be considered an attractive alternative in this domain to solve multi-objective problems.

Figures 13 and 14 show the best PF produced by MOPSO, MOGWO, MOALO, and the proposed MOCGO algorithms on the given real-world industrial engineering problems. The results of the comparative methods on BNH, CONSTR, DISK BRAKE, and 4-BAR TRUSS are shown in Fig. 13. The results of the comparative methods on WELDED BEAM, OSY, SPEED REDUCER, and SRN are shown in Fig. 14. These diagrams confirm that the

Table 13 Statistical analysis of the engineering problems to determine GD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
P1: BNH	Ave	3.3894E-02	6.2465E-02	5.3758E-02	**3.2599E-02**
	SD	2.1392E-03	1.5645E-02	1.7132E-02	**2.1109E-03**
P2: CONSTR	Ave	8.3098E-04	8.9220E-04	1.5711E-03	**7.2719E-04**
	SD	**3.3369E-05**	1.0565E-04	7.7806E-04	8.4147E-05
P3: DISK BRAKE	Ave	2.3045E-03	4.3311E-03	4.1887E-02	**2.1552E-03**
	SD	5.6273E-03	1.8074E-03	5.4951E-03	**3.8444E-04**
P4: 4-BAR TRUSS	Ave	1.4095E+01	1.1696E+01	2.5901E +00	**1.1373E +01**
	SD	**5.1580E-01**	2.3340E+00	2.2161E +00	8.6245E-01
P5: WELDED BEAM	Ave	1.1946E-02	3.9207E-02	**4.4399E-03**	5.3823E-03
	SD	1.9956E-03	3.8097E-02	6.9956E-04	**5.0405E-04**
P6: OSY	Ave	3.5765E+00	1.2736E+00	**7.0760E-01**	3.8851E +00
	SD	2.5250E+00	4.0179E-01	**2.4932E-01**	1.3209E +00
P7: SPEED REDUCER	Ave	8.2516E+01	8.2889E+00	7.6817E+00	**7.2212E+00**
	SD	9.8695E+01	1.6129E+00	3.4697E+00	**1.5340E+00**
P8: SRN	Ave	3.1617E-02	1.6743E-02	1.5798E-02	**1.0058E-02**
	SD	1.0695E-02	3.9924E-03	3.2892E-03	**1.1692E-03**

The bold number is the best result among other methods

Table 14 Statistical analysis of the engineering problems to determine IGD performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
P1: BNH	Ave	9.6868E-04	3.3817E-03	1.2198E-02	**7.0815E-04**
	SD	1.7147E-04	1.1513E-03	3.6455E-03	**4.3813E-05**
P2: CONSTR	Ave	5.1838E-04	7.4460E-04	2.5041E-03	**4.5869E-04**
	SD	5.5618E-05	1.7697E-04	9.6528E-04	**5.5228E-05**
P3: DISK BRAKE	Ave	**5.8831E-04**	7.2091E-04	1.7399E-03	8.1114E-04
	SD	**5.5995E-05**	1.1206E-04	7.7421E-04	7.2077E-05
P4: 4-BAR TRUSS	Ave	2.0010E-02	2.1409E-02	2.2004E-02	**2.0009E-02**
	SD	**3.9632E-05**	1.5445E-04	1.1024E-03	8.0745E-05
P5: WELDED BEAM	Ave	5.9705E-04	1.3441E-03	6.1770E-03	**5.7676E-04**
	SD	**4.6341E-05**	3.9377E-04	2.8159E-03	7.0072E-05
P6: OSY	Ave	1.4663E-02	7.6220E-03	8.1016E-03	**6.0964E-03**
	SD	8.6917E-03	6.4006E-04	2.2613E-04	**2.2429E-04**
P7: SPEED REDUCER	Ave	6.0305E-02	1.4243E-02	1.7737E-02	**1.0149E-02**
	SD	7.2130E-02	3.2032E-03	3.3878E-03	**3.2001E-03**
P8: SRN	Ave	4.5146E-04	2.4823E-03	6.2137E-03	**2.3639E-04**
	SD	1.1656E-04	1.0614E-03	1.8810E-03	**1.5491E-05**

The bold number is the best result among other methods

proposed MOCGO approaches are very close to the actual Pareto-optimal fronts with almost complete convergence. Moreover, the MOPSO, MOGWO, and MOALO demonstrate the poorest convergence. Finding optimal Pareto front values using the proposed method is preferable to alternative methods.

Table 15 Statistical analysis of the engineering problems to determine MS performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
P1: BNH	Ave	**1.0000E +00**	8.7156E-01	5.4090E-01	**1.0000E +00**
	SD	**0.0000E +00**	6.6957E-02	1.0692E-01	**0.0000E +00**
P2: CONSTR	Ave	9.9384E-01	9.7790E-01	7.7822E-01	**9.9470E-01**
	SD	6.8603E-03	2.0113E-02	7.6343E-02	**5.2161E-03**
P3: DISK BRAKE	Ave	9.9928E-01	9.9582E-01	9.1779E-01	**9.9933E-01**
	SD	1.1417E-03	1.2807E-02	2.0904E-01	**7.6139E-04**
P4: 4-BAR TRUSS	Ave	1.4876E+00	1.3787E+00	8.4793E-01	**1.4879E +00**
	SD	5.4502E-04	4.6120E-02	1.2454E-01	**2.3406E-16**
P5: WELDED BEAM	Ave	1.0072E+00	**1.1107E+00**	6.2463E-01	1.0000E+00
	SD	6.1053E-02	1.0194E-01	6.9354E-02	**6.0679E-02**
P6: OSY	Ave	3.2390E-01	6.6490E-01	5.7530E-01	**1.0671E +00**
	SD	3.4284E-01	**2.5118E-02**	2.6351E-02	2.8286E-01
P7: SPEED REDUCER	Ave	2.3456E-01	7.6707E-01	6.6868E-01	**8.1560E-01**
	SD	2.9012E-02	**2.4182E-02**	6.0939E-02	5.2331E-02
P8: SRN	Ave	9.0900E-01	7.0014E-01	3.9177E-01	**9.8067E-01**
	SD	4.5463E-02	8.0177E-02	8.2165E-02	**1.5671E-02**

