Arabian Journal for Science and Engineering (2019) 44:7277–7296
https://doi.org/10.1007/s13369-019-03806-w

RESEARCH ARTICLE - SYSTEMS ENGINEERING

An Efficient Grey Wolf Optimizer with Opposition-Based Learning and Chaotic Local Search for Integer and Mixed-Integer Optimization Problems

Shubham Gupta[1] · **Kusum Deep**[1]

Received: 25 August 2018 / Accepted: 11 March 2019 / Published online: 3 April 2019

Abstract

Determining the global optima of integer and mixed-integer nonlinear problems is a useful contribution in various engineering applications. Swarm intelligence is a well-known branch of nature-inspired algorithms which tries to determine the solution with the help of intelligent and collective behaviour of social creatures. Grey wolf optimizer (GWO) is one of the recently developed efficient algorithms which are quite popular nowadays. In the present study, first, the GWO is proposed for solving integer and mixed-integer optimization problems, and secondly, an improved version of GWO named IMI-GWO is proposed. The IMI-GWO attempts to alleviate from the major issues of premature convergence and slow convergence of classical GWO. In IMI-GWO, the opposition-based learning maintains the diversity and the chaotic search locally exploits the regions around the best solutions. To evaluate the performance of IMI-GWO, a set of 16 integer and mixed-integer problems and two engineering application problems, namely gear train and pressure vessel design problems, have been considered. The performance of the IMI-GWO is compared with other algorithms which are applied to solve these problems in the literature and with some recent algorithms. The comparison illustrates the better performance of the proposed algorithm.

Keywords Swarm intelligence · Grey wolf optimizer · Integer and mixed-integer optimization problems · Opposition-based learning · Chaotic local search

1 Introduction

In engineering applications, many problems are formulated as mathematical optimization problems where the decision variables are of integer/mixed-integer type. These optimization problems are known as integer/mixed-integer nonlinear optimization problems. The general form of mixed-integer optimization problem can be stated as

$$\textit{\textbf{Max/Min}}\ \ \boldsymbol{f}(\boldsymbol{X}), \quad \boldsymbol{X} = (x_1, x_2, \ldots, x_d) \tag{1}$$

$$\textit{\textbf{s.t.}}\quad g_j(\boldsymbol{X}) \le 0 \quad j = 1, 2, \ldots, m \tag{2}$$

$$h_k(\boldsymbol{X}) = 0 \quad k = m+1, m+2, \ldots, m+p. \tag{3}$$

$$a_i \le x_i \le b_i \quad i = 1, 2, \ldots, d, \tag{4}$$

where $\boldsymbol{X}$ is d-dimensional decision vector whose components x_i $(i = 1, 2, \ldots, d)$ can be integer or non-integer, a_i and b_i are the lower and upper bounds for the ith component of a decision variable $\boldsymbol{X}$, g_j, $(j = 1, 2, \ldots, m)$ and h_k $(k = m+1, m+2, \ldots, m+p)$ are inequality and equality constraints, respectively, and f is an objective function. The functions g_j, h_k and f can be linear or nonlinear.

The integer and mixed-integer optimization problems are difficult to solve due to the nature of the decision variables. Gradient-based optimization approaches ensure that the optima is attained and can be used to solve various optimization problems blindly because of their theoretical evidence. These techniques are limited to tackle only some special formulation where mathematical structures such as convexity, continuity and differentiability are involved. To deal with the problems where these mathematical structures are absent, numerous optimization techniques based on stochastic search are used. These techniques are also known as nature-inspired techniques because they are developed by being inspired from natural phenomena. Genetic algorithm (GA) [1], differential evolution (DE) [2], particle swarm opti-

✉ Shubham Gupta
shubh.dma2015@iitr.ac.in

Kusum Deep
kusumfma@iitr.ac.in

1 Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee 247667, India

mization (PSO) [3], tabu search (TS) [4], artificial bee colony (ABC) algorithm [5], ant colony optimization (ACO) [6], etc. are some well-known nature-inspired optimization techniques. In the literature, these techniques have shown their dominant potential against deterministic optimization techniques to solve real-world nonlinear, non-convex and non-differentiable optimization problems. The nature-inspired algorithms are very popular nowadays because of their simplicity, easy implementation, derivative-free mechanism and flexibility. The flexibility of nature-inspired algorithms allows them to be implemented on any optimization problem without knowing any information regarding the problem. In the nature-inspired algorithms, two conflicting operators, namely exploration and exploitation, play an important role to proceed the search process towards promising regions [7]. The amount of balance between these operators decides the search efficiency of an algorithm.

The no free lunch (NFL) theorem [8], which was the revolutionary development in the field of nature-inspired algorithms, violates the existence of an algorithm which is suitable for all type of optimization problems. In this way, the theorem answers the facts of developing new algorithms and improvement in existing algorithms. There are many advantages and disadvantages for working with various nature-inspired algorithms such as low/high exploration, low/high exploitation and imperfect balance between exploitation and exploration, and therefore, various improved versions of existing algorithms have been developed to enhance the search-ability of existing algorithms. Since the NFL theorem opposes the existence of an ideal algorithm, the researchers always try to enhance the performance of existing algorithms so that they can be applied to solve wide range of optimization problems. The most common difficulties which are inherent in nature-inspired algorithms are stagnation in local optima, premature convergence, inefficient exploration and/or exploitation.

In the field of nature-inspired algorithms, swarm intelligence is a popular branch, where the individuals share useful information about the search space with other individuals in the swarm. The information sharing helps in exploring the search space and sometimes provides a sudden jump towards more promising search regions. Particle swarm optimization (PSO) [3], ant colony optimization (ACO) [6], artificial bee colony (ABC) algorithm [5], whale optimization algorithm (WOA) [9], spider monkey optimization (SMO) [10] and grey wolf optimizer (GWO) [11] are some algorithms in the field of swarm intelligence (SI)-based algorithms.

Nature-inspired algorithms have shown their potential to solve integer/mixed-integer problems [12–14]. In [15], simulated annealing (SA) [16] has been used to solve MINLP. In [17], Mohan and Nguyen proposed a random search technique incorporating with simulated annealing to solve MINLP. In [18], Deep et al. have solved MINLP using real-coded genetic algorithm hybridized with Laplace crossover.

The above works motivated the authors to develop a simple algorithm which is easy to implement and efficient in terms of accuracy for solving integer and mixed-integer constrained optimization problems. Since the recently developed algorithm GWO [11] is quite efficient in the literature, it has been applied to solve various real-life application problems. Combined heat and power economic dispatch problem has been solved using GWO in [19] and [20]. For flow shop scheduling problem, GWO is used in [21]. To train the q-Gaussian radial basis, an improved version of GWO has been proposed by Muangkote et al. [22]. In [23], the solution of non-convex economic load dispatch problem is attempted with GWO. To find the solution of optimal reactive power dispatch problem, Sulaiman et al. [24] have used GWO. Although GWO has been successfully applied to many real-world application problems but like other algorithms, it suffers the problem of stagnation at sub-optimal solutions. Therefore, to prevent such a situation, various attempts have been made by the researchers; for example, Mittal et al. [25] have proposed a modified version of GWO to main a suitable balance between exploration and exploitation in GWO. In [26], an improved version of GWO is proposed by giving weight to the leaders of the wolf pack. Gupta and Deep, in [27,28], proposed a random walk-based GWO which explores the more promising search region and provided a better guiding direction to find the solution of optimization problems. In [26], an improved GWO based on Levy flight is proposed to solve real-life problems. In [29], the search equation of GWO is modified to enhance the exploration.

Although in the literature, various improved versions of GWO are presented, still in some cases, GWO suffers the problem of stagnation at sub-optimal solutions and premature convergence [30,31]. The performance of GWO on multimodal and composite problems is evidence of such cases. Moreover, because of no free lunch theorem, there is always a scope for improving the algorithm so that they can be applied to a wide variety of optimization problems. Therefore, all the above facts have motivated us to propose an improved version of GWO which is more efficient to solve constrained and mixed-integer optimization problems with high accuracy. In this direction, in the present paper, constraint and discrete version of GWO called IMI-GWO is proposed. The unconstraint and continuous version of this algorithm is already proposed in [32] by the same authors. In the paper, the performance of classical GWO is also discussed on constrained integer and mixed-integer problems. To handle the constraints of the problems, a simple constraint handling technique based on the constraint violation is employed in algorithms, classical GWO and IMI-GWO. The discrete components of the decision variable are handled by truncation mechanism. The integer and mixed-integer optimization problems generally evaluate the enhanced explorative search-ability of search agents in an algorithm. In order to evaluate the performance of the proposed algorithm, a set of 16 integer and mixed-integer problems and 2 well-known engineering problems with discrete search space are considered. In the paper, constraint and discrete version of classical GWO is named as MI-GWO.

The remainder of the paper is organized as follows—Sect. 2 provides an overview of classical GWO algorithm. In Sect. 3, an improved version of grey wolf optimizer named as I-GWO and based on opposition-based learning and chaotic local search is proposed. In Sect. 4, an extended version, IMI-GWO of I-GWO for integer and mixed-integer optimization problems, has been proposed. In Sect. 5, the experimental results are presented and the comparison is made with variants of GWO, recent optimization algorithms and some other popular nature-inspired algorithms that are used to solve mixed-integer problems in the literature. In Sect. 6, gear train design and pressure vessel design problems are solved by using the proposed algorithm IMI-GWO and finally, Sect. 7 concludes the paper based on the work and suggests some future ideas.

2 Grey Wolf Optimizer (GWO)

Grey wolf optimizer (GWO) was developed in 2014 by Mirjalili et al. [11] from the inspiration of leadership hierarchy characteristic of the grey wolves. Grey wolves are considered as the apical predator, i.e. they occupy the highest level in the food chain. Grey wolves always hunt the prey in groups, their group is called pack and the size of the pack may vary from 5 to 11 wolves. In order to maintain the discipline, democracy and dictatorships in the pack, wolves categorize their group into two classes. In the first class, leading wolves (alpha, beta and delta) are included. Alpha is the dominant wolf which is responsible for all the important decisions within the pack. Beta can be considered as a subsidiary wolf to the alpha and conveys the crucial decisions to the other wolves. Delta wolves are the caretakers, sentinels of the pack. The remaining wolves belong to the second class and are known as omega wolves; they iteratively follow the leaders alpha, beta and delta to approach the prey. The leadership hierarchy of grey wolf pack is presented graphically in Fig. 1.

Muro et al. [33] have described the three steps which are followed by the grey wolves to kill the prey. In the first step, wolves track the prey, in the second step, they encircle the prey with the help of leaders, and in the last step, they attack prey to accomplish their hunting process. In the algorithm, the top three wolves having the best fitness value are called as alpha, beta and delta in order to simulate the leadership behaviour of a wolf pack. To incorporate the encircling behaviour of wolves in mathematical form, Mirjalili et al. [11] have proposed the following equations to update the wolf position

$$X_{t+1} = X_{P,t} - A \cdot D, \tag{5}$$

where t is iteration count, $X_{P,t}$ is the prey position in iteration t, X_{t+1} is the updated position of wolf position X_t, and A and D are the coefficient and difference vector which are defined as

$$A = 2 \cdot a \cdot r_1 - a \tag{6}$$

$$D = |C \cdot X_{P,t} - X_t|, \tag{7}$$

where

$$C = 2 \cdot r_2 \tag{8}$$

r_1 and r_2 are the uniformly distributed random vectors whose components lie between 0 and 1. The scalar a linearly decreases from 2 to 0 over iterations and can be formulated as

$$a = 2 - 2 \cdot \left(\frac{t}{T}\right); \tag{9}$$

here, T represents the total number of iterations which is predefined as the termination criteria for algorithm.

