INNA: An improved neural network algorithm for solving reliability optimization problems

Tanmay Kundu1 · Harish Garg2

Received: 14 February 2021 / Accepted: 28 June 2022 / Published online: 1 August 2022
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

Abstract
The main objective of this paper is to present an improved neural network algorithm (INNA) for solving the reliability-redundancy allocation problem (RRAP) with nonlinear resource constraints. In this RRAP, both the component reliability and the redundancy allocation are to be considered simultaneously. Neural network algorithm (NNA) is one of the newest and efficient swarm optimization algorithms having a strong global search ability that is very adequate in solving different kinds of complex optimization problems. Despite its efficiency, NNA experiences poor exploitation, which causes slow convergence and also restricts its practical application of solving optimization problems. Considering this deficiency and to obtain a better balance between exploration and exploitation, searching procedure for NNA is reconstructed by implementing a new logarithmic spiral search operator and the searching strategy of the learner phase of teaching–learning-based optimization (TLBO) and an improved NNA has been developed in this paper. To demonstrate the performance of INNA, it is evaluated against seven well-known reliability optimization problems and finally compared with other existing meta-heuristics algorithms. Additionally, the INNA results are statistically investigated with the Wilcoxon sign-rank test and Multiple comparison test to show the significance of the results. Experimental results reveal that the proposed algorithm is highly competitive and performs better than previously developed algorithms in the literature.

Keywords Neural network algorithm · Teaching–learning-based optimization · Constrained optimization · Reliability redundancy allocation problem

List of symbols

- \( n = (n_1, n_2, \ldots, n_m) \), the redundancy allocation vector
- \( m \) Number of subsystems
- \( n_i \) The number of components in subsystem \( i \)
- \( n_{i_{\text{max}}} \) Maximum number of components in subsystem \( i \)
- \( n_{i_{\text{min}}} \) Minimum number of components in subsystem \( i \)
- \( r_i \) The components reliability in subsystem \( i \)
- \( b \) is the vector of resource limitation
- \( c_i \) The component cost in subsystem \( i \)
- \( w_i \) The component weight in subsystem \( i \)
- \( C \) Upper limit of the system’s cost
- \( V \) Upper limit of the system’s volume
- \( W \) Upper limit of the system’s weight
- \( v_i \) The component volume in subsystem \( i \)
- \( g_j \) The \( j \)th constraint function
- \( R_i = 1 - (1 - r_i)^{n_i}, \) is the reliability of the \( i \)th subsystem
- \( R_S \) The system reliability
- \( Z^+ \) Set of positive integers in the discrete space
- \( VTV \) The variables that received the value 2 in optimum stage
- \( \text{Dim} \) Dimensions
- \( \text{slack}(g) = (b_k - g_k) \) for \( k = 1, 2, \ldots, l \)
- \( \text{Std} \) Standard deviation
- \( \text{MHAs} \) Meta-heuristics algorithms
- \( \text{RRAP} \) Reliability redundancy allocation problems
- \( \text{P1-P7} \) Reliability issues

1 Department of Mathematics, Chandigarh University, Mohali, Punjab 140413, India
2 School of Mathematics, Thapar Institute of Engineering & Technology, Deemed University, Patiala, Punjab 147004, India
1 Introduction

Since 1950, reliability optimization plays a progressively crucial role because of its critical requirements on engineering and industrial applications like automotive industries, military, aerospace, computer and communication systems, transportation systems, etc. To be more competitive in practical life, the basic goal of reliability engineering is always to improve the reliability of product components or manufacturing systems. Generally, the reliability optimization problem can be distinguished into two classes: integer and mixed-integer reliability problems. In the integer reliability problem, the main task is to allocate the number of system redundant components while the reliability of the components is known. On the other hand, both the redundancy allocation and the reliability of the component are to be designed together in the mixed-integer reliability problem. RRAP [1] is such kind of a problem in which system reliability is maximized through the choices of redundancy and component reliability. To optimize a system RRAP, redundancy levels are considered as integer values and component reliabilities are taken as continuous values lie between zero and one. Therefore, RRAP is a mixed-integer problem approach with the objective of maximizing system reliability under constraints such as the system cost, volume, and weight [2–8]. Obviously, an excellent reliability design facilitates a system to run more safely and reliably. Thus, the study of reliability optimization problems has been achieved great attention and has become a hot research topic in the engineering field.

Because of complexity, RRAP has been considered to be an NP-hard combinatorial optimization problem which has been already considered as the subject of much prior work in many different formulations and over various optimization approaches. Due to this computational difficulty, meta-heuristics algorithms (MHAs) have been successfully applied to a wide range of practical optimization problems to deal with them. In MHAs, optimization problems are assumed like a black box which indicates that there is no need to find the derivative of the mathematical models any more, rather it can be solved only by detecting the inputs and outputs. These methods use a random population of individuals in the search space, apply probabilistic rules and also approximate the optimal solution rather than finding the mathematical optimum, which makes these methods more flexible to find better solutions compared to deterministic methods for solving optimization problems.

Inspiring by biological phenomena and human characteristics, several authors have been developed a variety of population-based optimization techniques to address complex optimization and in terms of the inspiring source, this can be broadly classified into three categories: evolutionary-based, swarm-based, and human-based algorithms. Evolutionary algorithms (EA) mimic the mechanism of biological evolution, which is initiated with a population of random individuals. The first ever EA named Genetic algorithm (GA) based on the Darwinian principle of natural evolution is proposed by Goldberg and Holland [9]. Swarm-based algorithms mimic the social behaviours or intelligence of animals in nature and the main virtue of such algorithms is their collaborative survive ability. Particle swarm optimization (PSO) [10], which resembles the bird flocking social behaviour, is considered as the first swarm-based algorithm introduced in the literature. Finally, the human-based algorithms are inspired by some human nature, activities or perception. A list of some nature-inspired algorithms with their operators is reported in Table 1. Further, several researchers have applied some well-known MHAs to deal with the RRAP, including GA [11–14], PSO [15–17], CS [18, 19], ACO [20], BBO [21], ABC [22], HS [23–26] and DE [27]. In addition to the above mentioned studies, some novel MHAs have also been introduced to solve RRAP, such as grey wolf optimizer algorithm [28] and cuckoos search algorithm [29]. Again, Qiang et al. [30] presented a novel artificial fish swarm algorithm which mimic the social behaviours of fish swarm, for solving large-scale RRAPs. The above-cited algorithms has been successfully tested to solve various kinds of real-world optimization problems. Although, there are some noticeable deficiencies on these algorithms in solving some optimization problems. For example, ABC with its simple structure and an advantage of few parameters, it has been successfully applied to solve various kinds of real-world optimization problems. Despite these efficiencies, the ABC suffers from the problem of local optima stagnation and also it fails to maintain a proper balance between exploration and exploitation. Again, for HHO, it has a good global searching ability and provides high accuracy in extracting the optimal parameters. Although HHO experience is satisfactory in exploration but devoid of exploitation, which forces slow convergence. In case of the algorithm SSA, it has a powerful neighbourhood search ability and it can easily fitted for wide search space which makes SSA an efficient technique. But, there are some major drawbacks of SSA in solving some optimization problems like local optima stagnation, poor convergence tendency, lacking of exploitation etc. Like ABC, HHO and SSA, the other cited algorithms also has some advantages as well as disadvantages in solving optimization problems.