The bold number is the best result among other methods

Table 16 Statistical analysis of the engineering problems to determine S performance

Functions		Algorithm			
		MOPSO	*MOGWO*	*MOALO*	*MOCGO*
P1: BNH	Ave	1.0901E+00	1.7477E+00	8.8402E-01	**8.9059E-01**
	SD	1.3631E-01	4.7352E-01	4.4648E-01	**8.2898E-02**
P2: CONSTR	Ave	5.8740E-02	**5.4894E-02**	6.9071E-02	6.8135E-02
	SD	**7.8936E-03**	8.3095E-03	1.8623E-02	9.7763E-03
P3: DISK BRAKE	Ave	1.1452E-01	1.3331E-01	1.4457E-01	**1.1247E-01**
	SD	1.3022E-02	1.9321E-02	1.1754E-02	**1.0146E-02**
P4: 4-BAR TRUSS	Ave	5.3605E +00	5.9045E+00	4.6988E+00	**3.3785E +00**
	SD	**2.6169E-01**	1.7462E+00	1.1721E+00	1.0577E +00
P5: WELDED BEAM	Ave	2.3432E-01	4.4716E-01	2.2431E-01	**1.9650E-01**
	SD	2.5702E-02	2.1585E-01	1.3595E-01	**2.0170E-02**
P6: OSY	Ave	**1.1278E +00**	1.6275E+00	1.4382E+00	7.2702E+00
	SD	1.4675E+00	**4.7312E-01**	4.8498E-01	2.4714E+00
P7: SPEED REDUCER	Ave	3.4532E+01	2.0624E+01	3.6722E+01	**2.0025E +01**
	SD	4.4378E+00	3.1861E+00	9.3623E +00	**2.7905E +00**
P8: SRN	Ave	2.2396E+00	2.6491E+00	1.7386E+00	**1.6190E +00**
	SD	4.7324E-01	1.2558E+00	8.6872E-01	**2.3299E-01**

The bold number is the best result among other methods

4.2.3.6 Comparison of MOCGO with MOCryStAl, MOHHO, MSSA The effectiveness of the proposed MOCGO is examined using additional optimization problems since the new multi-objective algorithms should be assessed using a few challenging real-world optimization problems. This subsection compares MOCGO with MOCryStAl [38], MOHHO [54], and MSSA [55] algorithms by the criterion of Ave and SDT for engineering problems. The outcomes of comparative approaches for GD, IGD, MS, and S are displayed in Tables 17, 18, 19, and 20. According to the

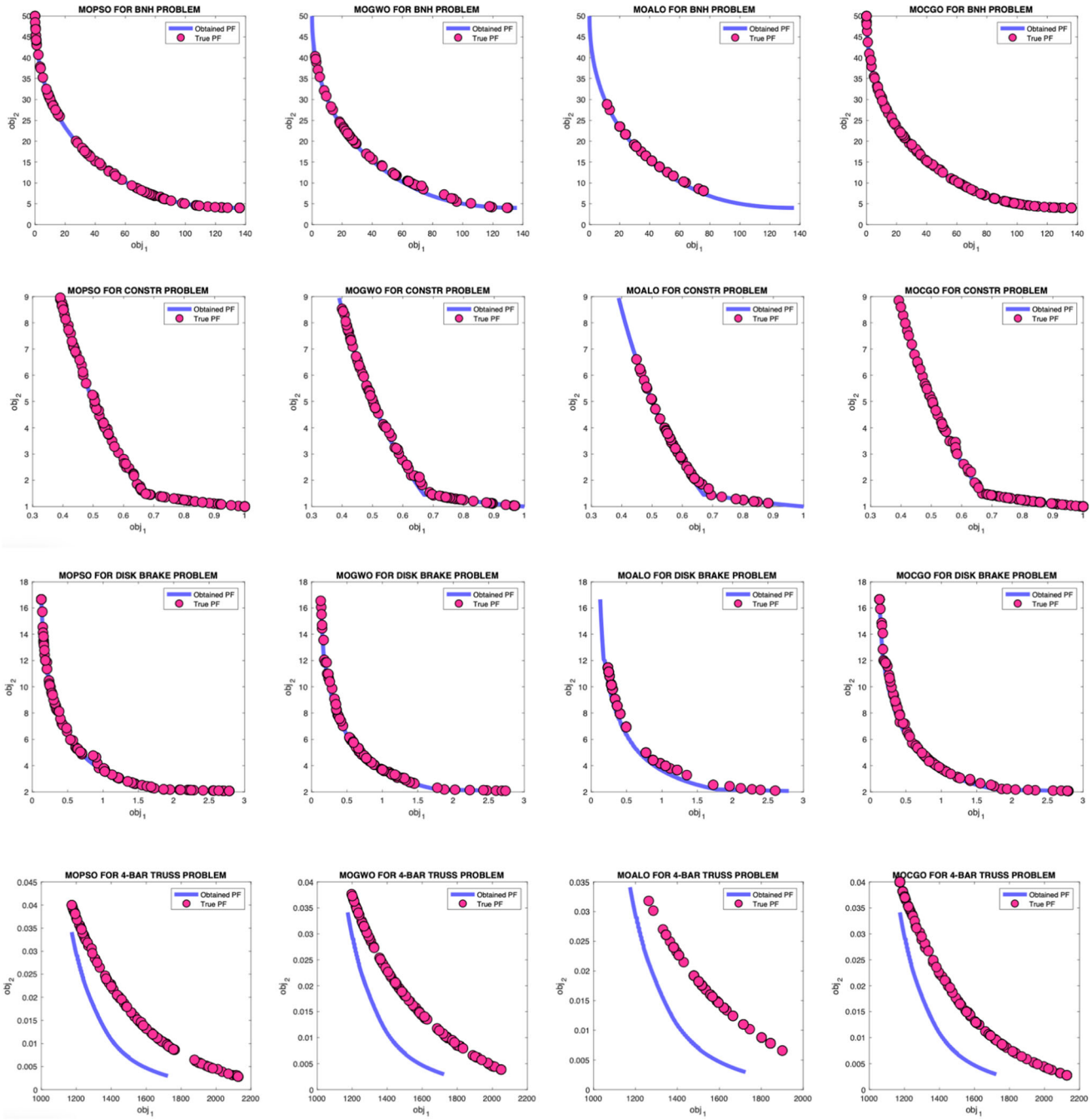

Fig. 13 Results of Pareto front results for BNH, CONSTR, DISK BRAKE, and 4-BAR TRUSS with MOCGO, MOPSO, MOALO, and MOGWO