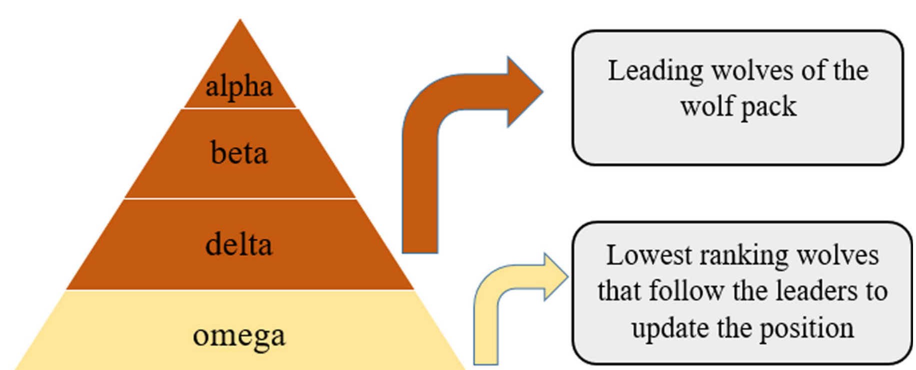

Fig. 1 Leadership hierarchy of grey wolves

In the algorithm, it was assumed that all the leaders alpha, beta and delta have enough information regarding the prey location. Therefore, all the leaders will contribute to the hunting process. Thus, in each iteration, each wolf update its state with the help of leaders by assuming them as a hypothetical prey for the current iteration. The equations that simulate the hunting process of wolves can be stated as

$$X_1 = X_{\alpha,t} - A_\alpha \cdot D_\alpha \tag{10}$$

$$X_2 = X_{\beta,t} - A_\beta \cdot D_\beta \tag{11}$$

$$X_3 = X_{\delta,t} - A_\delta \cdot D_\delta \tag{12}$$

$$X_{t+1} = \mathrm{avg}(X_1, X_2, X_3), \tag{13}$$

where $avg\ (X_1, X_2, X_3)$ denotes the average state of the positions that are obtained with the help of alpha, beta and delta.

In this way, on repeating the steps of encircling and hunting mathematically an optimum can be acquired for any optimization problem. The overall summary of grey wolf optimizer is described in Algorithm 1 [11].

```
Initialize the wolf population
Initialize the parameters a and maximum number of iterations T
Select the leading wolf alpha, beta and delta from the population according
to their objective function value
t =0 ( initialize the iteration count)
while t < T. Do
        update each wolf with the help of equations (5) − (13)
        update the leaders and parameter a
        t = t +1
end of while
```

Algorithm 1: Grey Wolf Optimizer algorithm

3 Proposed Improved Grey Wolf Optimizer (I-GWO)

Opposition-based learning (OBL), introduced by Tizhoosh [34], is an effective strategy to accelerate the convergence, by considering the estimate and its opposite estimate simultaneously of a current solution. It was proved mathematically that the opposite numbers are more likely to be closer to the optimal solution than the purely random ones [35]. In the present work, OBL has been employed in GWO to accelerate the convergence and to explore the promising regions which are unexplored in order to prevent GWO from the problem of premature convergence due to stagnation at local optima. The OBL generates opposite points which help in escaping from local optima. Also, to maintain a proper balance between exploration and exploitation, chaotic local search has been incorporated in GWO. Briefly, the concept of opposition-based learning and chaotic local search can be summarized as follows.

3.1 Opposition-Based Learning

Generally, nature-inspired algorithms start with a random population of solutions and try to find the optimal solution under the predefined termination criteria. The search process stops when the termination criteria is satisfied or the population of solutions is trapped in a local optimum. In opposition-based learning, the search in both directions, random and its opposite direction provides a higher chance to find the unknown optima of the problem. The OBL is very fruitful when the optimum positions are just in the opposite direction of a current solution. Therefore, in the present work, an OBL phase is integrated into GWO to move out from local optima when the stagnation occurs and to explore the more promising search regions of search space. The concept of opposite numbers and the OBL phase which is used in GWO is defined as follows.

3.1.1 Opposite Points

The concept of opposite numbers was first introduced in 2005 by Tizhoosh [34]. In this concept, the solution and its opposite solution both are considered to enhance the search process. In [34], it is suggested that the computational cost can also be reduced by applying the concept of opposite numbers. The opposite number can be defined as follows.

Let $x \in \boldsymbol{R}$ be a point lying in the interval $[a, b]$, then its opposite number $\hat{x}$ can be calculated as follows [34]

$$\hat{x} = a + b - x, \tag{14}$$

where a and b are lower and upper bounds for the variable x. The similar concept can be extend to the vectors of $\boldsymbol{R^d}$ in a following manner [34]:

Let $\boldsymbol{X} = (x_1, x_2, \ldots, x_d) \in \boldsymbol{R^d}$ be any point in a d-dimensional space, then its opposite point $\boldsymbol{X}^* = (x_1^*, x_2^*, \ldots, x_d^*)$ can be calculate as follows

$$x_i^* = a_i + b_i - x_i \tag{15}$$

for $i = 1, 2, \ldots, d$, where a_i and b_i are the lower and upper limit for a component x_i.

The OBL phase which is integrated in GWO is described in Algorithm 2.

```
for i = 1:n
    for j = 1:d
        op_ij = a_j + b_j − x_ij
    end
end
return OP = [op_ij]
```

Algorithm 2. OBL phase in I-GWO

3.2 Chaotic Local Search

In order to maintain a balance between exploration and exploitation in GWO, the present paper hybridizes GWO with greedy chaotic local search. A similar chaotic local search has been used in differential evolution [36]. The logistic chaotic function is used in the chaotic local search to generate a chaotic sequence [37] as:

$$\alpha_m = 4\,\alpha_{m-1}(1-\alpha_{m-1}) \text{ and } \alpha_0 = rand(0,1) \quad m = 1, 2, \ldots, N, \ldots, \tag{16}$$

where α_0, the uniformly distributed variable between 0 and 1 is the initial value of the chaotic system. The wolf generated by chaotic local search can be mathematically presented as follows

$$z = x_{rand} + \alpha_m \cdot (x_{pbest} - x_{rand}), \tag{17}$$

where x_{rand} is a randomly chosen wolf to perform a chaotic search and x_{pbest} is the best wolf from the top $100 \cdot r$ population of wolves with $r = rand(0.1, 2/n)$. The random number r decides the percentage of the population which participates in chaotic local search phase. In the chaotic local search, this random number r helps in exploiting the promising regions around the top $[10, 200/n]$ fitted wolves of a pack. For example, if population size is 30, then 6 to 10 top fitted wolves are used in chaotic phase. The reason behind taking some set of top fitted wolves in place of a single fitted wolf is to prevent the wolf pack from the stagnation at local optimums and to discover the promising regions of a search space around the neighbourhood of top fitted wolf of the pack. The proposed greedy chaotic local search is helpful when the fittest solution of the pack traps in local optima and in that case other solutions selected for chaotic search provide a direction for the search process. In the algorithm, the generation of a better solution in a chaotic search is attempted up to N times. In the chaotic phase, the value of N will be less when the dimension is low. The reason behind selecting such value of N is to occasionally generate the better solutions around the current solution which is selected in the chaotic search. We do not want to generate a solution in each generation by chaotic local search because it has been used in the algorithm to prevent from stagnation at local optimums. The chaotic local search strategy used in GWO is presented in Algorithm 3.

$m = 0$
$N = d/5$
randomly select a wolf x_{rand} *from the wolf population*
$s^0 = rand(0,1)$
$r = rand(0.1, 2/n)$
while $m < N$
    randomly select a wolf x_{pbest} *from the top* $100 \cdot r$ *wolves of pack*
    for $j = 1$ *to* d
        $z_j = x_{rand,j} + \alpha_m \cdot (x_{pbest,j} - x_{rand,j})$
        if $z_j < a_j$
            set $z_j = a_j$
        end if
        if $z_j > b_j$
            set $z_j = b_j$
        end if
    end for
        if $f(z) \le f(x_{rand})$
            $x_{rand} = z$
            break
        end if
$m = m + 1$
$\alpha_m = 4\ \alpha_{m-1}(1 - \alpha_{m-1})$
end while

Algorithm 3. Chaotic local search phase used in GWO

Thus, the OBL and chaotic local search have been incorporated in GWO to enhance the performance of original grey wolf optimizer (GWO) in terms of exploring the new promising regions and balancing the operators' exploration and exploitation. The framework of the modified version of GWO, named as I-GWO in this paper, is presented in Algorithm 4. In Algorithm 3, opposition probability (p) decides whether OBL phase is introduced in I-GWO.

Initialize *the population of grey wolves* (P)
Evaluate *the fitness of each wolf*
Select *the leaders alpha, beta and delta for the search process*
Initialize *the algorithm parameters a and maximum number of iterations* T
Initialize *the iteration count* $t = 0$ *and opposition probability* (p)
While $t < T$
    Generate a uniformly distributed random number $y = rand\ ()$ *between* 0 *and* 1
    If $y \le p$
        apply OBL phase to generate a opposite wolf population (OP)
        select the top N wolves as a population from the set $P \cup OP$
    end if
    If $y > p$
        for *each wolf*
            update *the position of each wolf using equations* (5) − (13)
        end *for*
    end if
evaluate the fitness of all wolves
apply the chaotic local search
update the leaders alpha, beta and delta of wolf pack
$\boldsymbol{t = t + 1}$
end *while*
return the leader alpha

Algorithm 4. Algorithm of I-GWO

4 Extended GWO for Mixed-Integer Optimization

In the present paper, to handle the constraints of integer and mixed-integer constrained nonlinear optimization problems a very simple constraint handling technique (based on the constraint violation) is employed which can be summarized as follows

1. For each wolf x, evaluate the constraint violation [38] c_viol_x as follows

$$c\_viol_x = \sum_{j=1}^{m} G_i(x) + \sum_{k=m+1}^{p} H_j(x), \tag{18}$$

where

$$G_i(x) = \begin{cases} g_i(x) & if\ g_i(x) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

$$H_j(x) = \begin{cases} |h_j(x)| & if\ |h_j(x)| - \in > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where $\in$ is a predefined tolerance parameter, which is fixed to be 10^{-4} in the present paper.

2. Sort the population in an ascending order of constraint violation. Furthermore, sort the feasible wolves in increasing order of objective function value (for minimization problems).
3. Now select top three wolves as the leaders for the wolf pack.

The above-described constraint handling technique is very easy to incorporate in any search algorithm and it is just an indirect form of constraint handling technique proposed by Deb based on some feasibility rules [38] where each individual solution is compared with remaining solutions to select the best one.

Moreover, to deal with the integral component of a decision vector in integer and mixed-integer optimization problems, truncation procedure is employed in an algorithm which is described as follows

Let x_i be the ith component of a decision vector $\boldsymbol{X}$. Then it can be truncated to obtain a new component x_i' having integer value as for

$$\theta = x_i - [x_i] \tag{21}$$

$$x_i' = \begin{cases} [x_i] + 1, & \theta \geq 0.5, \\ [x_i], & \text{otherwise,} \end{cases} \tag{22}$$

where $[x_i]$ is the greatest integer value of x_i less than x_i. Obviously for a integer component $\theta = 0$.

In the present paper, the extended versions of GWO and I-GWO with the incorporation of truncation strategy and constraint handling technique are named as MI-GWO and IMI-GWO, respectively.