According to the “No Free Launch (NFL)” theorem [52], there exist no MHAs best fitted to solve all optimization problems. Alternatively, it may happen that a particular algorithm gives efficient solutions for some optimization problems, but it may fail to perform well on
et al. [60] first introduced that a neural network technique has been developing very rapidly. Impressively, Sadollah to solve various kinds of optimization problems, and which RRAP.

In recent years, neural network technique has been used to solve various kinds of optimization problems, and which has been developing very rapidly. Impressively, Sadollah et al. [60] first introduced that a neural network technique can be implemented to design an optimization algorithm and proposed a novel meta-heuristic method called neural network algorithm. NNA is one of the newest meta-heuristic algorithms, which is inspired by artificial neural networks (ANNs) and biological nervous systems and the important characteristics like, simple structure, robustness, and scalability, makes NNA an efficient method for solving various kinds of real world problems. Compared to other existing MHAs, the exclusive structure of ANNs guides NNA to find the global optimum solution and shows strong global exploration ability with the minimum chance of getting captured in a local optimum. Despite of these efficiency, the basic NNA has some noticeable deficiency in solving some optimization problems. Firstly, the NNA suffers from the problem of local optima stagnation in solving some large scale optimization problems such as large-scale reliability system with higher dimensions and CEC (Congress of Evolutionary Computation) benchmark test problems. Secondly, the NNA experience is satisfactory in exploration but it fails to maintain a proper balance between exploration and exploitation due to lack of exploitation ability. Finally, it has slow convergence tendency and sometimes, it required more time to evaluate a new solution for some problems. These shortcomings will reduce its applications in some optimization problems with limitations and researchers have applied different search mechanisms and adopted modified operators to upgrade the original NNA. To mention a few- Zhang et al. [61] introduced a novel hybrid algorithm GNNA combining GWO and NNA for solving RRAP. An improved GWO algorithm called random walk gray wolf optimizer (RW-GWO) is presented by Gupta et al. [58] to obtain the optimal redundancies to optimize the system reliability with several constraints like volume, weight, and system cost. Recently, Kundu et al. [59] proposed a hybrid salp swarm algorithm with teaching–learning-based optimization (HSSATLBO) for solving RRAP.

In recent years, neural network technique has been used to solve various kinds of optimization problems, and which has been developing very rapidly. Impressively, Sadollah et al. [60] first introduced that a neural network technique can be implemented to design an optimization algorithm and proposed a novel meta-heuristic method called neural network algorithm. NNA is one of the newest meta-heuristic algorithms, which is inspired by artificial neural networks (ANNs) and biological nervous systems and the important characteristics like, simple structure, robustness, and scalability, makes NNA an efficient method for solving various kinds of real world problems. Compared to other existing MHAs, the exclusive structure of ANNs guides NNA to find the global optimum solution and shows strong global exploration ability with the minimum chance of getting captured in a local optimum. Despite of these efficiency, the basic NNA has some noticeable deficiency in solving some optimization problems. Firstly, the NNA suffers from the problem of local optima stagnation in solving some large scale optimization problems such as large-scale reliability system with higher dimensions and CEC (Congress of Evolutionary Computation) benchmark test problems. Secondly, the NNA experience is satisfactory in exploration but it fails to maintain a proper balance between exploration and exploitation due to lack of exploitation ability. Finally, it has slow convergence tendency and sometimes, it required more time to evaluate a new solution for some problems. These shortcomings will reduce its applications in some optimization problems with limitations and researchers have applied different search mechanisms and adopted modified operators to upgrade the original NNA. To mention a few- Zhang et al. [61] introduced a novel hybrid algorithm GNNA combining GWO and NNA for solving global numerical optimization problems. The core idea of this work is to make full use of good global search ability of NNA and fast convergence of GWO. A new hybrid TLNNA algorithm based on TLBO and NNA is developed by Zhang et al. [62] for solving engineering design optimization problems. In 2022, Kundu and Garg [63] proposed a new TLNNABC hybrid algorithm to solve reliability and engineering design optimization problems. In this algorithm, the structure of ABC algorithm has been improved by incorporating the features of the NNA and TLBO.

Teaching and learning are two common human social behaviours and are also an important motivating process in which an individual tries to learn from others. A regular
A improved neural network algorithm (INNA) is the main contribution of this paper can be stated as follows: and there is still much room left for future research. The enhancement. Finally, being a new optimization method compared to the conventional NNA by the population diversity shows that the performance of INNA is improved competition also help to identify how the proposed INNA performs and NNA. The measurement of exploration and exploitation for investigating the search behaviour of both INNA and NNA. The logarithmic search operator helps to accelerate the convergence speed of INNA, whereas the excellent global exploration ability of NNA helps to find a better global optimal solution and produces efficient and effective results for solving RRAP. In this study, a population diversity definition of the proposed method is also introduced and performed the exploration-exploitation evaluation for investigating the search behaviour of both INNA and NNA. The measurement of exploration and exploitation also help to identify how the proposed INNA performs better on an optimization problem. The experimental study shows that the performance of INNA is improved compared to the conventional NNA by the population diversity enhancement. Finally, being a new optimization method and there is still much room left for future research. The main contribution of this paper can be stated as follows:

- A improved neural network algorithm (INNA) is proposed by combining the features of TLBO and a new search operator. Proposed algorithm mainly contains the structure of the basic NNA and, meantime, it has been reconstructed by embedding the searching strategy of the learner phase of TLBO and a new logarithmic spiral search operator.
- The proposed method makes a proper balance between exploration and exploitation in which the basic NNA looks after the exploration part and the presence of a new search operator and the searching strategy of TLBO increases the exploitation capability of the algorithm.
- To validate the effectiveness and efficiency of the proposed INNA, it is examined against seven well-known RRAPs that includes series system, series-parallel system, complex (bridge) system, overspeed protection system, convex system, mixed series-parallel system, and large-scale system with dimensions 36, 38, 40, 42 and 50. Section 5 illustrates that the proposed method gives an effective result and also provides superior performance compared to other existing MHAs in terms of best optimal solutions and others.
- In order to check the statistical significance on the results obtained from the proposed INNA and the existing algorithms, a number of tests have been carried out, such as rank-tie, Wilcoxon sign-rank test, Kruskal–Wallis test, and multiple comparison tests. From these computed results, it is verified that the proposed algorithm produces an effective result and also outperforms other existing algorithms in terms of the best optimal solutions and the maximum possible improvement.