GD measure in Table 17, the suggested strategy achieved encouraging outcomes in six out of all evaluated problems. In contrast MOCGO and MSSA obtained some of the best outcomes in terms of Ave in this table, whereas MOCryStAl and MOHHO did not. Table 18 presents the results that were obtained by making comparisons using the various techniques for each of the challenges that were investigated with regard to IGD. Regarding the Ave findings for the IGD measure, which are derived in Table 18, MOCGO has the capability to achieve acceptable results in any case. According to Table 19, the MOCryStAl and the MOHHO are only capable of offering the best results for

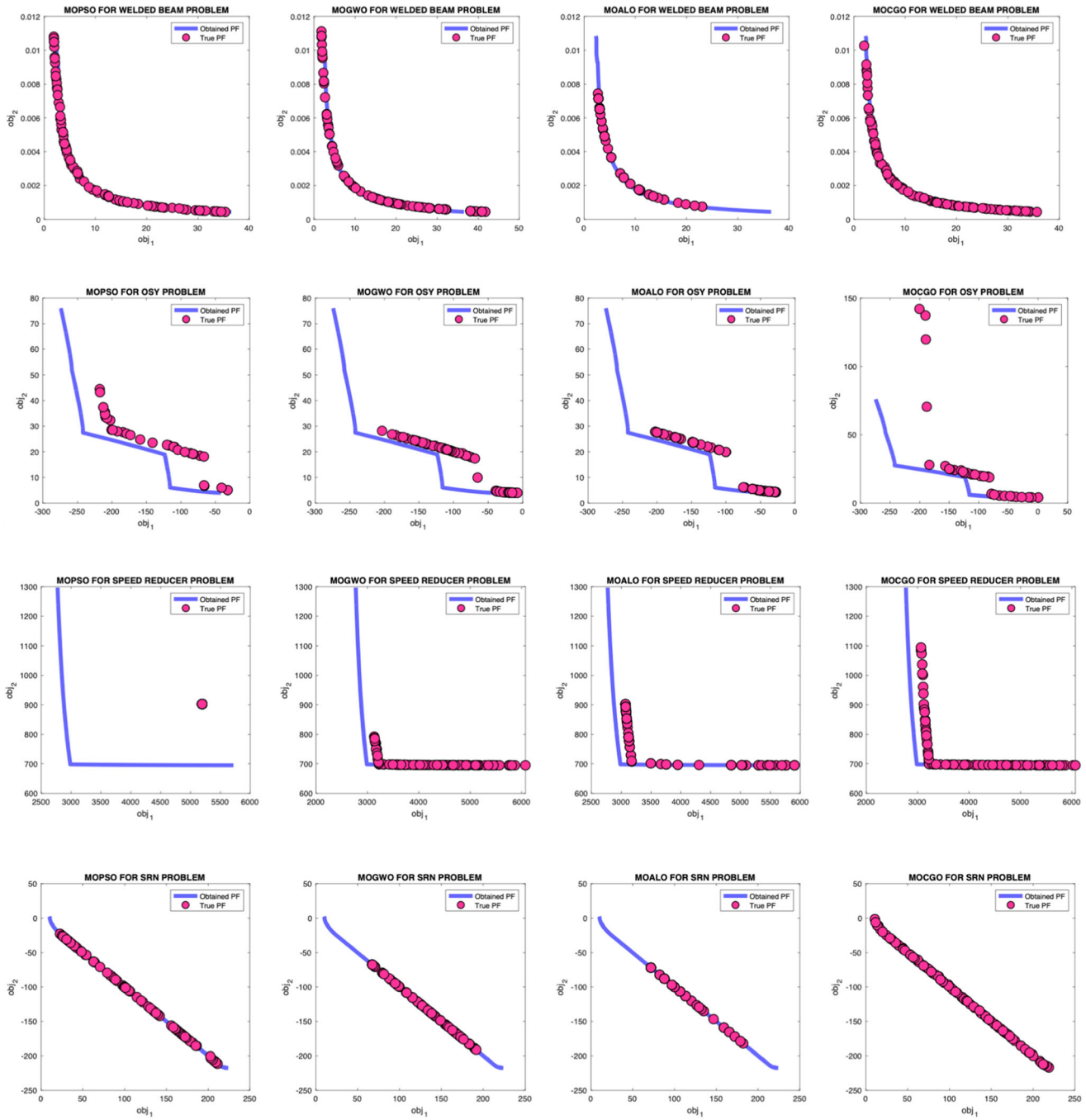

Fig. 14 Results of Pareto front results for WELDED BEAM, OSY, SPEED REDUCER, and SRN with MOCGO, MOPSO, MOALO, and MOGWO

one or two of the test problems that are taken into consideration when employing the MS metric to deal with engineering problems. In six of these problems, the suggested MOCGO is able to outperform the other approaches, proving its ability to handle this class of challenging issues.