5 Experimental Results and Analysis

In this section, the proposed algorithm (IMI-GWO) is tested on 16 integer and mixed-integer optimization problems. These problems have been taken from different sources of the literature and are reported in "Appendix". This problem set consists of integer and mixed-integer optimization problems. All these problems together have been solved in [18]. In the present paper, these problems have been solved by MI-GWO, IMI-GWO, variants of GWO, recent algorithms such as SCA, MFO, SSA and other algorithms which are used to solve the same set of problems in [18]. For a fair comparison, same population size ($10 \times$ *dimension of the problem*) for the test problem P1–P15 and ($3 \times$ *dimension of the problem*) for the test problem P16 has been taken.

In total, 100 runs of various algorithms under consideration in this study have been conducted on each problem. A particular run is considered as a successful run if the error value in objective function value (*error* = |*known objective function value-obtained objective function value*|) is less than 0.01. This threshold is taken same as in [18]. All the experiments have been conducted on MATLAB 2014a with 4GB RAM and i5 processor system.

In Table 1, the performance of the proposed algorithm is measured using some statistical metrics—best, median, average, worst and standard deviation of objective function values obtained in 100 runs. In this table, the results of the proposed IMI-GWO algorithm has been compared with MI-GWO, variants of GWO (modified GWO [25], fitness GWO [26] and weighted GWO [26]) and some recent nature-inspired algorithms such as sine–cosine algorithm (SCA) [39], moth-flame optimization (MFO) algorithm [40] and salp swarm algorithm (SSA) [41] based on the various statistical measures. The comparison in the table has been made by considering the same number of function evaluations. In Table 2, the comparison between all these algorithms is made based on the number of success obtained in 100 runs. In this table, the number of function evaluations used to solve the problems and optima of the problems are also presented.

In Table 4, the performance of proposed algorithm (IMI-GWO) is compared with AXNUM [13] which is a real-coded genetic algorithm using non-uniform mutation, MI-LXPM [18] which is also a real-coded genetic algorithm using Laplace crossover and power mutation, and RST2ANU [17] which is a random search technique using simulated annealing. These algorithms are selected for comparison because, in the literature, these algorithms are used to solve the same set of problems. Therefore, for a fair comparison, the results are reported from [18]. In this table, the number of successes obtained in 100 runs and average function evaluations used by algorithms are reported. To compare the results, the parameter setting of population size and termination criteria is taken the same in IMI-GWO and MI-GWO as [18].

5.1 Analysis and Discussion of Results

Tables 1 and 2 show that the proposed algorithm (IMI-GWO) performs better as compared to other algorithms. Various sta-

Table 1 Best, median, average, worst and standard deviation of objective function values obtained in 100 runs by proposed algorithm IMI-GWO, MI-GWO and other variants of GWO

Problems	Algorithms	Best	Median	Average	Worst	STD
P1	MI-GWO	2	2.0002	2.10172	2.2379	0.11757
	IMI-GWO	2	2.0001	2.00024	2.0070	0.00072
	mGWO	2	2.0006	2.10186	2.2382	0.11756
	Fitness GWO	2	2.0004	2.10884	2.2382	0.11836
	Weighted GWO	2	2.0004	2.101768	2.2374	0.11757
	SCA	2	2.003	2.078054	2.2441	0.11089
	MFO	2	2	2.021249	2.2361	0.06791
	SSA	2	2	2.080391	2.2363	0.112332
P2	MI-GWO	2.124	2.126	2.133	2.497	0.04688
	IMI-GWO	2.124	2.125	2.126	2.131	0.00145
	mGWO	2.124	2.127	2.134	2.497	0.04673
	Fitness GWO	2.125	2.126	2.126	2.132	0.00171
	Weighted GWO	2.125	2.126	2.133	2.497	0.04693
	SCA	2.124	2.131	2.134	2.161	0.00838
	MFO	2.125	2.125	2.140	2.558	0.07417
	SSA	2.1245	2.14065	2.15154	2.2694	0.032628
P3	MI-GWO	1.0771	1.25	1.18215	1.2500	0.08215
	IMI-GWO	1.0767	1.0793	1.07953	1.0830	0.00139
	mGWO	1.0770	1.0864	1.07647	1.2500	0.19290
	Fitness GWO	1.0767	1.0802	1.07361	1.2500	0.19279
	Weighted GWO	1.0772	1.0800	1.06673	1.2500	0.18958
	SCA	1.1534	1.2500	1.24820	1.2500	0.01272
	MFO	1.0765	1.2500	1.16537	1.2500	0.08679
	SSA	1.0858	1.25	1.248358	1.25	0.01642
P4	MI-GWO	−6924.32	−6812.77	−6802.78	−6613.11	70.50406
	IMI-GWO	−6944.91	−6861.52	−6838.72	−5876.2	124.60240
	mGWO	−6937.21	−6752.60	−6738.32	−6279.88	128.97570
	Fitness GWO	−6950.46	−6859.83	−6841.48	−6674.14	66.65962
	Weighted GWO	−6937.76	−6814.16	−6801.53	−6511.85	90.13805
	SCA	−6394.02	−6162.97	−6072.82	−5713.53	262.17450
	MFO	−6938.93	−6220.97	−6032.55	−1471.14	989.71420
	SSA	−6950.76	−4410.49	−4405.49	−1236.8985	1798.513
P5	MI-GWO	−68	−68	−67.68	−36	3.2
	IMI-GWO	−68	−68	−68	−68	0
	mGWO	−68	−68	−67.68	−36	3.2
	Fitness GWO	−68	−68	−68	−68	0
	Weighted GWO	−68	−68	−58.07	0	21.82
	SCA	−68	−68	−60.72	−8	14.69
	MFO	−68	−68	−67.28	−56	2.86
	SSA	−68	−42	−42.69	−12	14.16119
P6	MI-GWO	−6	−6	−6	−6	0
	IMI-GWO	−6	−6	−6	−6	0
	mGWO	−6	−6	−6	−6	0
	Fitness GWO	−6	−6	−6	−6	0
	Weighted GWO	−6	−6	−5.97	−3	0.3
	SCA	−6	−6	−6	−6	0
	MFO	−6	−6	−5.97	−3	0.3
	SSA	−6	−6	−6	−6	0

Table 1 continued

Problems	Algorithms	Best	Median	Average	Worst	STD
P7	MI-GWO	99.2452	99.2452	99.2452	99.2452	100
	IMI-GWO	99.2452	99.2452	99.2452	99.2452	100
	mGWO	99.2452	99.2452	99.2452	99.2452	100
	Fitness GWO	99.2452	99.2452	99.2452	99.2452	100
	Weighted GWO	99.2452	99.2452	99.2452	99.2452	100
	SCA	99.2452	99.2452	99.2452	99.2452	100
	MFO	99.2452	99.2452	99.2452	99.2452	100
	SSA	99.2452	99.2452	99.2452	99.2452	1.43E−14
P8	MI-GWO	3.5575	4.2508	4.1674	4.9502	0.57079
	IMI-GWO	3.5575	3.5576	3.5705	4.8159	0.12582
	mGWO	3.5578	3.5578	3.6032	4.9501	0.33899
	Fitness GWO	3.5576	3.5576	3.6092	4.5577	0.24658
	Weighted GWO	3.5575	3.5576	3.6188	4.9501	0.29299
	SCA	3.5590	4.5854	4.4084	5.5908	0.50736
	MFO	3.5575	3.5590	3.6528	4.6369	0.24699
	SSA	3.5575	3.55815	3.591677	4.6328	0.154735
P9	MI-GWO	−32217.42	−32217.42	−32217.42	−32217.42	0
	IMI-GWO	−32217.42	−32217.42	−32217.42	−32217.42	0
	mGWO	−32217.40	−32217.40	−32217.40	−32217.40	3.29E−11
	Fitness GWO	−32217.40	−32217.40	−32217.40	−32217.40	3.29E−11
	Weighted GWO	−32217.40	−32217.40	−32217.40	−32217.40	3.29E−11
	SCA	−32217.42	−32217.42	−32217.42	−32217.40	3.29E−11
	MFO	−32030.90	−31042.30	−30965.60	−29306.50	584.59
	SSA	−32033.8	−31080.7	−30960.3	−29256.673	552.8311
P10	MI-GWO	−0.94347	−0.94347	−0.94224	−0.91896	0.00432
	IMI-GWO	−0.94347	−0.94347	−0.94311	−0.93634	0.00129
	mGWO	−0.94347	−0.94347	−0.94308	−0.93555	0.004101
	Fitness GWO	−0.94347	−0.94347	−0.94310	−0.93555	0.004906
	Weighted GWO	−0.94347	−0.94347	−0.94308	−0.93175	0.004297
	SCA	−0.94347	−0.94347	−0.94314	−0.93164	0.008678
	MFO	−0.94347	−0.94347	−0.94310	−0.94115	0.001945
	SSA	−0.9532	−0.9508	−0.94794	−0.91664	0.0077
P11	MI-GWO	8	8	8.58	18	1.90
	IMI-GWO	8	8	8	8	0
	mGWO	8	8	8.6	14	1.81
	Fitness GWO	8	8	8.78	14	2.03
	Weighted GWO	8	8	8.48	14	1.63
	SCA	8	14	12.64	18	3.96
	MFO	8	8	8.47	14	1.51
	SSA	8	13	11.76	18	3.429286
P12	MI-GWO	14	14	14.44	16	0.83
	IMI-GWO	14	14	14.38	16	0.79
	mGWO	14	14	14.50	16	0.87
	Fitness GWO	14	14	14.66	16	0.95
	Weighted GWO	14	14	14.28	16	0.70
	SCA	14	14	14.22	16	0.63
	MFO	14	14	14.16	16	0.51
	SSA	14	14	14.3	16	0.689019

Table 1 continued

Problems	Algorithms	Best	Median	Average	Worst	STD
P13	MI-GWO	−42.6321	−42.6321	−42.6321	−42.6321	0
	IMI-GWO	−42.6321	−42.6321	−42.6321	−42.6321	0
	mGWO	−42.6321	−42.6321	−42.6321	−42.6321	8.57E−14
	Fitness GWO	−42.6321	−42.6321	−42.6321	−42.6321	8.57E−14
	Weighted GWO	−42.6321	−42.6321	−42.6321	−42.6321	8.57E−14
	SCA	−42.6321	−42.6321	−42.247409	−23	2.11198
	MFO	−42.6321	−42.6321	−40.3834	−23	5.34957
	SSA	−42.6321	−42.6321	−40.3834	−23	5.349567
P14	MI-GWO	5.83E−07	5.50E−04	5.50E−04	2.64E−03	5.30E−04
	IMI-GWO	1.14E−06	5.30E−04	6.60E−04	2.71E−03	5.70E−04
	mGWO	8.29E−07	5.83E−04	7.99E−04	5.86E−03	9.14E−04
	Fitness GWO	9.46E−07	5.17E−04	6.65E−04	2.13E−03	5.38E−04
	Weighted GWO	2.63E−06	5.25E−04	6.85E−04	3.48E−03	6.15E−04
	SCA	1.07E−05	1.31E−03	2.48E−03	1.40E−02	3.31E−03
	MFO	1.61E−05	5.32E−03	8.06E−03	2.85E−02	7.83E−03
	SSA	1.46E−05	7.21E−03	9.20E−03	2.85E−02	8.86E−03
P15	MI-GWO	807	807	830.70	947	43.97899
	IMI-GWO	807	807	807	807	0
	mGWO	807	807	832.10	1032	47.99505
	Fitness GWO	807	807	842.49	1361	73.93602
	Weighted GWO	807	807	832.75	1062	50.71895
	SCA	892	1361	1504.92	4863	772.91030
	MFO	807	815	827.14	1032	38.17266
	SSA	807	807	808.06	821	2.469286
P16	MI-GWO	−0.99995	−0.99927	−0.99893	−0.99593	0.00103
	IMI-GWO	−0.99995	−0.99990	−0.99989	−0.99938	0.00101
	mGWO	−0.99987	−0.99713	−0.99573	−0.979	0.00405
	Fitness GWO	−0.99995	−0.9989	−0.9983	−0.9899	0.0021
	Weighted GWO	−0.99995	−0.9984	−0.9979	−0.9895	0.0021
	SCA	−0.99995	−0.97565	−0.96774	−0.85535	0.029733
	MFO	−0.99995	−0.9985	−0.9989	−0.9897	0.00368
	SSA	−0.99995	−0.97589	−0.96863	−0.85785	0.009853