The rest of the papers is organized as follows: Sect. 2 briefly describes the basic NNA and TLBO algorithms. Section 3 presents a new logarithmic spiral search operator, the proposed algorithm INNA, and exploration-exploitation measurement. The RRAPs are described in Sect. 4. Section 5 presents the experimental results of the proposed INNA and compares them with several existing algorithms. The results obtained have also been validated through statistical test analysis. Finally, the conclusions and future works are drawn in Sect. 6.

2 Background

In this section, the basic NNA and the conventional TLBO algorithms are briefly described.

2.1 Neural network algorithm

NNA [60] is one of the newest MHAs and is followed by the nature of ANNs and biological nervous systems. In most of the cases, ANNs are used for making prediction and the main feature of this is to receive input data and predict the relationship between input and target data. Generally, the input values of the ANNs are obtained by experiments, calculations, and so on. Thus, it can be concluded that the ANNs are trying to map input data to the target data to minimize the error between the predicted solutions and the target solutions by iteratively changing weight function values. Although, the basic goal of the
Generate new solution: In NNA algorithm, a population matrix \( X = (x_{ij}) \) \((i = 1, 2, \ldots, N_p; j = 1, 2, \ldots, D)\) of size \( N_p \times D \) is randomly generated in the search space, where, \( N_p \) is the population size and \( D \) is the number of dimensions. In the algorithm, every individual of the \( N_p \) solutions are given by Eq. (1).

\[
x_i = [x_{i1}, x_{i2}, \ldots, x_{iD}] \quad \text{for} \quad i = 1, 2, \ldots, N_p \quad \text{(1)}
\]

Then for every individual of the population \( N_p \), a weight vector \( w_i = [w_{i1}, w_{i2}, \ldots, w_{iN_p}] \) is generated which satisfying Eq. (2)

\[
\sum_{j=1}^{N_p} w_{ij} = 1, \quad 0 \leq w_{ij} \leq 1, \quad i = 1, 2, \ldots, N_p \quad \text{(2)}
\]

After forming the weight matrix, a new solution has been generated using Eqs. (3) and (4), where \( t \) is the current number of iteration. Figure 1 describes the process of population generation in NNA.

\[
\begin{align*}
\chi_{\text{new},j}^{t+1} &= \sum_{i=1}^{N_p} w_{ij}^{t} \chi_i^{t}, \quad j = 1, 2, \ldots, N_p \\
\chi_i^{t+1} &= \chi_i^{t} + \chi_{\text{new},j}^{t+1}
\end{align*}
\]

2. Weight matrix update: Based on the best weight value known so-called “target weight”, the weight matrix can be updated according to Eq. (5)

\[
w_{ij}^{t+1} = w_{ij}^{t} + 2 \cdot d \cdot (w_{\text{Target}}^{t} - w_{ij}^{t}) \quad \text{for} \quad i = 1, 2, \ldots, N_p
\]

Here \( d \) is the random number between 0 and 1. The best solution obtained in each iteration is considered as the target solution \( x_{\text{Target}}^{t} \) and the associated weight is taken as the target weight vector \( w_{\text{Target}}^{t} \). The target weight vector \( w_{\text{Target}}^{t} \) and the target solution \( x_{\text{Target}}^{t} \) are updated simultaneously. If \( x_{\text{Target}}^{t} \) is equal to \( x_{k}^{t} (k \in [1, N_p]) \) at iteration \( t \), \( w_{\text{Target}}^{t} = w_{k}^{t} \).

3. Bias operator: The bias operator in the NNA helps to explore the search space (exploration process) and performs the same as the mutation operator in the GA. Generally, the bias operator restricts the algorithm from early convergence and customizes a number of individuals in the population. In the bias operator, the modification factor \( \beta \) is initially set to 1 (i.e. 100 per cent of the chance to reconstruct all individuals in the population) and its value has been flexibly reduced at each iteration using the following reduction formula given in Eq. (6).

\[
\beta^{t+1} = 0.99 \cdot \beta^{t}
\]

In NNA, there are two parts in bias operator named bias of population and bias of weight matrix. Firstly, a random number \( \Lambda_{p}^{\text{Bias}} \) is generated for bias of population according to the Eq. (7)

\[
\Lambda_{p}^{\text{Bias}} = [\beta \times D] \quad \text{(7)}
\]

Then a set \( \text{Pop} \) is introduced by randomly selecting \( \Lambda_{p}^{\text{Bias}} \) integers between 0 and \( D \). Let, \( \text{lb} = (lb_1, lb_2, \ldots, lb_{N_p}) \) and \( \text{ub} = (ub_1, ub_2, \ldots, ub_{N_p}) \) represents the lower and the upper bounds of design variables respectively. Then the bias of population is described by the following Eq. (8)
where $r$ is a random number between 0 and 1. Again, in case of bias weight matrix, a random number $N^w_{Bias}$ is firstly generated, that is equal to $[\beta \times N_p]$. Then a new set $T$ is formed consisting $N^w_{Bias}$ randomly selected integers between 0 and $N_p$ and the bias weight matrix is defined according to Eq. (9)

$$w^l_{i,t} \sim U(0, 1), \quad \text{for } l = 1, 2, \ldots, N^w_{Bias}$$

where $U(0, 1)$ is a random number between 0 and 1.

4. Transfer operator: In the NNA, the transfer function operator is transferring the current solution to a new position to upgrade and develop better quality solutions to the current best solution. Improvement of the solutions is achieved by moving current new pattern solutions closer to the target solution, which is expressed as the following Eq. (10)

$$x^{t+1}_i = x^t_i + 2 \cdot r \cdot (x^t_{Target} - x^t_i) \quad \text{for } i = 1, 2, \ldots, N_p$$

2.2 Teaching–learning-based optimization (TLBO)

Like other population-based algorithms, Rao et al. [49] proposed an algorithm called TLBO based on the conventional teaching–learning aspects of a classroom. In TLBO, a group of learners is recognised as the population and various subjects taught to learners represents different design variables. The fitness value indicates the students grade after learning and the student with the best fitness value is witnessed as the teacher. This algorithm describes two basic modes of learning: (1) through teacher (known as teacher phase) and (2) interacting with the other learners (known as the learner phase). The working procedure of the TLBO algorithm is explained below -

2.2.1 Teacher phase

In the teacher phase, let us assume, at any iteration $t$, the number of subjects or course offered to the learners is $D$ and $N_p$ denotes the population size (i.e. number of learners).