Table 20 compares and summarizes the statistical outcomes of various multi-objective strategies together with the suggested algorithm. In four of the cases, it was found that MOCGO can outperform the other approaches in terms

Table 17 Statistical analysis of the engineering problems to determine GD performance

Functions		Algorithm			
		MOCryStAl	*MOHHO*	*MSSA*	*MOCGO*
P1: BNH	Ave	5.9356E-02	4.1654E-02	6.4575E-02	**3.2599E-02**
	SD	7.6599E-03	3.4567E-03	1.5035E-02	**2.1109E-03**
P2: CONSTR	Ave	1.5219E-03	1.4387E-03	1.6878E-03	**7.2719E-04**
	SD	5.9590E-04	1.5540E-04	5.8950E-04	**8.4147E-05**
P3: DISK BRAKE	Ave	7.5745E-03	9.1759E-03	6.2545E-03	**2.1552E-03**
	SD	1.6833E-03	1.4540E-02	3.3423E-03	**3.8444E-04**
P4: 4-BAR TRUSS	Ave	8.8970E+00	1.3565E +01	**7.7966E+00**	1.1373E+01
	SD	1.2099E+00	**7.3564E-01**	3.5486E +00	8.6245E-01
P5: WELDED BEAM	Ave	6.6659E-02	2.4582E-02	6.3467E-03	**5.3823E-03**
	SD	1.1624E-01	2.9564E-02	3.0983E-03	**5.0405E-04**
P6: OSY	Ave	6.7807E+00	3.9454E+00	**9.4637E-01**	3.8851E+00
	SD	3.4919E+00	2.5565E+00	**1.9612E-01**	1.3209E+00
P7: SPEED REDUCER	Ave	3.7694E+01	1.5398E+01	7.8657E+00	**7.2212E+0**
	SD	1.6132E+01	4.1560E+00	3.6189E+00	**1.5340E+0**
P8: SRN	Ave	1.1112E-01	2.8454E-02	4.1566E-02	**1.0058E-02**
	SD	4.7566E-02	1.3896E-02	2.4971E-02	**1.1692E-03**

The bold number is the best result among other methods

Table 18 Statistical analysis of the engineering problems to determine IGD performance

Functions		Algorithm			
		MOCryStAl	*MOHHO*	*MSSA*	*MOCGO*
P1: BNH	Ave	1.5496E-03	3.6978E-03	7.5631E-03	**7.0815E-04**
	SD	2.3504E-04	3.0065E-04	4.5344E-03	**4.3813E-05**
P2: CONSTR	Ave	1.1648E-03	1.1722E-03	2.1706E-03	**4.5869E-04**
	SD	2.6604E-04	2.3570E-04	7.3799E-04	**5.5228E-05**
P3: DISK BRAKE	Ave	9.6085E-04	1.6907E-03	2.5113E-03	**8.1114E-04**
	SD	2.1021E-04	2.4650E-04	1.1615E-03	**7.2077E-05**
P4: 4-BAR TRUSS	Ave	2.0095E-02	2.1202E-02	2.1434E-02	**2.0009E-02**
	SD	**4.5863E-05**	8.0770E-04	3.8292E-04	8.0745E-05
P5: WELDED BEAM	Ave	1.3180E-03	1.6897E-03	4.8242E-03	**5.7676E-04**
	SD	3.4394E-04	4.7980E-04	3.5852E-03	**7.0072E-05**
P6: OSY	Ave	9.2891E-03	1.3466E-02	7.7532E-03	**6.0964E-03**
	SD	2.4468E-03	5.5216E-03	1.2021E-03	**2.2429E-04**
P7: SPEED REDUCER	Ave	3.1325E-02	3.9332E-02	1.4933E-02	**1.0149E-02**
	SD	1.1989E-02	1.2667E-02	5.7981E-03	**3.2001E-03**
P8: SRN	Ave	3.3603E-04	1.4880E-03	2.2308E-03	**2.3639E-04**
	SD	6.6580E-05	2.4660E-04	1.5266E-03	**1.5491E-05**

The bold number is the best result among other methods

Table 19 Statistical analysis of the engineering problems to determine MS performance

Functions		Algorithm			
		MOCryStAl	*MOHHO*	*MSSA*	*MOCGO*
P1: BNH	Ave	9.9045E-01	9.8754E-01	7.6222E-01	**1.0000E +00**
	SD	2.0236E-02	1.1756E-02	1.3378E-01	**0.0000E +00**
P2: CONSTR	Ave	9.5474E-01	9.5076E-01	9.0536E-01	**9.9470E-01**
	SD	2.8125E-02	2.0265E-02	4.8380E-02	**5.2161E-03**
P3: DISK BRAKE	Ave	**1.0000E+00**	**1.0000E +00**	7.9510E-01	9.9933E-01
	SD	5.2642E-03	1.3966E-02	1.3013E-01	**7.6139E-04**
P4: 4-BAR TRUSS	Ave	1.4436E+00	1.4590E+00	1.2019E +00	**1.4879E+0**
	SD	8.7530E-02	2.9578E-02	1.8532E-01	**2.3406E-16**
P5: WELDED BEAM	Ave	1.0709E+00	**1.1190E+00**	7.9085E-01	1.0000E+00
	SD	1.2073E-01	1.1650E-01	1.4645E-01	**6.0679E-02**
P6: OSY	Ave	7.1700E-01	5.5698E-01	6.2048E-01	**1.0671E+00**
	SD	2.7281E-01	4.1566E-01	**8.0930E-02**	2.8286E-01
P7: SPEED REDUCER	Ave	3.8453E-01	3.8154E-01	7.2004E-01	**8.1560E-01**
	SD	1.6080E-01	2.1776E-01	6.9034E-02	**5.2331E-02**
P8: SRN	Ave	9.7667E-01	8.7678E-01	7.0583E-01	**9.8067E-01**
	SD	1.9187E-02	3.1567E-02	1.5109E-01	**1.5671E-02**

The bold number is the best result among other methods

Table 20 Statistical analysis of the engineering problems to determine S performance