tistical measures which are presented in Table 1 demonstrate the competitive ability of the proposed algorithm than other algorithms. The success rates of variants of GWO are as follows: mGWO has achieved 100% success in four problems, fitness GWO has achieved 100% success in seven problems and weighted GWO has achieved 100% success in five problems, while the proposed IMI-GWO has achieved 100% fitness in thirteen problems. Therefore, in terms of the success rate and accuracy in obtaining the solutions, the proposed algorithm (IMI-GWO) outperforms its variants and other recent algorithms. In order to prove that the better results are not being obtained just by chance, statistical analysis of the results is very essential [42,43]. Therefore, in the present work, to verify the significant improvement in the proposed algorithm, a nonparametric Wilcoxon rank-sum test [43] is employed on the obtained results through various algorithms. The obtained $p-$values and the statistical decisions are presented in Table 3. Table 3 shows that the IMI-GWO is significantly better than MI-GWO (classical version of GWO) in the problems F1 to F4, F8, F10 to F12, F15 and F16, while in the other problems both the algorithms are statically equivalent. Similarly, the IMI-GWO either significantly beats or performs equal to the mGWO, fitness GWO and weighted GWO. In the problems F7, F9 and F13, the classical GWO (MI-GWO) and variants of GWO such as mGWO, fitness GWO and weighted GWO provide the similar and optimum results; therefore, the statistical comparison shows the equivalent performance. The statistical comparison of the IMI-GWO algorithm with recent algorithms such as SCA, SSA and MFO also demonstrates the better ability

Table 2 Success and average function evaluations (f_E) used by proposed IMI-GWO, variants of GWO and some recent optimization approaches

Problem	Optima	f_E	MI-GWO	mGWO	Fitness GWO	Weighted GWO	SCA	MFO	SSA	IMI-GWO
P1	2	1300	57	57	54	57	66	91	66	100
P2	2.124	320	98	95	100	98	58	89	44	100
P3	1.07654	9000	38	24	32	37	0	48	1	100
P4	−6961.8138	10,000	0	0	0	0	0	0	0	0
P5	−68	630	99	99	100	80	79	94	8	100
P6	−6	160	100	100	100	99	100	99	100	100
P7	99.2452	1200	100	100	100	100	100	100	100	100
P8	3.5575	2×10^5	40	72	72	73	6	63	86	98
P9	−32217.42	50	100	100	100	100	100	0	0	100
P10	−0.94347	1760	97	96	96	96	96	100	95	100
P11	8	3250	91	90	87	92	24	91	40	100
P12	14	39,000	78	75	67	86	84	90	83	81
P13	−42.632	40	100	100	100	100	89	74	74	100
P14	0	90	100	100	100	100	95	66	62	100
P15	807	50,000	76	76	57	75	0	34	68	100
P16	−0.99995	170	100	89	99	99	23	100	100	100

Table 3 Comparison of statistical results obtained from Wilcoxon rank-sum test

Problem	MI-GWO	mGWO	Fitness GWO	Weighted GWO	SCA	MFO	SSA
P1	2.36E−27(+)	2.29E−32 (+)	7.65E−31(+)	2.48E−29(+)	2.14E−38(+)	9.12E−21(+)	9.12E−15(+)
P2	5.59E−38(+)	5.59E−39(+)	5.56E−39(+)	5.52E−39 (+)	5.64E−39 (+)	5.64E−40 (+)	5.64E−39 (+)
P3	4.08E−11(+)	9.21E−06 (+)	3.31E−01 (=)	1.69E−01(=)	2.06E−09(+)	3.29E−09(+)	8.02E−39(+)
P4	2.07E−05(+)	2.25E−10(+)	3.15E−01(=)	3.08E−04(+)	1.00E−07(+)	1.67E−04(+)	5.01E−23(+)
P5	6.62E−01(=)	3.22E−01(=)	1.00E−00(=)	2.71E−06 (+)	6.03E−05(+)	3.12E−30(+)	1.62E−34(+)
P6	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)	1.37E−04(+)	1.00E−00(=)	3.97E−04(+)	1.00E−00(=)
P7	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)	1.00E−00(=)
P8	1.39E−18(+)	2.61E−02 (+)	1.22E−01 (=)	2.65E−02 (+)	9.34E−16(+)	2.11E−06 (+)	1.06E−07(+)
P9	1.00E−00(=)	1.00E−00(=)	3.42E−01(=)	1.24E−01(=)	1.11E−04(+)	1.34E−37(+)	5.64E−39(+)
P10	3.41E−10(+)	9.06E−01(=)	4.91E−01(=)	5.59E−01(=)	2.51E−01(=)	6.51E−07(+)	7.81E−06(+)
P11	1.73E−03 (+)	1.22E−03(+)	2.02E−04(+)	4.03E−03 (+)	6.64E−27(+)	2.33E−04 (+)	1.35E−19(+)
P12	3.45E−02 (+)	3.08E−01(=)	2.45E−02 (+)	3.43E−01(=)	6.03E−01(=)	1.38E−02 (+)	6.14E−01(=)
P13	1.00E−00(=)	1.00E−00(=)	8.92E−01(=)	7.12E−01(=)	6.78E−04(+)	3.62E−12(+)	5.34E−08(+)
P14	3.32E−01(=)	3.21E−19 (+)	8.69E−01(=)	8.69E−01(=)	3.26E−18(+)	9.48E−20(+)	1.38E−20(+)
P15	6.11E−07(+)	2.01E−07(+)	2.27E−13(+)	1.02E−07(+)	1.78E−39(+)	1.78E−06(+)	8.96E−10(+)
P16	1.68E−35(+)	5.18E−36(+)	3.96E−29(+)	1.69E−30(+)	5.18E−36(+)	2.32E−38(+)	3.34E−41(+)

of search in the proposed algorithm. In most of the problems, the IMI-GWO algorithms outperform SSA, MFO and SCA. Thus, the table clearly indicates the better performance of the proposed IMI-GWO algorithm as compared to the other comparative algorithms, and therefore, the IMI-GWO algorithm can be considered a better optimizer.

In Table 4, the comparison of the proposed IMI-GWO is made with some other state-of-the-art algorithms which are applied in the literature to solve the same problem set. The table shows that out of sixteen problems in thirteen problems IMI-GWO gives 100% success and only in one problems algorithm is not able to find any success. However, RST2ANU algorithm is 100% successful in seven problems, and in seven problems, the success is below 50%. In AXNUM algorithm, 100% success is obtained only in four problems, and in six problems, success is below 50%. MI-LXPM able to solve six problems with 100% success and provides success below 50% only in two problems. Thus, in most of the problems IMI-GWO algorithm outperforms other algorithms in sense of success rate. Overall in terms of

Table 4 Comparison of results for problems P1–P16

Problem	IMI-GWO		RST2ANU		AXNUM		MI-LXPM	
	S	f_E	S	f_E	S	f_E	S	f_E
P1	100	1300	47	173	86	1728	84	172
P2	100	300	54	657	67	82	85	64
P3	100	9000	4	2, 21, 129	35	65, 303	43	18, 608
P4	0	–	2	14, 89, 713	82	45, 228	**93**	10, 933
P5	100	630	75	2673	95	13, 820	100	671
P6	100	120	100	108	100	432	100	84
P7	100	1200	0	–	45	16, 077	**59**	7447
P8	98	2, 00, 000	15	1, 80, 859	3	1950	41	3571
P9	100	50	100	189	100	4946	100	100
P10	100	1760	100	545	33	700	93	258
P11	100	3250	100	2500	97	863	100	171
P12	81	39, 000	29	6445	19	3, 80, 115	71	2, 99, 979
P13	100	40	100	35	91	456	99	77
P14	100	90	19	214	100	1444	100	78
P15	100	50, 000	100	3337	9	2, 67, 177	92	2437
P16	100	170	100	697	100	256	100	250

S = average percentage of success, f_E = average number of function evaluations

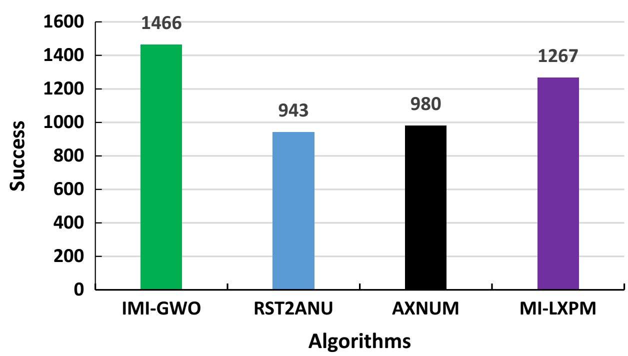

Fig. 2 Performance of various algorithms in terms of total number of success

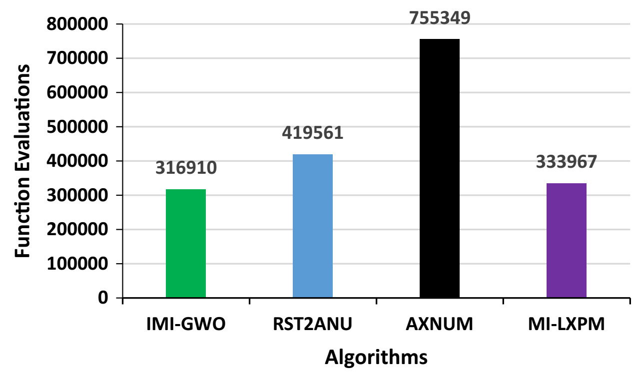

Fig. 3 Performance of various algorithms in terms of total function evaluations

success and average function evaluations, IMI-GWO gives very competitive results as compared to other search algorithms. Figures 2 and 3 show the performance of IMI-GWO, RST2ANU, AXNUM and MI-LXPM in terms of success and total average function evaluations used by algorithms. In these figures, problems where the algorithm is successful at least ones are considered.

5.2 Convergence Analysis

The convergence behaviour for the algorithms IMI-GWO and MI-GWO corresponding to the various problems are plotted in Figs. 4 and 5. These curves are plotted corresponding to the mean value of the objective functions for intermediate iterations in 100 runs. In these graphs, the horizontal axis represents the number of iterations and the vertical axis represents the objective function value. The convergence curve shows the efficiency of the proposed IMI-GWO algorithm in terms of better convergence behaviour.