---

**Neural network algorithm (NNA)**

1. Initialization: Maximum Iterations, Population size ($N_p$)
2. Initialize a random population $x_i$ and the weight matrix $w_i$ (for $i = 1, 2, \ldots, N_p$)
3. Evaluate the fitness value of the population and select the optimal solution and optimal weight
4. repeat
5. Generate new populations $x^{t+1}$ using equations (3) and (4), and update the weight matrix $w^{t+1}$ via equation (5)
6. for each individual $i \in N_p$
7. Generate a random number $r \in [0, 1]$
8. if $r \leq \beta^t$
9. A random number $N^w_{Bias} = [\beta \times D]$ is generated
10. for $i = 1$ to $N^w_{Bias}$
11. update the solution $x^{t+1}_i$ using equation (8) \(\triangleright\) Perform bias population
12. end
13. A random number $N^w_{Bias} = [\beta \times N_p]$ is generated
14. for $i = 1$ to $N^w_{Bias}$
15. update the weight $w^{t+1}_i$ using equation (9) \(\triangleright\) Perform bias weight
16. end
17. else
18. update the solution using equation (10) \(\triangleright\) Perform transfer operator
19. end if
20. end for
21. Update the modification factor $\beta^{t+1}$ using equation (6)
22. Calculate the fitness value for each solution and select the optimal solution $x^{t+1}_{Target}$ and optimal weight $w^{t+1}_{Target}$
23. Until (stop condition = false)
24. Post process results and visualization

Fig. 2 Pseudo-code for NNA
learners). In this phase, the basic intention of a teacher is to transfer knowledge among the students and also to improve the average result of the class. Here, the parameter \( \text{Mean}_j(t) \) indicates the mean result of the learners in \( j \)th subject \( (j = 1, 2, \ldots, D) \) and at generation \( t \), it is given by Eq. (11).

\[
\text{Mean}_j(t) = [M'_1, M'_2, \ldots, M'_D]
\] (11)

Let \( X_\text{Teacher}(t) \) indicates the learner with the best objective function value at iteration \( t \) and is recognised as the teacher. The teacher tries to give his/her maximum effort to increase the knowledge of each student in the class, but learners will gain knowledge according to their talent and also by the quality of teaching. Then, the difference vector between the teacher and the average results of students can be calculated given by the Eq. (12).

\[
G_{ij}^\text{Mean}(t) = c \times [X_\text{Teacher}(t) - T_F \cdot \text{Mean}_j(t)]
\] (12)

where \( c \) indicates a random number lies between 0 and 1, \( T_F \) denotes the teaching factor and its value can be 1 or 2. Based on the \( G_{ij}^\text{Mean}(t) \), the existing solution of the population can be updated and is given by the following Eq. (13)

\[
X_{ij}^\text{new}(t) = X_{ij}(t) + G_{ij}^\text{Mean}(t)
\] (13)

The new solution \( X_{ij}^\text{new}(t) \) in generation \( t \) is accepted if it found to be a better than the previous one.

### 2.2.2 Learner phase

In addition to learning from the teacher, interaction with other students is also an effective way to enhance their knowledge. A learner can also gain new information from other learners having more knowledge than him or her. In this phase, a student \( X_p \) randomly select classmate \( X_q \) \( (\neq X_p) \) to obtain more knowledge. If \( X_p \) performs better, \( X_p \) moves towards \( X_q \); otherwise, moves away from it. The following formulas (14) and (15) can be used to describe this process:

\[
X_{p,ij}^\text{new}(t) = X_{p,ij}(t) + m \times (X_{q,ij}(t) - X_{p,ij}(t)),
\]

if \( f(X_{q,ij}(t)) < f(X_{p,ij}(t)) \) (14)

\[
X_{p,ij}^\text{new}(t) = X_{p,ij}(t) + m \times (X_{p,ij}(t) - X_{q,ij}(t)),
\]

if \( f(X_{q,ij}(t)) > f(X_{p,ij}(t)) \) (15)

Where \( m \) is a random number between 0 and 1 and \( f(X_{p,ij}(t)) \) and \( f(X_{q,ij}(t)) \) are fitness values of \( X_{p,ij}(t) \) and \( X_{q,ij}(t) \) respectively. The pseudocode of the basic TLBO is given in Fig. 3.

### 2.3 Constraint handling technique

The standard form of the constrained optimization problem is formulated as follows:

\[
\text{Optimize } f(x)\quad \text{subject to } g_i(x) \leq 0, \quad i = 1, 2, \ldots, M
\]

\[
x^L_k \leq x_k \leq x^U_k, \quad k = 1, 2, \ldots, n
\]

where \( f(x) \) is the objective function, \( g_i(x) \) is the set of inequality and equality constraints and \( x = [x_1, x_2, \ldots, x_n] \) is the decision variables. \( x^L_k \) and \( x^U_k \) are the lower and upper limits respectively for each \( x_k, (k = 1, 2, \ldots, n) \) defined in the search space \( S \).

Often, it is very difficult to obtain the optimal solution of the optimization problems by a MHA because of the presence of constraints in it, and also some new solutions generated by these methods may be infeasible which violates some constraints. To overcome this situation, a penalty function method is introduced, which can convert a constrained optimization problem to an unconstrained optimization problem and as a result, a global feasible solution can be achieved in an equitable time. The basic goal of a penalty function is to penalize the infeasible solutions. In this study, an exterior penalty function method is used to penalize the infeasible solutions by penalizing the objective value. Dealing with this penalty function, the maximum constrained problem \( f(x) \) can be converted into minimum problem \( F(x) \) as follows

\[
\text{minimize } F(x) = -f(x) + \lambda \sum_{j \in S} \max(0, g_j(x))
\]

Here, \( \lambda \) represents the penalty coefficient.

### 2.4 Exploration and exploitation measurement

In this study, an in-depth empirical analysis is performed to examine the searching behaviour of the proposed INNA in terms of diversity. Through diversity measurement, it is possible to measure explorative and exploitative capabilities of the algorithm. In the exploration phase, the difference expands between the values of dimension \( D \) within the population and hence swarm individuals are scattered in the search space. On the other hand, in exploitation phase, the difference reduces and swarm individuals are clustered to a dense area. These two concepts are ubiquitous in any MHAs. In case of finding the globally optimal location, the exploration phase maximizes the efficiency in order to visit unseen neighbourhoods in the search space. Contrarily, through exploitation, an algorithm can successfully converge to a neighbourhood with high possibility of global optimal solution. A proper balance between
Teaching-Learning based optimization (TLBO)

1. **Initialization**: Maximum Iterations, Population size ($N_p$)
2. Initialize a random population including $N_p$ solutions $X_i$ for $i = 1, 2, \ldots, N_p$
3. Calculate the fitness value of each solution and set the optimal solution as the teacher ($X_{Teacher}$).
4. **repeat**
5. **for** each individual $i \in N_p$
6. Calculate the mean result according to equation (11).
7. Update the solution using equations (12) and (13). ▶ Perform Teacher phase
8. Calculate the fitness value of the new solution $X_{i,new}^t$.
9. Accept the new solution if it gives a better result than the old one.
10. **end for**
11. **for** each individual $i \in N_p$
12. Update the solution using equations (14) and (15). ▶ Perform Learner phase
13. Calculate the fitness value of the new solution $X_{i,new}^t$.
14. Accept the new solution if it gives a better result than the old one.
15. **end for**
16. **Until** (stop condition = false)
17. Post process results and visualization