Functions		Algorithm			
		MOCryStAl	*MOHHO*	*MSSA*	*MOCGO*
P1: BNH	Ave	1.4362E +00	8.9888E-01	1.2546E +00	**8.9059E-01**
	SD	4.3676E-01	4.0676E-01	3.9410E-01	**8.2898E-02**
P2: CONSTR	Ave	1.1085E-01	**4.2865E-02**	5.5908E-02	6.8135E-02
	SD	2.6254E-02	2.5756E-02	1.4886E-02	**9.7763E-03**
P3: DISK BRAKE	Ave	1.3453E-01	1.1676E-01	1.2768E-01	**1.1247E-01**
	SD	2.3710E-02	2.2658E-02	7.0776E-02	**1.0146E-02**
P4: 4-BAR TRUSS	Ave	1.1012E+01	5.9945E+00	6.1160E+00	**3.3785E+00**
	SD	4.6044E+00	1.7439E+00	1.6605E+00	**1.0577E+00**
P5: WELDED BEAM	Ave	3.6825E-01	2.3454E-01	**1.8912E-01**	1.9650E-01
	SD	1.6472E-01	7.0549E-02	8.7588E-02	**2.0170E-02**
P6: OSY	Ave	9.7125E+00	**9.1437E-01**	1.3502E+00	7.2702E +00
	SD	2.1108E+00	6.6901E-01	**6.2967E-01**	2.4714E +00
P7: SPEED REDUCER	Ave	8.9121E +01	3.1545E+01	**1.3952E +01**	2.0025E +01
	SD	7.6068E+01	3.5450E+00	9.8199E+00	**2.7905E +00**
P8: SRN	Ave	3.0058E+00	1.6546E+00	2.2721E+00	**1.6190E +00**
	SD	7.6260E-01	5.5434E-01	9.3573E-01	**2.3299E-01**

The bold number is the best result among other methods

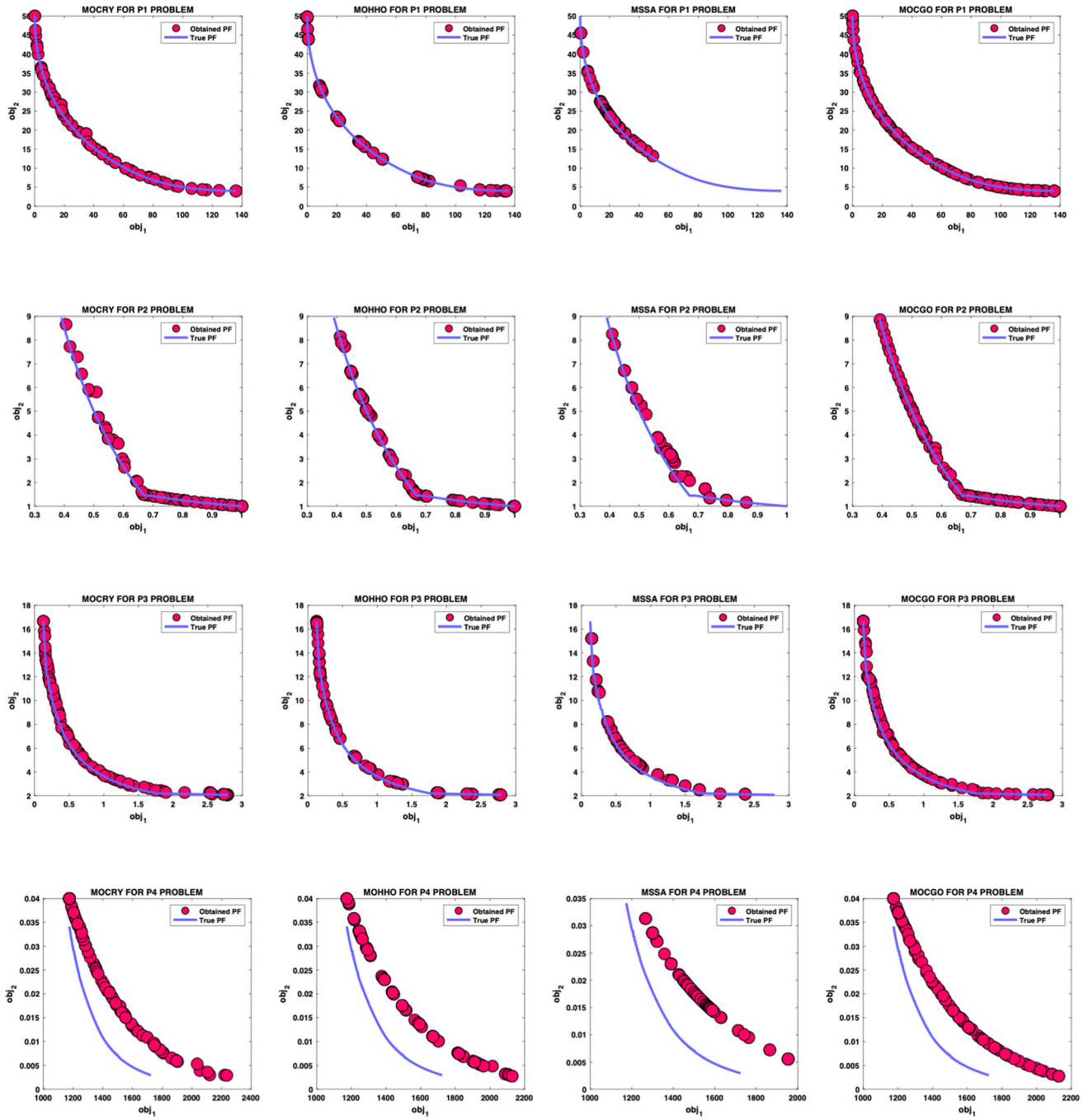

Fig. 15 Results of Pareto front results for BNH, CONSTR, DISK BRAKE, and 4-BAR TRUSS with MOCGO, MOCryStAl, MOHHO, and MSSA