5.3 Performance Index Analysis

In order to analyse the comparative performance of IMI-GWO, RST2ANU, AXNUM and MI-LXPM algorithms, the performance index (PI) is calculated [44] for all the algorithms. PI analyses the performance of algorithms based on the weights assigned to success and function evaluations. The performance index for the ith algorithm can be evaluated as

$$PI_i = \frac{1}{N}\sum\nolimits_{j=1}^{N} w_1 a_1^j + w_2 a_2^j, \tag{23}$$

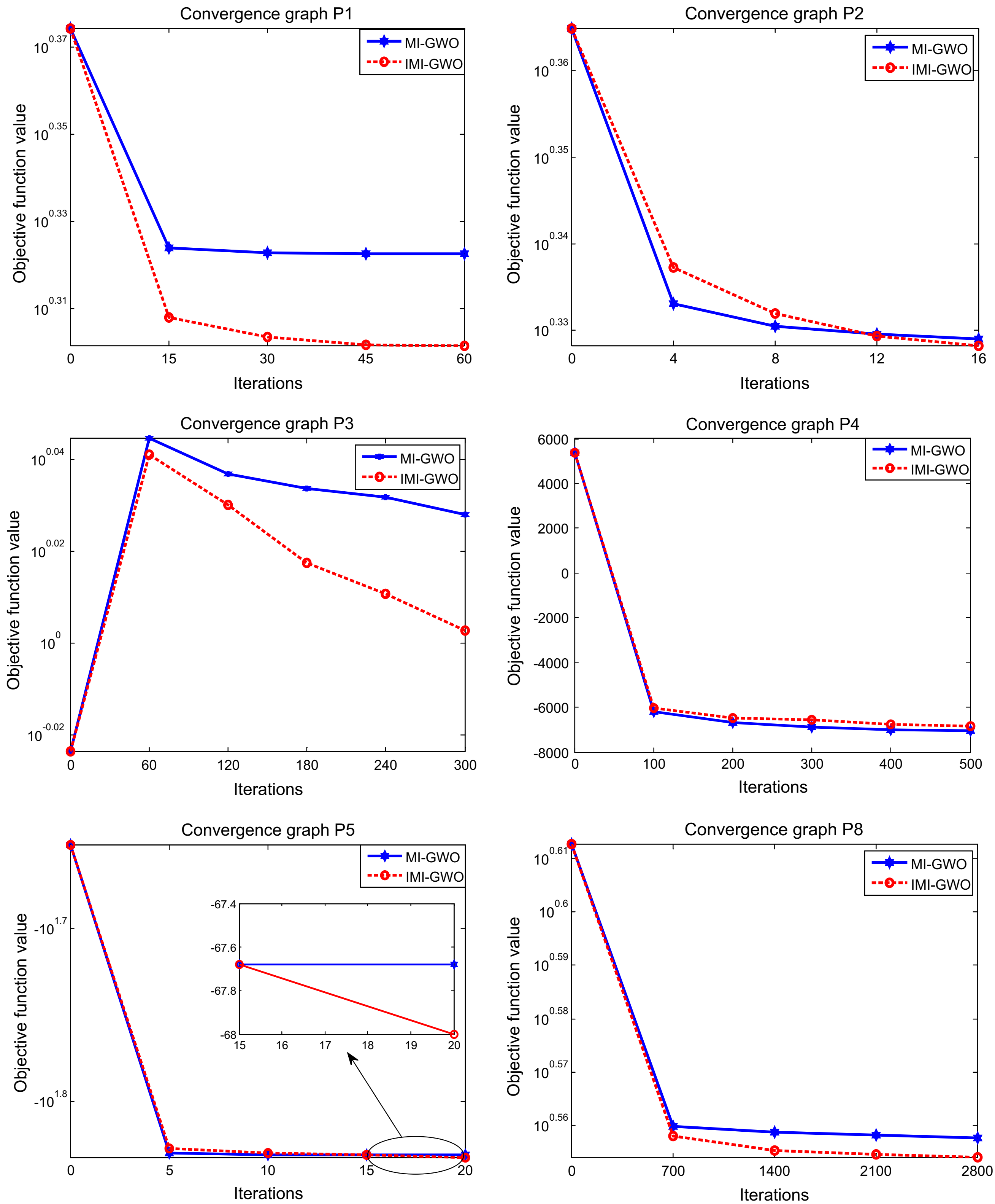

Fig. 4 Convergence graphs for selected mixed-integer problems

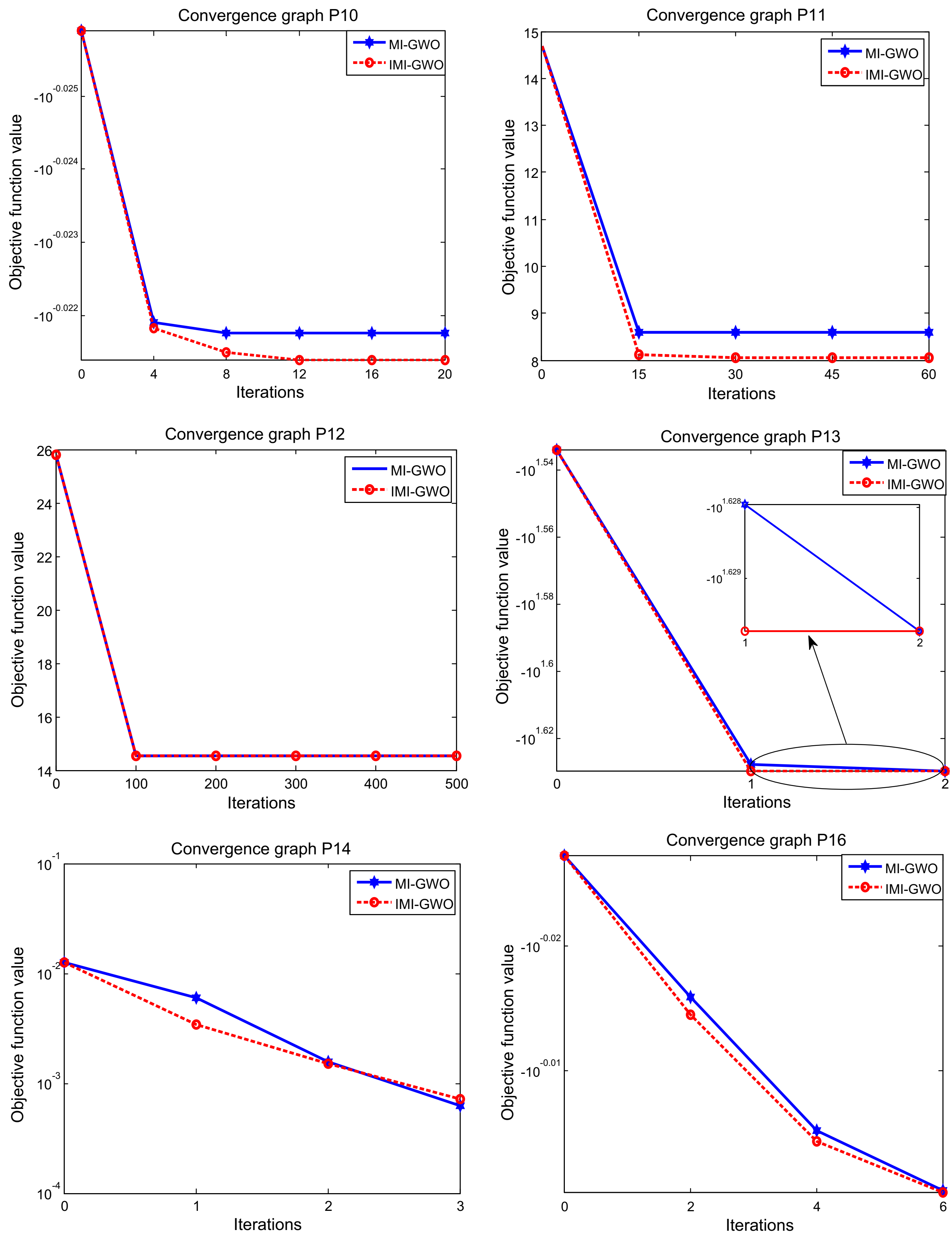

Fig. 5 Convergence graphs for selected mixed-integer problems

Table 5 Performance index (PI) value for various algorithms with ranking corresponding to different weights (w)

Algorithm ↓	weight(w) → 0	0.2	0.4	0.6	0.7	0.8	1
IMI-GWO	0.5842(3)	0.6643(3)	0.7445(2)	0.8247(2)	0.8647(1)	0.9048(1)	0.9850(1)
RST2ANU	0.7245(2)	0.7143(2)	0.7041(3)	0.6939(3)	0.6889(3)	0.6838(3)	0.6736(3)
AXNUM	0.5011(4)	0.5315(4)	0.5619(4)	0.5923(4)	0.6075(4)	0.6227(4)	0.6531(4)
MI-LXPM	0.7828(1)	0.7992(1)	0.8155(1)	0.8319(1)	0.8401(2)	0.8483(2)	0.8646(2)

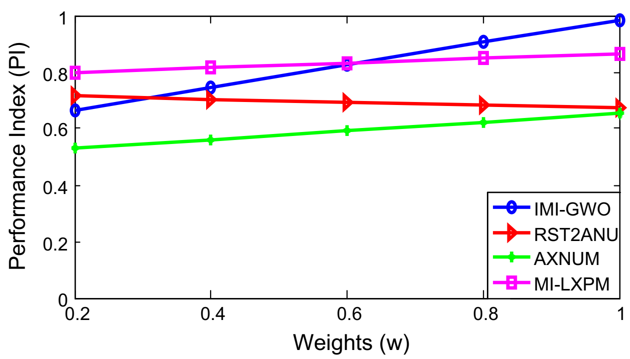

Fig. 6 Performance index graph for various algorithms

where

$$a_1^j = \frac{Sr^j}{Tr^j} \text{ and } a_2^j = \begin{cases} \frac{Mf^j}{Af^j} & \text{if } Sr^j > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

Sr^j denotes the number that the algorithm i is successful on the problem j and Tr^j denotes the total number of times the jth problem is solved. Also, N is the total number of problems, and Mf^j is the minimum of average function evaluations used by all the considered algorithm for the jth problem. Af^j is the average number of function evaluations used by ith algorithm for the jth problem.

Furthermore, w_1 and w_2 are non-negative weights such that $w_1 + w_2 = 1$. If $w_1 = w$, then $w_2 = 1 - w$. The performance index (PI) value is calculated corresponding to the weights $w = 0, 0.2, 0.4, 0.6, 0.7, 0.8$ and $w = 1$. The obtained PI values for all the algorithms are reported in Table 5, and the algorithms are ranked for different weights corresponding to the PI value in the same table. The PI graphs for all the algorithms are shown in Fig. 6. The figures show that IMI-GWO algorithm has better PI than MI-LXPM when $w \geq 0.7$, i.e. when more weight is assigned to the success factor than function evaluations, while IMI-GWO algorithm has better PI than AXNUM and RST2ANU for all the weights $w \geq 0.4$.

Therefore, the analysis of results based on PI shows that IMI-GWO algorithm solves all the problems with better success as compared to other algorithms. The PI analysis also shows that when function evaluations and success both are the requirements of the user then the IMI-GWO algorithm outperforms all other reported algorithms.

5.4 Computational Complexity

The computational complexity of an optimization algorithm is a key metric for evaluating the run-time of an algorithm. The computational complexity can be defined based on the structure of the algorithm. The computational complexity of GWO depends on the number of wolves in a pack, the dimension of the problem and the maximum number of iterations. Overall, by analysing the steps of algorithms from their pseudo-codes, the computational complexity of the proposed algorithm IMI-GWO, classical GWO and other algorithms is defined as follows:

$$O(\text{IMI-GWO}) = O(T\,(n \times d) + N \times d) \tag{25}$$

$$O(GWO) = O(T(n \times d)) \tag{26}$$

$$O\,(mGWO) = O\,(Fitness\,GWO)$$

$$= O(Weighted\,GWO) = O(T(n \times d)) \tag{27}$$

$$O\,(SCA) = O\,(MFO) = O\,(SSA) = O(T(n \times d)), \tag{28}$$

where T represents the maximum number of iterations, n represents the size of the wolf pack, d represents the size of the dimension and N represents the number as defined in chaotic search phase. The complexity of all the algorithms is calculated with the help of their pseudo-codes.