Fig. 3 Pseudo-code for TLBO

Fig. 4 Candidate population representation for exploration-exploitation

\[ Div_j(t) = \frac{\sum_{i=1}^{N_p} [med(x_j(t)) - x_{ij}(t)]}{N_p} \]  \hspace{1cm} (17)

\[ Div(t) = \frac{\sum_{j=1}^{D} Div_j(t)}{D} \]  \hspace{1cm} (18)

Where, $x_{ij}(t)$ denotes the $j$-th dimension of $i$-th swarm individual in $N_p$ population in iteration $t$, whereas $med(x_j(t))$ is median of dimension $j$. $Div_j(t)$ and $Div(t)$ indicates the diversity in the $j$-th dimension and the average of diversity of all dimensions respectively. After determining the population diversity $Div$ for all the iterations, it is now possible to calculate the exploration and
exploitation percentage ratios during search process, using Eqs. (19) and (20) respectively:

\[
\text{Expl}(\%) = \frac{\text{Div}(t)}{\text{Div}_{\text{max}}(t)} \times 100
\]  

\[
\text{Expt}(\%) = \frac{|\text{Div}(t) - \text{Div}_{\text{max}}(t)|}{\text{Div}_{\text{max}}(t)} \times 100
\]

where \( \text{Expl}(\%) \) and \( \text{Expt}(\%) \) denotes exploration and exploitation percentages respectively for iteration \( t \), whereas \( \text{Div}_{\text{max}}(t) \) is the maximum population diversity in all iterations \( T \).

3 Proposed method: INNA

This section is divided into three subsections. Firstly, a new logarithmic spiral search operator is introduced in Sect. 3.1. The design structure and the implementation of INNA are described in Sect. 3.2. Finally, exploration and exploitation measurement of the proposed INNA is discussed.

3.1 New search operator

In original NNA, all the individuals followed the same direction pattern which leads some individuals to move aside from the promising region of the search space and as a result, the convergence rate of the NNA decreases. Therefore, we proposed a new logarithmic spiral search operator in our proposed algorithm (INNA) to overcome this issue and that can be expressed by the Eq. (21).

\[
X_{t+1}^i = |X_i^t - X_{\text{Target}}^t| \cdot e^{\beta \theta} \cdot \cos(2\pi \theta) + X_{\text{Target}}^t
\]

where \( \beta \) is a constant and taken as 1. This parameter controls the specific shape of the spiral. In addition, \( \theta = 2(1 - \frac{t}{T}) - 1 \), is a parameter that decreases linearly from 1 to \(-1\) as the number of iteration increases, where \( t \) and \( T \) indicates the current iteration and the maximum number of iterations, respectively. The updated positions of obtained solutions in each iteration for the logarithmic spiral search model are described in Fig. 5 and it is clear from that figure, as the value of \( \theta \) switches from 1 to \(-1\), current solutions pointedly move closer to the target solution. The exploitation capability of the algorithm is highlighted, and the convergence speed is further enhanced.

3.2 The proposed INNA

The detailed framework of the proposed algorithm INNA is demonstrated in this section. The traditional NNA is simple and effective swarm optimization technique that has been used to solve various kinds of practical optimization problems. Benefiting from the unique structure of artificial neural networks, NNA has a great exploration ability. Despite its strong global search ability, NNA has a noticeable deficiency being its slow convergence speed as it fails to manage the convenient balance between exploitation and exploration and these shortcomings will also reduce its applications in some optimization problems with limitations. To get rid of these types of insufficiency, the updating phase of the solution is enhanced by reconstructing the basic formation of the NNA. During this
modification, the searching mechanism of the learner phase of TLBO (Eqs. (14) and (15)) and a new logarithmic spiral search operator (Eq. (21)) are implemented into the main structure of the NNA. Further, $CP_1$ and $CP_2$ are two predetermined parameters lie in the range $[0,1]$ which are introduced in our proposed INNA to control the probability of selecting the above searching strategies. The TLBO algorithm having first convergence speed and much better computational complexity than several existing algorithms makes it an exceptional search algorithm. Thus, the inclusion of TLBO and the new search operator adds more flexibility to the NNA and subsequently, the exploration and exploitation abilities of the NNA algorithm are also improved. Therefore, in the search process of INNA, TLBO and the logarithmic spiral search operator aims on improved. Therefore, in the search process of INNA, the local search and NNA accentuate on the global search, and the searching mechanism of the learner phase (5).

**Step 3:** The fitness value of each individual of the population is evaluated and the best one i.e., the optimal solution and the optimal weight is selected.

**Step 4:** In this step, a new solution is generated and the weight matrix is updated through Eqs. (3)–(5).

**Step 5:** A random number is generated and if it is less than $\beta^t$, perform bias operator to update the current solution. Otherwise, depending on the controlling parameters $CP_1$ and $CP_2$, current solution is updated either using the searching operator of TLBO, or via logarithmic spiral search operator or using the transfer operator.

**Step 6:** In this step, $\beta^t$ is updated and the current population is evaluated. Then, the greedy search mechanism is performed and the optimal solution $x^{t+1}_{\text{Target}}$ and the optimal weight $w^{t+1}_{\text{Target}}$ are selected.

**Step 7:** Go to step 3 if the termination criterion is not satisfied, otherwise stop the process.

The pseudo-code and the detailed flowchart of the proposed INNA has been shown in Figs. 6 and 7 respectively.

To evaluate the computational complexity of the proposed INNA algorithm, the complexity of the algorithm is calculated according to the worst-case complexity. Thus, Big-O notation is used here as a common terminology. Complexity is dependent upon population size ($N_p$), dimensions ($D$) and the maximum number of iterations ($T$). In the initialization phase, the computational complexity of INNA is $O(N_p)$ after initializing the population of $N_p$ individuals. After that, the fitness of each individual is evaluated in the main loop of the INNA algorithm, so the computational complexity in this stage becomes $O(T \cdot N_p)$. Finally, the current position of each search agent is updated via different searching strategy in the population update stage, so the computational complexity in this stage is $O(T \cdot N_p \cdot D)$. After complete analysis, the computational complexity of INNA is calculated as follows:

$$O(\text{INNA}) = O(\text{Initialization}) + O(\text{Fitness evaluations}) + O(\text{Population update})$$

i.e.,

$$O(\text{INNA}) = O(N_p) + O(T \cdot N_p) + O(T \cdot N_p \cdot D) = O(T \cdot N_p \cdot D).$$