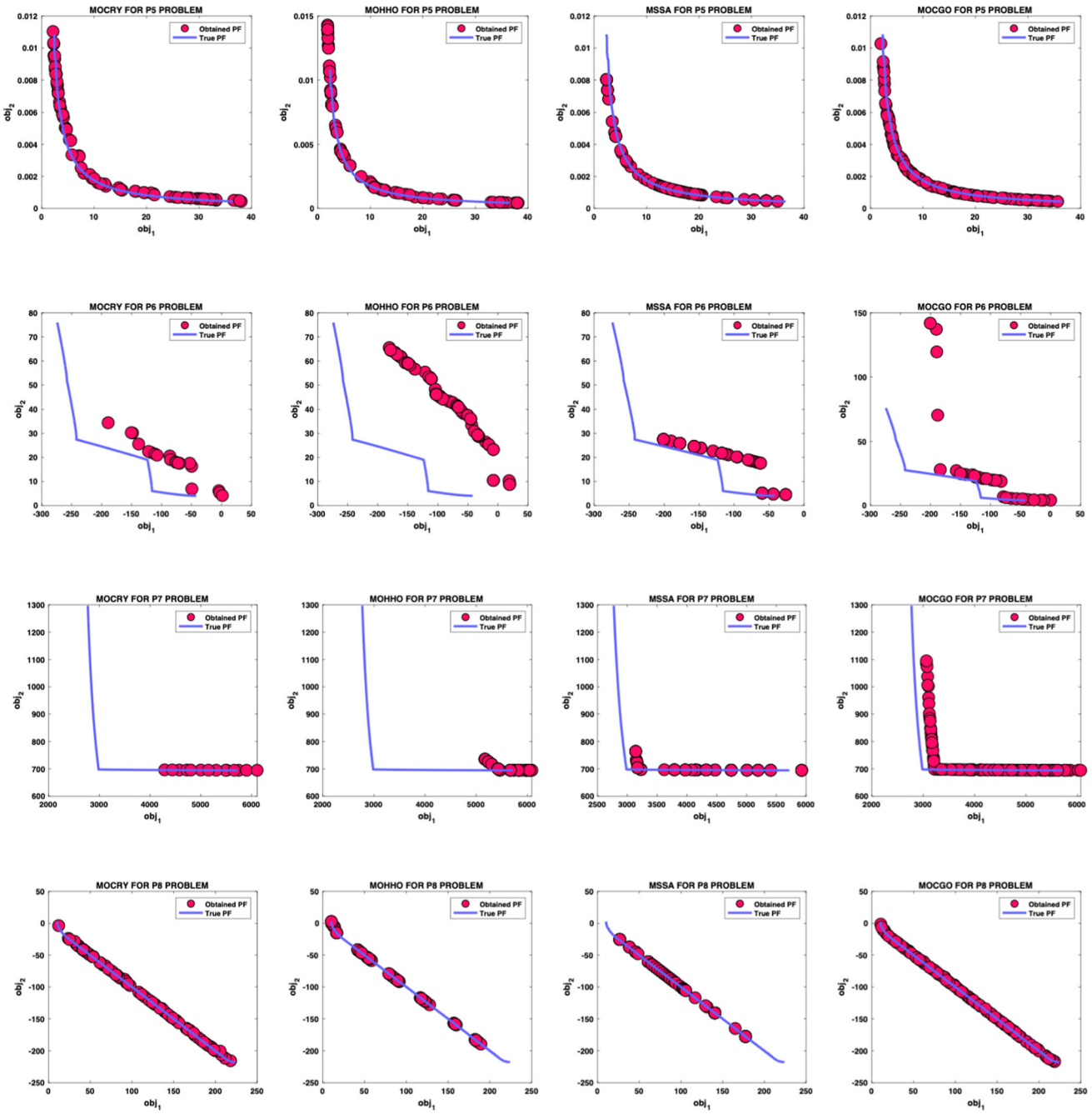

Fig. 16 Results of Pareto front results for WELDED BEAM, OSY, SPEED REDUCER, and SRN with MOCGO, MOCryStAl, MOHHO, and MSSA

of the S index, although the other approaches, like MSSA and MOHHO, also generate highly competitive results.

The results of the comparative methods on all engineering problems are shown in Figs. 15 and 16. These figures can be examined and it can be seen that MSSA and MOHHO exhibit the worst convergence while having strong coverage in CONSTR, DISK BRAKE and WELDED BEAM. The MOCryStAl and MOCGO, however, both offer excellent convergence toward all the real Pareto-optimal fronts.

5 Conclusion and future works

The multi-objective version of Chaos Game Optimization (CGO) as one of the newly suggested innovative metaheuristic algorithms is developed in this work. The CGO's inspiring concept is based on certain chaos theory concepts, in which the formation of fractals by the chaotic game concept and the fractal's self-similarity difficulties are considered. The proposed approach was compared to well-known algorithms such as MOPSO, MOGWO, MOALO, MOCryStAl, MOHHO, and MSSA for result confirmation. As a consequence, when compared to the previously described method, the results from this technique are quite competitive. The Completions on Evolutionary Computation (CEC-09) benchmark problems with some constrained mathematical (i.e., ZDT and DTLZ) are utilized for performance evaluation of multi-objective versions of CGO. Some real-world engineering design problems are tested to evaluate the MOCGO method's efficiency. The research shows that the proposed MOCGO can get higher rankings than competing methods when evaluating IGD, GD, and S indices and the MS index. Results showed that the proposed MOCGO technique could get one closer to the Pareto front in mathematical and engineering issues, which means better solutions.

In the future, the solution of multi-modal and nonlinear functionally demanding technical issues and engineering design obstacles, such as truss structures and the development of the structural health evaluation, may be used for the proposed MOCGO.

Appendix: Used in this work are constrained multi-objective test cases

CONSTR:

This issue has a convex Pareto front with two constraints and two design variables.

$$Minimize: f_1(x) = x_1 \tag{A.1}$$

$$\text{Minimize}: f_2(x) = (1 + x_2)/x_1 \tag{A.2}$$

$$where: g_1(x) = 6 - (x_2 + 9x_1) \tag{A.3}$$

$$g_2(x) = 1 + (x_2 - 9x_1) \tag{A.4}$$

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

SRN:

Srinivas and Deb [59] proposed a continuous Pareto-optimal front for the next challenge.

$$\text{Minimize}: f_1(x) = 2 + (x_1-2)^2+(x_2-1)^2 \tag{A.5}$$

$$\text{Minimize}: f_2(x) = 9x_1-(x_2-1)^2 \tag{A.6}$$

$$\text{where}: g_1(x) = x_1^2 + x_2^2 - 255 \tag{A.7}$$

$$g_2(x) = x_1 - 3x_2 + 10 \tag{A.8}$$

$$-20 \le x_1 \le 20, -20 \le x_2 \le 20$$

BNH:

Binh and Korn [60] were the first to suggest the following problem:

$$\text{Minimize}: f_1(x) = 4x_1^2 + 4x_2^2 \tag{A.9}$$

$$\text{Minimize}: f_2(x) = (x_1 - 5)^2 + (x_2 - 5)^2 \tag{A.10}$$

$$\text{where}: g_1(x) = (x_1 - 5)^2 + x_2^2 - 25 \tag{A.11}$$

$$g_2(x) = 7.7 - (x_1 - 8)^2 - (x_2 + 3)^2 \tag{A.12}$$

$$0 \le x_1 \le 5, 0 \le x_2 \le 3$$

OSY:

For the OSY test problem, Osyczka and Kundu [61] offered five separate zones. There are additional six limitations and six design variables to consider, as listed below:

$$\text{Minimize}: f_1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 \tag{A.13}$$

$$\text{Minimize}: f_2(x) = [25(x_1 - 2)^2 + (x_2 - 1)^2 + (x_3 - 1) + (x_4 - 4)^2 + \left(x_5 - 1\right)^2] \tag{A.14}$$

$$\text{Where: } g_1(x) = (2 - x_1 - x_2) \tag{A.15}$$

$$g_2(x) = -6 + x_1 + x_2 \tag{A.16}$$

$$g_3(x) = -2 - x_1 + x_2 \tag{A.17}$$

$$g_4(x) = -2 + x_1 - 3x_2 \quad \text{(A.18)}$$

$$g_5(x) = -4 + x_4 + (x_3 - 3)^2 \quad \text{(A.19)}$$

$$g_6(x) = 4 - x_6 - (x_5 - 3)^2 \quad \text{(A.20)}$$

$$0 \le x_1 \le 10, 0 \le x_2 \le 10, 1 \le x_3 \le 5 \quad \text{(A.21)}$$

$$0 \le x_4 \le 6, 1 \le x_5 \le 5, 0 \le x_6 \le 10$$

Funding Open access funding provided by Óbuda University.

Data availability Data will be available upon the request to the corresponding author.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4(2):65–85
2. J. Kennedy and R. Eberhart, (1995) Particle swarm optimization, In: Proceedings of ICNN'95-international conference on neural networks, vol 4, pp 1942–1948.
3. Dorigo M, Blum C (2005) Ant colony optimization theory: a survey. Theor Comput Sci 344(2–3):243–278
4. Kaveh A, Talatahari S, Khodadadi N (2020) Stochastic paint optimizer: theory and application in civil engineering. Eng Comput 38:1921–1952
5. Abdollahzadeh B, Gharehchopogh FS, Khodadadi N, Mirjalili S (2022) Mountain gazelle optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. Adv Eng Softw 174:103282
6. Khazalah A et al (2023) Image processing identification for sapodilla using convolution neural network (CNN) and transfer learning techniques. Classification Applications with Deep Learning and Machine Learning Technologies. Springer, Cham, pp 107–127
7. Abdelhamid AA et al (2022) Classification of monkeypox images based on transfer learning and the al-biruni earth radius optimization algorithm. Mathematics 10(19):3614
8. Gu Z-M, Wang G-G (2020) Improving NSGA-III algorithms with information feedback models for large-scale many-objective optimization. Futur Gener Comput Syst 107:49–69
9. J. D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, In: Proceedings of the first international conference on genetic algorithms and their applications, 1985, 1985.
10. Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731
11. Liu Q, Li X, Liu H, Guo Z (2020) Multi-objective metaheuristics for discrete optimization problems: a review of the state-of-the-art. Appl Soft Comput 93:106382
12. H. Ishibuchi, N. Tsukamoto, and Y. Nojima, Evolutionary many-objective optimization: a short review, In: 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence), 2008, pp 2419–2426
13. Murata T, Ishibuchi H (1995) MOGA: multi-objective genetic algorithms. IEEE Int Conf Evolut Comput 1:289–294
14. Luo J, Liu Q, Yang Y, Li X, Chen M, Cao W (2017) An artificial bee colony algorithm for multi-objective optimisation. Appl Soft Comput 50:235–251
15. Khodadadi N, Mirjalili SM, Zhao W, Zhang Z, Wang L, Mirjalili S (2023) Multi-objective artificial hummingbird algorithm. Advances in Swarm Intelligence. Springer, Cham, pp 407–419
16. Dhiman G et al (2021) MOSOA: a new multi-objective seagull optimization algorithm. Expert Syst Appl 167:114150
17. C. A. C. Coello and M. S. Lechuga, 2002 MOPSO: a proposal for multiple objective particle swarm optimization, In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), vol. 2, pp 1051–1056.
18. C.-W. Tsai, Y.-T. Huang, and M.-C. Chiang, 2014 A non-dominated sorting firefly algorithm for multi-objective optimization, In: 2014 14th International Conference on Intelligent Systems Design and Applications, pp 62–67.
19. Azizi M, Talatahari S, Khodadadi N, Sareh P (2022) Multi-objective atomic orbital search (MOAOS) for global and engineering design optimization. IEEE Access 10:67727–67746
20. N. Khodadadi, F. Soleimanian Gharehchopogh, and S. Mirjalili, 2022 MOAVOA: a new multi-objective artificial vultures optimization algorithm, Neural Comput. Appl, pp. 1–39
21. Das AK, Nikum AK, Krishnan SV, Pratihar DK (2020) Multi-objective Bonobo Optimizer (MOBO): an intelligent heuristic for multi-criteria optimization. Knowl Inf Syst 62(11):4407–4444
22. Khodadadi N, Abualigah L, Mirjalili S (2022) Multi-objective Stochastic Paint Optimizer (MOSPO). Neural Comput Appl 34(20):18035–18058
23. Khodadadi N, Mirjalili SM, Mirjalili S (2022) Multi-objective Moth-Flame Optimization Algorithm for Engineering Problems. Handbook of Moth-Flame Optimization Algorithm. CRC Press, Boca Raton, pp 79–96
24. N. Khodadadi, F. S. Gharehchopogh, B. Abdollahzadeh, and S. Mirjalili, 2022 AMHS: Archive-based multi-objective harmony search algorithm, In: Proceedings of 7th International Conference on Harmony Search, Soft Computing and Applications, pp 259–269.
25. Sharma S, Khodadadi N, Saha AK, Gharehchopogh FS, Mirjalili S (2022) Non-dominated sorting advanced butterfly optimization algorithm for multi-objective problems. J Bionic Eng 20(819):843
26. Talatahari S, Azizi M (2021) Chaos game optimization: a novel metaheuristic algorithm. Artif Intell Rev 54(2):917–1004
27. Cui Y, Geng Z, Zhu Q, Han Y (2017) Multi-objective optimization methods and application in energy saving. Energy 125:681–704
28. Arroyo JEC, dos Santos Ottoni R, de Paiva Oliveira A (2011) Multi-objective variable neighborhood search algorithms for a single machine scheduling problem with distinct due windows. Electron Notes Theor Comput Sci 281:5–19