Thus, from the experimental results, it can be observed that the proposed algorithm (IMI-GWO) has solved the integer and mixed-integer optimization problems with better accuracy and better success rate in most of the test problems as compared to the algorithms. Therefore, in terms of acquiring better accuracy, the proposed algorithm can be considered a better optimizer than classical GWO and other algorithms. But, in terms of computational complexity, the proposed algorithm is more complex than other algorithms.

Table 6 Comparison of results on gear train design problem

Algorithm	Optimal solution				$f_{1.min}$	Max function evaluations
	T_d	T_b	T_a	T_f		
IMI-GWO	19	16	43	49	2.7009×10^{-12}	600
CS	19	16	43	49	2.7009×10^{-12}	5000
MBA	16	19	49	43	2.7009×10^{-12}	1,120
GA	33	14	17	50	1.3620×10^{-9}	N/A
ABC	19	16	44	49	2.78×10^{-9}	40,000
ALM	33	15	13	41	2.147×10^{-8}	N/A
MI-GWO	20	20	47	59	9.7409×10^{-10}	600
Cauchy GWO	19	16	43	49	2.7009×10^{-12}	600
SSA	31	13	49	57	9.9399×10^{-11}	850
MFO	23	16	50	51	1.1834×10^{-09}	850
WOA	17	14	30	55	1.3616×10^{-09}	850
SCA	28	17	55	60	2.7009×10^{-12}	1270

6 Applications of IMI-GWO

6.1 Gear Train Design Problem

This problem is the study of parameters T_d, T_b, T_a and T_f with the objective to minimize the gear ratio [45,46]. T_d, T_b, T_a and T_f denote the number of tooth for four gears of a train. The mathematical formulation of the problem can be stated as follows:

$$\textbf{\textit{Min}} \quad f_1(X) = \left(\frac{1}{6.931} - \frac{T_d T_b}{T_a, T_f}\right)^2 = \left(\frac{1}{6.931} - \frac{x_1 x_2}{x_3, x_4}\right)^2 \tag{29}$$

$$X = \left(T_d, T_b, T_a, T_f\right) = (x_1, x_2, x_3, x_4)$$
$$\textbf{\textit{s.t.}} \quad 12 \le T_d, T_b, T_a, T_f \le 60. \tag{30}$$

The obtained best solution by IMI-GWO, MI-GWO and other algorithm is reported in Table 6. The table shows that obtained solution for IMI-GWO is same as algorithm CS [47], MBA [48], but CS and MBA consume more function evaluations. Also this problem has been solved by artificial bee colony (ABC) [49], genetic algorithm (GA) [50], ALM [51] and cauchy GWO [52]. In the table, the results are also reported for some recent studies such as SSA [41], MFO [40], WOA, [9] and SCA [39]. The results presented in Table 6 verify the better performance of IMI-GWO algorithm as compared to the other algorithms.

6.2 Pressure Vessel Design

The objective of the problem is to minimize the total cost including the cost of welding, material, forming. This problem consists of two discrete parameters, namely thickness of shell (T_s) and thickness of head (T_h), and two other parameters, namely inner radius (R) and length of cylindrical section of vessel (L). The discrete parameters (T_s) and (T_h) are integral multiple of 0.0625. Mathematically the problem can be stated as follows:

$$\textbf{\textit{Min}} \quad \boldsymbol{f_2}(\boldsymbol{x}) = 0.6224\, x_1 x_3 x_4 + 1.7781\, x_2 x_3^2 + 3.1661\, x_1^2 x_4 + 19.84\, x_1^2 x_3 \tag{31}$$

$$x = (x_1, x_2, x_3, x_4) = (\boldsymbol{T_s}, \boldsymbol{T_h}, \boldsymbol{R}, \boldsymbol{L})$$

$$\textbf{\textit{s.t.}} \quad g_1(x) = -x_1 + 0.0193\, x_3 \le 0 \tag{32}$$

$$g_2(x) = -x_2 + 0.00954\, x_3 \le 0 \tag{33}$$

$$g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \tag{34}$$

$$g_4(x) = x_4 - 240 \le 0 \tag{35}$$

$$1 \times 0.0625 \le x_1, x_2 \le 99 \times 0.0625 \tag{36}$$

$$10 \le x_3, x_4 \le 200. \tag{37}$$

In the literature, this problem is solved by augmented Lagrangian multiplier [51], branch-and-bound [53] classical techniques. Some nature-inspired algorithms, for example, PSO [54], GA (Coello) [55] and GA (Deb and Gene) [56], PSO (He and Wang) [57], GSA [58] are also used to solve this problem. For this problem, 30 runs have been considered and 2,40,000 function evaluations are used by proposed algorithm. The obtained results are presented in Table 7. In the table, the results are also reported for some recent studies such as SSA [41], MFO [40], WOA, [9] and SCA [39]. The table verifies the better search-ability of the proposed algorithm in obtaining the optimal cost.

Table 7 Comparison of results on pressure vessel design problem

Algorithm	Optimal solution				Optimum cost $f_{2,min}$
	x_1	x_2	x_3	x_4	
IMI-GWO	0.8125	0.4375	42.09844	176.6378	6059.7408
Branch-and-bound	1.125	0.625	47.70	117.701	8129.1036
Lagrangian multiplier	1.125	0.625	58.291	43.69	7198.0428
GA (Deb and Gene)	0.9375	0.500	48.329	112.6790	6410.3811
GA (Coello)	0.8125	0.4345	40.3239	200.00	6288.7445
GWO	0.8125	0.4375	42.09834	176.6389	6059.7506
PSO (He and Wang)	0.8125	0.4375	42.0913	176.7465	6061.0780
GSA	1.1250	0.6250	55.9887	84.4542	8538.8360
SSA	0.9375	0.5	48.57508	110.0682	6370.7870
MFO	0.8125	0.4375	42.098445	176.636596	6059.7143
WOA	0.8125	0.4345	42.0983	176.6390	6059.7410
SCA	1.000	0.5	51.5833	87.2884	6467.8195

7 Conclusion and Future Scope

In the present study, an improved version of classical GWO called IMI-GWO is proposed to solve constrained and non-linear integer and mixed-integer optimization problems. To handle the constraints, a simple constraint handling technique based on constraint violation is employed which is a natural way of selecting the best solutions from the set of solutions. In the paper, two aims are focused: firstly to investigate the exploration and exploitation ability of classical GWO in solving the integer and mixed-integer problems, MI-GWO has been proposed, and secondly, an improved version called IMI-GWO is proposed to solve integer and mixed-integer problems with high accuracy. The IMI-GWO uses the concept of opposite numbers in enhancing the exploration ability of grey wolves in the algorithm. In the proposed IMI-GWO algorithm, the chaotic local search is used to explore the new regions around the set of best solutions. To investigate the performance of the proposed IMI-GWO algorithm, a set of 16 integer and mixed-integer optimization problems and two engineering optimization problems with discrete search space are taken. The comparison with state-of-the-art algorithms which are applied to solve these problems in the literature and the comparison with some recently improved version of GWO shows that the proposed algorithm IMI-GWO solves the problems having discrete search space with high accuracy and high success rate. The convergence analysis and performance index (PI) analysis also verify the better search efficiency of the proposed IMI-GWO algorithm as compared to other algorithms. The results on gear train design and pressure vessel design problems also demonstrate the better ability of IMI-GWO than other algorithms. Overall, from the experimental results and analysis through various metrics, it can be concluded that the proposed IMI-GWO is better optimizer than other comparative algorithms in terms of accuracy in determining the solution when dealing with discrete search space.

In the future, the proposed IMI-GWO algorithm can be used for discrete and mixed-integer programming problems of engineering applications. Also, we will investigate the performance of the proposed algorithm in solving the constrained and unconstrained multiobjective optimization problems. The binary version of the proposed algorithm can also be developed in future, and its performance can be examined on binary problems such as knapsack problem and quadratic assignment problem.

Acknowledgements The first author would like to thank the Ministry of Human Resources, Government of India, for funding this research (Grant No. MHR-02-41-113-429).

Appendix

Integer/Mixed-Integer Optimization Problems

$$\begin{aligned} \boldsymbol{P1}: \quad \boldsymbol{Min} \quad & f_1(x_1, x_2) = 2x_1 + x_2 \\ \boldsymbol{s.t.} \quad & 1.25 - x_1^2 - x_2 \leq 0 \\ & x_1 + x_2 - 1.6 \leq 0 \\ & 0 \leq x_1 \leq 1.6 \\ & x_2 \in \{0, 1\}. \end{aligned}$$

The global optima is 2 corresponding to $(x_1, x_2) = (0.5, 1)$. This problem has been taken from [59] and also reported in [12,16,18,60].

$$\begin{aligned} \boldsymbol{P2}: \quad \boldsymbol{Min} \quad & f_2(x_1, x_2) = 2x_1 - x_2 - \log(x_1/2) \\ \boldsymbol{s.t.} \quad & -x_1 - \log\left(\frac{x_1}{2}\right) + x_2 \leq 0 \end{aligned}$$

$$0.5 \le x_1 \le 1.5$$
$$*x_2 \in \{0, 1\}.$$

The global optima is 2.124 corresponding to $(x_1, x_2) = (1.375, 1)$. This problem is picked up from [12] and it is the modified form of the problem reported in [16,18,60].

$$\textbf{\textit{P3}}: \quad \textbf{\textit{Min}}\ f_3(x_1, x_2, x_3) = 0.8 + 5(x_1 - 0.5)^2 - 0.7x_3$$
$$\textbf{\textit{s.t.}} \quad -e^{(x_1-0.2)} - x_2 \le 0$$
$$1 + x_2 + 1.1\,x_3 \le 0$$
$$x_1 - 1.2\,x_3 - 0.2 \le 0$$
$$0.2 \le x_1 \le 1.1$$
$$-2.22554 \le x_2 \le -1$$
$$x_3 \in \{0, 1\}.$$

The global optima is 1.07654 corresponding to $(x_1, x_2, x_3) = (0.94194, -2.1, 1)$. This problem is selected from [60] and also solved in [12,16,18].

$$\textbf{\textit{P4}}: \quad \textbf{\textit{Min}}\ f_4(x_1, x_2) = (x_1 - 10)^3 + (x_2 - 20)^3$$
$$\textbf{\textit{s.t.}} \quad 100 - (x_1 - 5)^2 - (x_2 - 5)^2 \le 0$$
$$(x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0$$
$$13 \le x_1 \le 100$$
$$0 \le x_2 \le 100.$$

The global optima is -6961.81381 corresponding to $(x_1, x_2) = (14.095, 0.84296)$. This example is taken from [13] and also solved in [18].

$$\textbf{\textit{P5}}: \quad \textbf{\textit{Min}}\ f_5(x_1, x_2, x_3) = x_1^2 + x_1x_2 + 2\,x_2^2 - 6\,x_1 - 2\,x_2 - 12\,x_3$$
$$\textbf{\textit{s.t.}} \quad -15 + 2\,x_1^2 + x_2^2 \le 0$$
$$-3 - x_1 + 2\,x_2 + x_3 \le 0$$
$$0 \le x_1, x_2, x_3 \le 10 \text{ and integers.}$$

The global optima is -68 corresponding to $(x_1, x_2, x_3) = (2, 0, 5)$. This problem is chosen from [61] and also given in [18].

$$\textbf{\textit{P6}}: \quad \textbf{\textit{Min}}\ f_6(x_1, x_2, x_3, x_4) = (x_1 + 2\,x_2 + 3\,x_3 - x_4) \cdot (2\,x_1 + 5\,x_2 + 3\,x_3 - 6\,x_4)$$
$$\textbf{\textit{s.t.}} \quad 4 - x_1 - 2\,x_2 - x_3 - 3\,x_4 \le 0$$
$$x_i \in \{0, 1\} \text{ for } i = 1, 2, 3, 4.$$