### 4 Problem formulation

In the present-day scenario, the demands for highly reliable products and equipment are increasing day by day. Therefore, in recent years, the requirement of reliability analysis to evaluate the performance of products, equipment, and several engineering systems is also increasing. Reliability optimization can figure out these issues and capable of finding a high-quality system that performs efficiently and safely in a given period. In this section,
Improved NNA

begin
1. Initialization: Maximum Iterations, Population size ($N_p$)
2. Initialize a random population $x_i^t$ and the weight matrix $w_i^t$ (for $i = 1, 2, ..., N_p$)
3. Evaluate the fitness value of the population and select the optimal solution and optimal weight
4. while ($t < T$)
5. Generate new populations $x_i^{t+1}$ using equations (3) and (4), and update the weight matrix $w_i^{t+1}$ via equation (5)
6. for each individual $i \in N_p$
7. Generate a random number $r \in [0, 1]$
8. if $r \leq \beta^t$
9. Perform bias operator for updating the solutions $x_i^{t+1}$ and the weight $w_i^{t+1}$ via equations (8) and (9) respectively.
10. else
11. Generate a random number $r_1 \in [0, 1]$
12. if $r_1 < CP_1$
13. Update the solution using the learner phase of TLBO i.e., via equations (14) and (15)
14. else
15. Generate a random number $r_2 \in [0, 1]$
16. if $r_2 < CP_2$
17. Update the solution using logarithmic spiral search operator via equation (21)
18. else
19. Update the solution using equation (10)
20. end if
21. end if
22. end if
23. end for
24. Update the modification factor $\beta^{t+1}$ using equation (6)
25. Calculate the fitness value for each solution and select the optimal solution $x_{Target}^{t+1}$ and optimal weight $w_{Target}^{t+1}$
26. Return the individual with best objective value
end

Fig. 6 Pseudo-code for the proposed INNA

Fig. 7 Flowchart of the proposed INNA
RRAPs are considered to explore the performance of the INNA algorithm. Before introducing the RRAPs, we define the following assumptions:

4.1 Assumptions

(1) The failure of a component of any subsystem is independent of that of others i.e., the entire system will not be damaged.
(2) There is only two states of the components and the system i.e., operating or failure.
(3) All redundancy are considered as active redundancy and are not repaired.
(4) Component associates like reliability, weight, cost, volume etc. are fixed.

The system consists of \( m \) subsystems and in each subsystem \( i \), the least number active components required to function is given by \( n_i^{\text{min}} \) which constitutes the pre-specified lower bound of the redundancy level for that particular subsystem. On the other hand, the upper bound of the redundancy level for the \( i \)-th subsystem is denoted by \( n_i^{\text{max}} \), which is either supplied in advance or can be produced by solving the system constraints, if linear. The goal of the problem is to maximize system reliability by computing the number of redundant components \( n_i \) and the components reliability \( r_i \) in each subsystem satisfying the given resource constraints. The general form of the reliability–redundancy problem can be formulated as the non-linear integer-programming problem given by Eq. (24).

\[
\begin{align*}
\text{max} & \quad R_S(r_1, r_2, \ldots, r_m; n_1, n_2, \ldots, n_m) \\
\text{s.t.} & \quad g_k(n) = \sum_{i=1}^m g_{ki}(n) \leq b_k, \quad \text{for} \ k = 1, 2, \ldots, l \\
0.5 \leq r_i & \leq 1, \quad n_i^{\text{min}} \leq n_i \leq n_i^{\text{max}}, \quad i = 1, 2, \ldots, m. \\
\end{align*}
\]

In this study, seven benchmark problems of the RRAP have been considered such as series system, series-parallel system, complex (bridge) system, overspeed protection system, convex quadratic system, mixed series-parallel system and large scale system with dimensions 36, 38, 40, 42 and 50. All the above problems are shown to maximize the system’s reliability under different non-linear constraints and can be stated as the mixed integer linear problems. For each problem both the component reliability and the redundancy allocation are to be examined simultaneously and are formulated as follows.

**P1. Series System** [Fig. 8a] The series system is a nonlinear mixed-integer programming problem, which has been used in [14, 17, 18, 22, 70–73]. The problem formulation is given as follows:

\[
\text{max} \quad R_S(r_1, r_2, \ldots, r_m; n_1, n_2, \ldots, n_m) \\
\text{s.t.} \quad g_k(n) = \sum_{i=1}^m g_{ki}(n) \leq b_k, \quad \text{for} \ k = 1, 2, \ldots, l \\
0.5 \leq r_i \leq 1, \quad n_i^{\text{min}} \leq n_i \leq n_i^{\text{max}}, \quad i = 1, 2, \ldots, m. 
\]
The parameters \(a\) and \(\beta_i\) respectively. And also, subject to, the same constraint given by the Eqs. (25), (26) and (27) respectively. And also, 0.5 \(\leq r_i \leq 1, 1 \leq n_i \leq 5, n_i \in Z^+, i = 1, 2, \ldots, 5\). The coefficients of the series-parallel system are shown in the literature [18] and Table 2.

### Table 2 Values of parameters used in the literature

| Parameter used for P1 and P3 | 10^3 z_i | \(\beta_i\) | v_i | w_i | C | V | W |
|-------------------------------|--------|-------|---|---|---|---|---|
| 1                             | 2.330  | 1.5   | 1 | 7 | 175 | 110 | 200 |
| 2                             | 1.450  | 1.5   | 2 | 8 |      |     |     |
| 3                             | 0.541  | 1.5   | 3 | 8 |      |     |     |
| 4                             | 8.050  | 1.5   | 4 | 6 |      |     |     |
| 5                             | 1.950  | 1.5   | 2 | 9 |      |     |     |

| Parameter used for P2         | 10^3 z_i | \(\beta_i\) | v_i | w_i | C | V | W |
|-------------------------------|--------|-------|---|---|---|---|---|
| 1                             | 2.500  | 1.5   | 2 | 3.5 | 175 | 180 | 100 |
| 2                             | 1.450  | 1.5   | 4 | 4.0 |      |     |     |
| 3                             | 0.541  | 1.5   | 5 | 4.0 |      |     |     |
| 4                             | 0.541  | 1.5   | 8 | 3.5 |      |     |     |
| 5                             | 2.100  | 1.5   | 4 | 3.5 |      |     |     |

| Parameter used for P4         | 10^3 z_i | \(\beta_i\) | v_i | w_i | C | V | W |
|-------------------------------|--------|-------|---|---|---|---|---|
| 1                             | 1.0    | 1.5   | 1 | 6 | 400 | 250 | 500 |
| 2                             | 2.3    | 1.5   | 2 | 6 |      |     |     |
| 3                             | 0.3    | 1.5   | 3 | 8 |      |     |     |
| 4                             | 2.3    | 1.5   | 2 | 7 |      |     |     |

The coefficients of the series-parallel system are shown in the literature [18] and Table 2.

### P3. Complex (bridge) system [Fig. 8c]

Complex (bridge) system consists of five subsystems, which is a classical reliability-redundancy problem, and it has been investigated in [14, 17, 18, 21, 25, 70–72, 74, 77, 79, 80]. The formulation of the complex (bridge) is described as follows:

\[
\max R_5(r, n) = \prod_{i=1}^{5} R_i
\]

s.t. \(g_1(n) = \sum_{i=1}^{5} v_i n_i^2 - V \leq 0\),

\[
g_2(n) = \sum_{i=1}^{5} z_i \left( -1000 \frac{\beta_i}{ln(r_i)} \right) \left[ n_i + \exp \left( \frac{n_i}{4} \right) \right] - C \leq 0, \tag{26}
\]

\[
g_3(n) = \sum_{i=1}^{5} w_i n_i \exp \left( \frac{n_i}{4} \right) - W \leq 0, \tag{27}
\]

\[0.5 \leq r_i \leq 1, 1 \leq n_i \leq 5, n_i \in Z^+, i = 1, 2, \ldots, 5\]

The parameters \(z_i\) and \(\beta_i\) are physical features of system components. Constraints \(g_1(n), g_2(n),\) and \(g_3(n)\) represents volume, cost and weight constraint respectively. The coefficients of the series system are shown in the literature [18] and Table 2.