29. Zhang W, Liu Y (2008) Multi-objective reactive power and voltage control based on fuzzy optimization strategy and fuzzy adaptive particle swarm. Int J Electr Power Energy Syst 30(9):525–532
30. Özkış A, Babalık A (2017) A novel metaheuristic for multi-objective optimization problems: The multi-objective vortex search algorithm. Inf Sci (Ny) 402:124–148
31. Melin P, Sánchez D (2018) Multi-objective optimization for modular granular neural networks applied to pattern recognition. Inf Sci (Ny) 460:594–610
32. Zhang H, Peng Y, Hou L, Tian G, Li Z (2019) A hybrid multi-objective optimization approach for energy-absorbing structures in train collisions. Inf Sci (Ny) 481:491–506
33. Du P, Wang J, Guo Z, Yang W (2017) Research and application of a novel hybrid forecasting system based on multi-objective optimization for wind speed forecasting. Energy Convers Manag 150:90–107
34. Liu R, Li J, Mu C, Jiao L (2017) A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization. Eur J Oper Res 261(3):1028–1051
35. Wang H et al (2018) A hybrid multi-objective firefly algorithm for big data optimization. Appl Soft Comput 69:806–815
36. Guo W, Chen M, Wang L, Wu Q (2017) Hyper multi-objective evolutionary algorithm for multi-objective optimization problems. Soft Comput 21(20):5883–5891
37. Sinan Hasanoglu M, Dolen M (2018) Multi-objective feasibility enhanced particle swarm optimization. Eng Optim 50(12):2013–2037
38. Khodadadi N, Azizi M, Talatahari S, Sareh P (2021) Multi-objective crystal structure algorithm (MOCryStAl): introduction and performance evaluation. IEEE Access 9:117795–117812
39. Pereira JLJ, Oliver GA, Francisco MB, Cunha SS Jr, Gomes GF (2022) Multi-objective lichtenberg algorithm: a hybrid physics-based meta-heuristic for solving engineering problems. Expert Syst Appl 187:115939
40. Khodadadi N, Talatahari S, Dadras Eslamlou A (2022) MOTEO: a novel multi-objective thermal exchange optimization algorithm for engineering problems. Soft Comput 26(14):6659–6684
41. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
42. Mirjalili S, Saremi S, Mirjalili SM, L dos S Coelho, (2016) Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. Expert Syst Appl 47:106–119
43. Srinivas N, Deb K (1994) Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol Comput 2(3):221–248
44. Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative case study. International conference on parallel problem solving from nature. Springer, Cham, pp 292–301
45. Knowles JD, Corne DW (2000) Approximating the nondominated front using the Pareto archived evolution strategy. Evol Comput 8(2):149–172
46. Khodadadi N, Abualigah L, El-Kenawy ESM, Snasel V, Mirjalili S (2022) An archive-based multi-objective arithmetic optimization algorithm for solving industrial engineering problems. IEEE Access 10:106673–106698
47. Nouhi B, Khodadadi N, Azizi M, Talatahari S, Gandomi AH (2022) Multi-objective material generation algorithm (MOMGA) for optimization purposes. IEEE Access 10:107095–107115
48. D. A. Van Veldhuizen and G. B. Lamont, (1998) Multiobjective evolutionary algorithm research: A history and analysis, Citeseer
49. J. R. Schott, (1995) Fault tolerant design using single and multicriteria genetic algorithm optimization. Massachusetts Institute of Technology
50. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195
51. C. M. Fonseca, J. D. Knowles, L. Thiele, and E. Zitzler, (2005) A tutorial on the performance assessment of stochastic multiobjective optimizers, In: Third international conference on evolutionary multi-criterion optimization (EMO 2005), vol. 216, pp 240.
52. Zapotecas-Martinez S, Garcia-Najera A, Lopez-Jaimes A (2019) Multi-objective grey wolf optimizer based on decomposition. Expert Syst Appl 120:357–371
53. Mirjalili S, Jangir P, Saremi S (2017) Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems. Appl Intell 46(1):79–95
54. Yüzgeç U, Kusoglu M (2020) Multi-objective harris hawks optimizer for multiobjective optimization problems. BSEU J Eng Res Technol 1(1):31–41
55. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
56. Coello CAC, Pulido GT (2005) Multiobjective structural optimization using a microgenetic algorithm. Struct Multidiscip Optim 30(5):388–403
57. Ray T, Liew KM (2002) A swarm metaphor for multiobjective design optimization. Eng Optim 34(2):141–153
58. Kurpati A, Azarm S, Wu J (2002) Constraint handling improvements for multiobjective genetic algorithms. Struct Multidiscip Optim 23(3):204–213
59. Srinivasan N, Deb K (1994) Multi-objective function optimisation using non-dominated sorting genetic algorithm. Evol Comp 2(3):221–248
60. T. T. Binh and U. Korn, (1997) "MOBES: A multiobjective evolution strategy for constrained optimization problems," In: The third international conference on genetic algorithms (Mendel 97), vol 25, pp 27.
61. Osyczka A, Kundu S (1995) A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. Struct Optim 10(2):94–99