The global optima is -6 corresponding to $(x_1, x_2, x_3, x_4) = (0, 0, 1, 1)$. This problem has been taken from [59] and this represents a quadratic capital budgeting problem. This problem is also given in [16,18].

$$\textbf{\textit{P7}}: \quad \textbf{\textit{Min}}\ f_7(x_1, x_2, x_3) = 7.5\,x_1 + 5.5\,(1 - x_1) + 7\,x_2 + 6\,x_3 + 50\,\frac{x_1/(2\,x_1 - 1)}{0.9[1 - e^{(-0.5\,x_2)}]} + 50\,\frac{1 - [x_1/(2\,x_1 - 1)]}{0.8[1 - e^{(-0.4\,x_3)}]}$$
$$\textbf{\textit{s.t.}} \quad 0.9\left[1 - e^{(-0.5\,x_2)}\right] - 2\,x_1 \le 0$$
$$0.8\left[1 - e^{(-0.4\,x_3)}\right] - 2\,(1 - x_1) \le 0$$
$$x_2 - 10\,x_1 \le 0$$
$$x_3 - 10(1 - x_1) \le 0$$
$$x_2, x_3 \ge 0$$
$$x_1 \in \{0, 1\}.$$

The global optima is 99.245209 corresponding to $(x_1, x_2, x_3) = (1, 3.514237, 0)$. This problem has been chosen from [12].

$$\textbf{\textit{P8}}: \quad \textbf{\textit{Min}}\ f_8(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (x_4 - 1)^2 + (x_5 - 1)^2 + (x_6 - 1)^2 + (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 - \log(x_7 + 1)$$
$$\textbf{\textit{s.t.}} \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 - 5 \le 0$$
$$x_1^2 + x_2^2 + x_3^2 + x_6^2 - 5.5 \le 0$$
$$x_1 + x_4 - 1.2 \le 0$$
$$x_2 + x_5 - 1.8 \le 0$$
$$x_3 + x_6 - 2.5 \le 0$$
$$x_1 + x_7 - 1.2 \le 0$$
$$x_2^2 + x_5^2 - 1.64 \le 0$$
$$x_3^2 + x_6^2 - 4.25 \le 0$$
$$x_3^2 + x_5^2 - 4.64 \le 0$$
$$x_1, x_2, x_3 \ge 0$$
$$x_4, x_5, x_6, x_7 \in \{0, 1\}.$$

The global optima is 3.557463 corresponding to $(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (0.2, 1.280624, 1.954483, 1, 0, 0, 1)$. This problem has been given in [12] and also solve in [16,18,59,62].

$$\textbf{\textit{P9}}: \quad \textbf{\textit{Min}}\ f_9(x_1, x_2, x_3, x_4, x_5) = 5.357854\,x_1^2 + 0.835689\,x_3x_4 + 37.29329\,x_4 - 40792.141$$
$$\textbf{\textit{s.t.}} \quad a_1 + a_2\,x_5\,x_3 + a_3\,x_4\,x_2 - a_4\,x_1\,x_3 - 92 \le 0$$
$$a_5 + a_6\,x_5\,x_3 + a_7\,x_4\,x_5 + a_8x_1^2 - 110 \le 0$$
$$a_9 + a_{10}\,x_1\,x_3 + a_{11}\,x_4\,x_1 + a_{12}x_1\,x_2 - 25 \le 0$$
$$27 \le x_1, x_2, x_3 \le 45$$
$$x_4 \in \{78, 79, \ldots, 102\}$$
$$x_5 \in \{33, 34, \ldots, 45\}.$$

The global optima is -32217.42 corresponding to $(x_1, x_3, x_4) = (27, 27, 78)$ and for any combination of (x_2, x_5). This problem is reported in [12,16,18]. The coefficients of this problem are presented in Table 7.

$$
\begin{aligned}
\textbf{\textit{P10}}: \textbf{\textit{Min}}\ & f_{10}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) \\
& = -\left(1 - 0.1^{x_1} 0.2^{x_2} 0.15^{x_3}\right) \\
& \times \left(1 - 0.05^{x_4} 0.2^{x_5} 0.15^{x_6}\right)\left(1 - 0.02^{x_7} 0.06^{x_8}\right) \\
\textit{s.t.}\ & 1 - x_1 - x_2 - x_3 \le 0 \\
& 1 - x_4 - x_5 - x_6 \le 0 \\
& 1 - x_7 - x_8 \le 0 \\
& 3x_1 + x_2 + 2x_3 + 3x_4 + 2x_5 x_6 + 3x_7 \\
& \quad + 2x_8 - 10 \le 0 \\
& x_i \in \{0, 1\} \text{ for } i = 1, 2, \ldots, 8.
\end{aligned}
$$

The global optima is -0.94347 corresponding to $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) = (0, 1, 1, 1, 0, 1, 1, 0)$. This problem is taken from [63] and also solved in [16,18].

$$
\begin{aligned}
\textbf{\textit{P11}}: \quad \textbf{\textit{Min}}\ & f_{11}(x_1, x_2, x_3, x_4, x_5) \\
& = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 \\
\textit{s.t.}\ & 4 - x_1 - 2x_2 - x_4 \le 0 \\
& 3 - x_2 - 2x_3 \le 0 \\
& 5 - x_1 - 2x_5 \le 0 \\
& x_1 + 2x_2 + 2x_3 - 6 \le 0 \\
& 2x_1 + x_3 - 4 \le 0 \\
& x_1 + 4x_5 - 13 \le 0 \\
& 0 \le x_i \le 3 \text{ and integer for } i = 1, 2, 3, 4, 5.
\end{aligned}
$$

The global optima is 8 corresponding to $(x_1, x_2, x_3, x_4, x_5) = (1, 1, 1, 1, 2)$. This problem has been chosen from [64] and also reported in [18,61].

$$
\begin{aligned}
\textbf{\textit{P12}}: \quad \textbf{\textit{Min}}\ & f_{12}(x_1, x_2, x_3, x_4, x_5, x_6, x_7) \\
& = x_1 x_7 + 3x_2 x_6 + x_3 x_5 + 7x_4 \\
\textit{s.t.}\ & 6 - x_1 - x_2 - x_3 \le 0 \\
& 8 - x_4 - x_5 - 6x_6 \le 0 \\
& 7 - x_2 - 3x_5 - x_1 x_6 \le 0 \\
& 25 - 4x_2 x_7 - 3x_4 x_5 \le 0 \\
& 7 - 3x_1 - 2x_3 - x_5 \le 0 \\
& 3x_1 x_3 + 6x_4 + 4x_5 - 20 \le 0 \\
& x_6 x_7 + 2x_3 + 4x_1 - 15 \le 0 \\
& 0 \le x_1, x_2, x_3 \le 4 \\
& 0 \le x_4, x_5, x_6 \le 2 \\
& 0 \le x_7 \le 6 \\
& x_i\text{'s are integer for } i = 1, 2, \ldots, 7.
\end{aligned}
$$

The global optima is 14 corresponding to $(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (0, 2, 4, 0, 2, 1, 4)$. This problem is chosen from [64] and also solved in [18,62].

$$
\begin{aligned}
\textbf{\textit{P13}}: \quad \textbf{\textit{Min}}\ & f_{13}(x_1, x_2) = \exp(-x_1) + x_1^2 - x_1 x_2 \\
& - 3x_2^2 - 6x_2 + 4x_1 \\
\textit{s.t.}\ & 2x_1 + x_2 - 8 \le 0 \\
& -2 + x_2 - x_1 \le 0 \\
& 0 \le x_1, x_2 \le 3 \text{ and integers.}
\end{aligned}
$$

The global optima is -42.632 corresponding to $(x_1, x_2) = (1, 3)$. This problem is chosen from [65] and also solved in [18,61].

$$
\begin{aligned}
\textbf{\textit{P14}}: \quad \textbf{\textit{Min}}\ & f_{14}(x_1, x_2, x_3) = \sum_{i=1}^{9}\left[e^{\left(-\frac{(v_i - x_2)^{x_3}}{x_1}\right)} - 0.01\, i\right]^2 \\
& \text{where } v_i = 25 + (-50 \cdot log(0.01\, i))^{2/3} \\
\textit{s.t.}\ & 0 \le x_1 \le 100 \\
& 0 \le x_2 \le 25.6 \\
& 0 \le x_3 \le 5 \\
& x_1, x_2 \text{ integers.}
\end{aligned}
$$

The global optima is 0 corresponding to $(x_1, x_2, x_3) = (50, 25, 1.50)$. This problem is reported in [66] and also reported in [17,18].

$$
\begin{aligned}
\textbf{\textit{P15}}: \quad \textbf{\textit{Min}}\ & f_{15}(x_1, x_2, x_3, x_4, x_5) = x_1^2 + x_2^2 + 3x_3^2 \\
& + 4x_4^2 + 2x_5^2 - 8x_1 - 2x_2 - 3x_3 - x_4 - 2x_5 \\
\textit{s.t.}\ & 55 \le x_1 + x_2 + x_3 + x_4 + x_5 \le 400 \\
& x_1 + 2x_2 + 2x_3 + x_4 + 6x_5 - 800 \le 0 \\
& 2x_1 + x_2 + 6x_3 - 200 \le 0 \\
& x_3 + x_4 + 5x_5 - 200 \le 0 \\
& 48 - x_1 - x_2 - x_3 - x_4 \le 0 \\
& 34 - x_2 - x_4 - x_5 \le 0 \\
& 104 - 6x_1 - 7x_5 \le 0 \\
& 0 \le x_i \le 99 \text{ and integer for } i = 1, 2, \ldots, 5.
\end{aligned}
$$

The global optima is 807 corresponding to $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) = (16, 22, 5, 5, 7)$. This problem is taken from [67].

$$
\begin{aligned}
\textbf{\textit{P16}}: \quad \textbf{\textit{Min}}\ & f_{16}(m, r) = \prod_{i=1}^{4}\left[1 - (1 - r_i)^{m_i}\right] \\
\textit{s.t.}\ & \sum_{j=1}^{4} u_j m_j^2 \le u_Q \\
& \sum_{j=1}^{4} C\left(r_j\right)\left[m_j + e^{\left(\frac{m_j}{4}\right)}\right] \le c_Q \\
& \sum_{j=1}^{4} w_j\left[m_j \cdot e^{\left(\frac{m_j}{4}\right)}\right] \le w_Q
\end{aligned}
$$

Table 8 Data set for the coefficients in problem P9

a_1	85.334407	a_5	80.51249	a_9	9.300961
a_2	0.0056858	a_6	0.0071317	a_{10}	0.0047026
a_3	0.0006262	a_7	0.0029955	a_{11}	0.0012547
a_4	0.0022053	a_8	0.0021813	a_{12}	0.0019085

$$0 \le m_i \le 10 \text{ and integers}$$
$$0 \le r_j \le 1 - 10^{-6} \quad ,$$

where u_j is the product of weight and volume per element at stage j, w_j is the weight of the components at stage j and $C(r_j)$ is the cost of components with reliability r_j at stage j which is defined as

$$C(r_j) = \alpha_j \left(\frac{-T}{\ln r_j}\right)^{\beta_j}.$$

c_Q		400		
u_Q		250		
w_Q		500		
Operating time (T)		100 h		
j	$10^5\alpha_j$	β_j	u_j	w_j
1	1.0	1.50	1	6
2	2.3	1.50	2	6
3	0.3	1.50	3	8
4	2.3	1.50	2	7

Here α_j and β_j are the physical characteristics of components at stage j and T is the operating time during which the components must not fail (Table 8).