### P2. Series-Parallel System [Fig. 8b]

The Series-parallel system has been studied in many recent publications study such as [14, 17–19, 21, 22, 70–79]. The mathematical formulation is as follows:

\[
\max R_5(r, n) = 1 - (1 - R_1 R_2) (1 - (1 - R_3)(1 - R_4) R_5)
\]

subject to, the same constraint given by the Eqs. (25), (26) and (27) respectively. And also, \(0.5 \leq r_i \leq 1, 1 \leq n_i \leq 5, n_i \in Z^+, i = 1, 2, \ldots, 5\). The coefficients of the complex system are shown in the literature [18] and Table 2.

### P4. Overspeed protection system for a gas turbine [Fig. 8d]

The fourth problem is considered for the RRAP of the Overspeed protection system for a gas turbine. Overspeed detection is continuously provided by the electrical and mechanical systems. It is necessary to cut off the fuel supply in case of an overspeed occurs and thus, 4 control valves (V1-V4) must be closed. The control system is modeled as a 4-stage series system. This problem has been considered in [14, 17, 18, 21, 71, 72, 76, 77, 79, 80]. This reliability problem is formulated as follows:
max \( R_5(r, n) = \prod_{i=1}^{\frac{4}{2}} R_i \)

s.t., \( g_1(n) = \sum_{i=1}^{\frac{4}{2}} v_i n_i^2 - V \leq 0, \)

\( g_2(n) = \sum_{i=1}^{\frac{4}{2}} g_i \left( \frac{-1000}{\ln(r_i)} \right) b_i [n_i + \exp \left( \frac{n_i}{4} \right)] - C \leq 0, \)

\( g_3(n) = \sum_{i=1}^{\frac{4}{2}} w_i n_i \exp \left( \frac{n_i}{4} \right) - W \leq 0 \)

\( 0.5 \leq r_i \leq 1, 1 \leq n_i \leq 5, \quad n_i \in \mathbb{Z}^+ \), \( i = 1, 2, \ldots, 5 \)

The coefficients of the overspeed protection system are shown in the literature [18] and Table 2.

**P5. Convex quadratic reliability problem** This problem is an integer programming with convex quadratic constraints, which has been investigated by [14, 77, 81, 82]. The detailed mathematical formulation of this problem is as follows:

max \( R_5(r, n) = \prod_{i=1}^{\frac{10}{2}} (1 - (r_i)_i^n) \)

s.t., \( g_i(n) = \prod_{i=1}^{\frac{4}{2}} (a_j n_j^2 + C_j n_i) \leq b_j \)

\( n_i \in [1, 6], j = 1, 2, \ldots, 10 \), \( i = 1, 2, 3, 4 \)

The parameters \( r_i, a_j \) and \( C_j \) are generated from uniform distributions that lies between [0.80, 0.99], [0.10] and [0.01] respectively. A randomly generated set of values of these coefficients are given as follows: \( r_i = [0.81, 0.93, 0.92, 0.96, 0.99, 0.89, 0.85, 0.83, 0.94, 0.92]; \)

\( b_j = (2.0 \times 10^{15}, \ 3.1 \times 10^{12}, \ 5.7 \times 10^{13}, \ 9.3 \times 10^{12}); \)

\( a = \begin{pmatrix} 2 & 7 & 3 & 0 & 5 & 6 & 9 & 4 & 8 & 1 \\ 4 & 9 & 2 & 7 & 1 & 0 & 8 & 3 & 5 & 6 \\ 5 & 1 & 7 & 4 & 3 & 6 & 0 & 9 & 8 & 2 \\ 8 & 3 & 5 & 6 & 9 & 7 & 2 & 4 & 0 & 1 \end{pmatrix}; \)

\( C = \begin{pmatrix} 7 & 1 & 4 & 6 & 8 & 2 & 5 & 9 & 3 & 3 \\ 4 & 6 & 5 & 7 & 2 & 6 & 9 & 1 & 0 & 8 \\ 1 & 10 & 3 & 5 & 4 & 7 & 8 & 9 & 4 & 6 \\ 2 & 3 & 2 & 5 & 7 & 8 & 6 & 10 & 9 & 1 \end{pmatrix} \)

**Table 4** Available system resources for each system for P7

| \( n \) | \( i \) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|------|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 36   | \( b_i \) | 391 | 257 | 738 | 1454 |
| 38   | \( b_i \) | 416 | 278 | 778 | 1532 |
| 40   | \( b_i \) | 435 | 289 | 823 | 1621 |
| 42   | \( b_i \) | 458 | 306 | 870 | 1712 |
| 50   | \( b_i \) | 543 | 352 | 1040 | 2048 |

**P6. Mixed series-parallel system** The mixed series-parallel system is studied in [14, 77, 81, 82] and formulated as follows.

max \( R_5(r, n) = \prod_{i=1}^{\frac{15}{2}} (1 - (1 - r_i)_i^m) \)

s.t., \( g_1(n) = \sum_{i=1}^{\frac{15}{2}} c_i n_i - 400 \leq 0 \)

\( g_2(n) = \sum_{i=1}^{\frac{15}{2}} w_i n_i - 414 \leq 0 \)

\( n_i \geq 1, \quad n_i \in \mathbb{Z}^+, \quad i = 1, 2, \ldots, 15. \)

The coefficients of the mixed series-parallel system are taken from the literature [12] and are listed in Table 3.