The global optima is -0.999955 corresponding to $(r_1, r_2, r_3, r_4, m_1, m_2, m_3, m_4) = (5, 5, 4, 6, 0.899845, 0.887909, 0.948990, 0.851017)$. This problem is taken from [68] and also reported in [13,18].

References

1. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge (1992)
2. Storn, R.; Price, K.: Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**(4), 341–359 (1997)
3. Eberhart, R.; Kennedy, J.: A new optimizer using particle swarm theory. In: Micro Machine and Human Science, 1995. MHS'95. In: Proceedings of the Sixth International Symposium on (pp. 39–43). IEEE (1995)
4. Glover, F.: Tabu search—part I. ORSA J. Comput. **1**(3), 190–206 (1989)
5. Basturk, B.: An artificial bee colony (ABC) algorithm for numeric function optimization. In: IEEE Swarm Intelligence Symposium, Indianapolis, p. 2006 (2006)
6. Dorigo, M.; Birattari, M.; Stutzle, T.: Ant colony optimization. IEEE Comput. Intell. Mag. **1**(4), 28–39 (2006)
7. Črepinšek, M.; Liu, S.H.; Mernik, M.: Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput. Surv. **45**(3), 35 (2013)
8. Wolpert, D.H.; Macready, W.G.: No free lunch theorems for search (Vol. 10). Technical Report SFI-TR-95-02-010, Santa Fe Institute. (1995)
9. Mirjalili, S.; Lewis, A.: The whale optimization algorithm. Adv. Eng. Softw. **95**, 51–67 (2016)
10. Bansal, J.C.; Sharma, H.; Jadon, S.S.; Clerc, M.: Spider monkey optimization algorithm for numerical optimization. Memet. Comput. **6**(1), 31–47 (2014)
11. Mirjalili, S.; Mirjalili, S.M.; Lewis, A.: Grey wolf optimizer. Adv. Eng. Softw. **69**, 46–61 (2014)
12. Costa, L.; Oliveira, P.: Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. Comput. Chem. Eng. **25**(2), 257–266 (2001)
13. Li, Y.X.; Gen, M.: Nonlinear mixed integer programming problems using genetic algorithm and penalty function. In: Systems, Man, and Cybernetics, 1996., IEEE International Conference on (Vol. 4, pp. 2677–2682). IEEE (1996)
14. Tan, Y.; Tan, G.Z.; Deng, S.G.: Hybrid particle swarm optimization with chaotic search for solving integer and mixed integer programming problems. J. Central South Univ. **21**(7), 2731–2742 (2014)
15. Laarhoven, P.J.M.V.; Peter, J.M.; Aarts, E.H.L.: Simulated annealing. In: Simulated Annealing: Theory and Applications, pp. 7–15. Springer, Netherlands (1987)
16. Cardoso, M.F.; Salcedo, R.L.; de Azevedo, S.F.; Barbosa, D.: A simulated annealing approach to the solution of MINLP problems. Comput. Chem. Eng. **21**(12), 1349–1364 (1997)
17. Mohan, C.; Nguyen, H.T.: A controlled random search technique incorporating the simulated annealing concept for solving integer and mixed integer global optimization problems. Comput. Optim. Appl. **14**(1), 103–132 (1999)
18. Deep, K.; Singh, K.P.; Kansal, M.L.; Mohan, C.: A real coded genetic algorithm for solving integer and mixed integer optimization problems. Appl. Math. Comput. **212**(2), 505–518 (2009)
19. Jayakumar, N.; Subramanian, S.; Ganesan, S.; Elanchezhian, E.B.: Grey wolf optimization for combined heat and power dispatch with cogeneration systems. Int. J. Electr. Power Energy Syst. **74**, 252–264 (2016)
20. Song, H.M.; Sulaiman, M.H.; Mohamed, M.R.: An application of grey wolf optimizer for solving combined economic emission dispatch problems. Int. Rev. Model. Simul. **7**(5), 838–844 (2014)
21. Komaki, G.M.; Kayvanfar, V.: Grey Wolf optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time. J. Comput. Sci. **8**, 109–120 (2015)
22. Muangkote, N.; Sunat, K.; Chiewchanwattana, S.: An improved grey wolf optimizer for training q-Gaussian Radial Basis Functional-link nets. In: Computer Science and Engineering Conference (ICSEC), 2014 International, pp. 209-214. IEEE (2014)
23. Kamboj, V.K.; Bath, S.K.; Dhillon, J.S.: Solution of non-convex economic load dispatch problem using Grey Wolf Optimizer. Neural Comput. Appl. **27**(5), 1301–1316 (2016)
24. Sulaiman, M.H.; Mustaffa, Z.; Mohamed, M.R.; Aliman, O.: Using the gray wolf optimizer for solving optimal reactive power dispatch problem. Appl. Soft Comput. **32**, 286–292 (2015)
25. Mittal, N.; Singh, U.; Sohi, B.S.: Modified grey wolf optimizer for global engineering optimization. Appl. Comput. Intell. Soft Comput. **2016**, 8 (2016)
26. Heidari, A.A.; Pahlavani, P.: An efficient modified grey wolf optimizer with Lévy flight for optimization tasks. Appl. Soft Comput. **60**, 115–134 (2017)
27. Gupta, S.; Deep, K.: A novel random walk grey wolf optimizer. Swarm Evol. Comput. **44**, 101–112 (2018)

28. Gupta, S.; Deep, K.: Random walk grey wolf optimizer for constrained engineering optimization problems. Comput. Intell. **34**(4), 1025–1045 (2018)
29. Long, W.; Jiao, J.; Liang, X.; Tang, M.: An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. Eng. Appl. Artif. Intell. **68**, 63–80 (2018)
30. Tawhid, M.A.; Ali, A.F.: A Hybrid grey wolf optimizer and genetic algorithm for minimizing potential energy function. Memet. Comput. **9**(4), 347–359 (2017)
31. Abed-alguni, B.H.; Barhoush, M.: Distributed grey wolf optimizer for numerical optimization problems. Jordan. J. Comput. Inf. Technol. **4**, 130–149 (2018)
32. Gupta, S.; Deep, K.: An opposition based chaotic grey wolf optimizer for global optimization tasks. J. Exp. Theor. Artif. Intell. (2018). https://doi.org/10.1080/0952813X.2018.1554712
33. Muro, C.; Escobedo, R.; Spector, L.; Coppinger, R.P.: Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations. Behav. Process. **88**(3), 192–197 (2011)
34. Tizhoosh, H.R.: Opposition-based learning: a new scheme for machine intelligence. In: Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on (Vol. 1, pp. 695-701). IEEE (2005)
35. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M.: Opposition versus randomness in soft computing techniques. Appl. Soft Comput. **8**(2), 906–918 (2008)
36. Guo, Z.; Huang, H.; Deng, C.; Yue, X.; Wu, Z.: An enhanced differential evolution with elite chaotic local search. Comput. Intell. Neurosci. **2015**, 6 (2015)
37. Jia, D.; Zheng, G.; Khan, M.K.: An effective memetic differential evolution algorithm based on chaotic local search. Inf. Sci. **181**(15), 3175–3187 (2011)
38. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. **186**(2), 311–338 (2000)
39. Mirjalili, S.: SCA: a sine cosine algorithm for solving optimization problems. Knowl.-Based Syst. **96**, 120–133 (2016)
40. Mirjalili, S.: Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl.-Based Syst. **89**, 228–249 (2015)
41. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M.: Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv. Eng. Softw. **114**, 163–191 (2017)
42. Barr, R.S.; Golden, B.L.; Kelly, J.P.; Resende, M.G.; Stewart, W.R.: Designing and reporting on computational experiments with heuristic methods. J. Heuristics **1**(1), 9–32 (1995)
43. Derrac, J.; García, S.; Molina, D.; Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. **1**(1), 3–18 (2011)
44. Bharti: Controlled Random Search Techniques and Their Applications. Ph.D. Thesis. Department of Mathematics, University of Roorkee, India (1994)
45. Gandomi, A.H.: Interior search algorithm (ISA): a novel approach for global optimization. ISA Trans. **53**(4), 1168–1183 (2014)
46. Sandgren, E.: Nonlinear integer and discrete programming in mechanical design optimization. J. Mech. Des. **112**(2), 223–229 (1990)
47. Gandomi, A.H.; Yang, X.S.; Alavi, A.H.: Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng. Comput. **29**(1), 17–35 (2013)
48. Sadollah, A.; Bahreininejad, A.; Eskandar, H.; Hamdi, M.: Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl. Soft Comput. **13**(5), 2592–2612 (2013)
49. Sharma, T.K.; Pant, M.; Singh, V.P.: Improved local search in artificial bee colony using golden section search (2012) arXiv preprint arXiv:1210.6128
50. Deb, K.; Goyal, M.: A combined genetic adaptive search (GeneAS) for engineering design. Comput. Sci. Inform. **26**, 30–45 (1996)
51. Kannan, B.K.; Kramer, S.N.: An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J. Mech. Des. **116**(2), 405–411 (1994)
52. Gupta, S.; Deep, K.: Cauchy Grey Wolf Optimiser for continuous optimisation problems. J. Exp. Theor. Artif. Intell. **30**(6), 1051–1075 (2018)
53. Sandgren, E.: Nonlinear integer and discrete programming in mechanical design. In: Proceedings of the ASME Design Technology Conference (pp. 95–105) (1988)
54. He, Q.; Wang, L.: An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng. Appl. Artif. Intell. **20**(1), 89–99 (2007)
55. Coello, C.A.C.; Montes, E.M.: Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv. Eng. Inform. **16**(3), 193–203 (2002)
56. Deb, K.: GeneAS: a robust optimal design technique for mechanical component design. In: Evolutionary Algorithms in Engineering Applications (pp. 497–514). Springer, Berlin (1997)
57. He, Q.; Wang, L.: An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng. Appl. Artif. Intell. **20**(1), 89–99 (2007)
58. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S.: GSA: a gravitational search algorithm. Inf. Sci. **179**(13), 2232–2248 (2009)
59. Kocis, G.R.; Grossmann, I.E.: Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis. Ind. Eng. Chem. Res. **27**(8), 1407–1421 (1988)
60. Floudas, C.A.: Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications. Oxford University Press, Oxford (1995)
61. Nguyen, H.T.: Some global optimization techniques and their use in solving optimization problems in crisp and fuzzy environments (Doctoral dissertation. Department of Mathematics, Ph.D. Thesis, University of Roorkee, Roorkee, India) (1996)
62. Yuan, X.; Zhang, S.; Pibouleau, L.; Domenech, S.: Une méthode d'optimisation non linéaire en variables mixtes pour la conception de procédés. RAIRO-Oper. Res. **22**(4), 331–346 (1988)
63. Berman, O.; Ashrafi, N.: Optimization models for reliability of modular software systems. IEEE Trans. Softw. Eng. **19**(11), 1119–1123 (1993)
64. Salkin, H.M.: Integer Programming. Eddison Wesley Publishing Com, Amsterdam (1975)
65. Bazaraa, M.S.; Sherali, H.D.; Shetty, C.M.: Nonlinear Programming: Theory and Algorithms. Wiley, London (2013)
66. Himmelblau, D.M.: Applied Nonlinear Programming. McGraw-Hill Companies, New York (1972)
67. Conley, W.: Computer Optimization Techniques. Petrocelli Books, New Jersy (1984)
68. Dhingra, A.K.: Optimal apportionment of reliability and redundancy in series systems under multiple objectives. IEEE Trans. Reliab. **41**(4), 576–582 (1992)