**P7. Large-scale system reliability problem** Large-scale system reliability problem has been studied by [17, 26, 70, 73, 77, 78, 81] and the detailed mathematical formulation of this problem is as follows.

max \( R_S(r, n) = \prod_{i=1}^{\frac{m}{2}} R_i \) \( (28) \)

s.t., \( g_1(n) = \sum_{i=1}^{\frac{m}{2}} \alpha_i n_i^2 - \left( 1 + \frac{\theta}{100} \right) \sum_{i=1}^{\frac{m}{2}} \alpha_i l_i^2 \leq 0 \) \( (29) \)

\( g_2(n) = \sum_{i=1}^{\frac{m}{2}} \beta_i \exp \left( \frac{n_i}{2} \right) - \left( 1 + \frac{\theta}{100} \right) \sum_{i=1}^{\frac{m}{2}} \beta_i \exp \left( \frac{l_i}{2} \right) \leq 0 \) \( (30) \)

\( g_3(n) = \sum_{i=1}^{\frac{m}{2}} \gamma_i n_i - \left( 1 + \frac{\theta}{100} \right) \sum_{i=1}^{\frac{m}{2}} \gamma_i l_i \leq 0 \) \( (31) \)

**Table 3** Parameter used for P6

| \( i \) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| \( r_i \) | 0.90 | 0.75 | 0.65 | 0.80 | 0.85 | 0.93 | 0.78 | 0.66 | 0.78 | 0.91 | 0.79 | 0.77 | 0.67 | 0.79 | 0.67 |
| \( c_i \) | 5 | 4 | 9 | 7 | 7 | 5 | 6 | 9 | 4 | 5 | 6 | 7 | 9 | 8 | 6 |
| \( w_i \) | 8 | 9 | 6 | 7 | 8 | 8 | 9 | 6 | 7 | 8 | 9 | 7 | 6 | 5 | 7 |
\[
g(n) = \sum_{i=1}^{m} \delta_i \sqrt{n_i} - \left(1 + \frac{\theta}{100}\right) \sum_{i=1}^{m} \delta_i \sqrt{\bar{d}_i} \leq 0
\]

Here, \( \bar{d}_i \) indicates the lower bound of \( n_i \). The parameter \( \theta \) indicates the tolerance error that implies 33% of the minimum requirement of each available resource \( l_i \). The average minimum resource requirements for the reliability system with \( m \) subsystems is given by \( \sum_{i=1}^{m} g_\beta(l_i) \). In this way, we set the available system resources [26] for reliability systems with 36, 38, 40, 42, and 50 subsystems, respectively, as shown in Tables 4 and 5.

### 5 Experimental results and discussions

This section presents the results of all of the above-mentioned reliability optimization problems analysed by the proposed INNA algorithm. It is divided into the following four parts. Section 5.1 introduces the experiment settings including parameters settings and maximum possible improvement (MPI). Section 5.2 describes the results obtained by the proposed algorithm and compared the performance with a number of existing approaches that are presented Table 6. The performance in terms of population diversity and the exploration-exploitation measurement of INNA and the conventional NNA are described in Sect. 5.3. Finally, the statistical analysis of the proposed algorithm and all compared algorithms are illustrated in Sect. 5.4.

#### 5.1 Experiment settings

##### 5.1.1 Parameter settings

The proposed algorithm is implemented in MATLAB (2015a) on the personal laptop with AMD Ryzen 3 2200 U with Radeon Vega Mobile Gfx 2.50GHz and 4.00 GB of RAM in Windows 10. In this study, to compare the results by INNA statistically with the other existing optimizers like, ABC, NNA, TLBO, SSA, HHO, SMA, and SCA, the initial population sizes were set as 100 for each algorithm and also the parameters of these compared algorithms are considered as: ABC (Maximum number of trials i.e., limit = 100), NNA (modification factor, \( \beta = 1 \)), TLBO (teaching factor, TF = 1 or 2), HHO (\( \beta = 1.5 \)), SMA (control parameter, \( z = 0.03 \)), SCA (parameter, \( a = 2 \)); Due to the stochastic nature of metaheuristics algorithms, it might be unreliable if one considers the results obtained in a single run. Therefore, 30 independent runs were performed for all...
applied algorithms ABC, NNA, TLBO, SSA, HHO, SMA, SCA and INNA for solving every reliability optimization problems. In our experiment, for every independent run, the maximum number of iterations for each algorithm is taken as 1500.

5.1.2 Maximum possible improvement (MPI)

For each reliability optimization problem, the system reliability is to be maximized by computing both the components reliability \( r_i \) and the number of redundant components \( n_i \) for each subsystem. During the computational procedure, the redundant components \( n_i \) are firstly considered as real variables and after completion of the optimization process, the real values are converted to their respective nearest integer values. In this study, we introduce the maximum possible improvement (MPI) index to evaluate the performance of INNA and is expressed by the Eq. (33).

\[
\text{MPI} = \frac{R_S \text{ (INNA)} - R_S \text{ (Others)}}{1 - R_S \text{ (Others)}}
\]  

(33)

Where \( R_S \text{ (INNA)} \) denotes the best optimal solution obtained by the proposed algorithm and \( R_S \text{ (Others)} \) implies the best result obtained by the other compared approaches and the greater MPI indicates greater improvement.

5.2 INNA comparison with existing optimizers

This section describes the performance evaluation of proposed INNA in terms of best solution and the maximum possible improvement value. The results obtained by the proposed algorithm is compared with the other existing optimizers and the results of the compared algorithms are taken from their respective papers. The comparative analysis for solving the reliability problems are presented in Tables 7, 8, 9, 10, 11, 12.

For the series system (P1), Table 7 shows that the best optimal solution obtained by the proposed INNA is 0.931682387907051, which is preferable to all compared algorithms GA [14], SCA [70], SAA [72], IA [22], ABC1 [79], IPSO [17], CS2 [18], PSO [71], NAFSA [84], SSO [71], PSSO [71], MICA [85] with the improvements 3.24E-01 %, 3.50E-03 %, 4.65E-01 %, 7.01E-05 %, 5.68E-04 %, 3.50E-03 %, 4.13E-04 %, 3.87E+01 %, 1.76E-04 %, 2.63E-01 %, 1.33E-04 %, and 4.39E-03 % respectively.

Table 7 Some existing meta-heuristic algorithms for solving reliability optimization problems

| Algorithms | Methods | Authors and published year |
|------------|---------|-----------------------------|
| 1. SCA     | Soft computing approach | Gen and Yun [70] |
| 2. SAA     | Simulated annealing algorithm | Kim et al. [72] |
| 3. GA      | Genetic algorithm (GA) | Yokota et al. [14] |
| 4. IA      | Immune based two-phase approach | Hsieh and You [22] |
| 5. ABC1    | Artificial bee colony algorithm | Yeh and Hsieh [79] |
| 6. IPSO    | Improved particle swarm optimization | Wu et al. [17] |
| 7. CS1     | Cuckoo search (CS) algorithm | Valian and Valian [73] |
| 8. CS2     | Cuckoo search algorithm | Garg [18] |
| 9. PSO/SSO/PSSO | Particle-based swarm optimization algorithm | Huang [71] |
| 10. ICS    | Improved CS algorithm | Valian et al. [78] |
| 11. CS-GA  | Hybrid CS and genetic algorithm | Kanagaraj et al. [19] |
| 12. ABC2   | Artificial bee colony | Garg et al. [74] |
| 13. INGHS  | Improved novel global harmony search | Ouyang et al. [83] |
| 14. MPSO   | Modified particle swarm optimization | Liu and Qin [76] |
| 15. EBBBO  | Efficient biogeography-based optimization | Garg [21] |
| 16. EGHS   | Effective global harmony search algorithm | Zou et al. [25] |
| 17. NMDE   | Novel modified DE | Zou et al. [80] |
| 18. NGHS   | Novel global HS algorithm | Zou et al. [26] |
| 19. CPSO   | Co-evolutionary PSO | He and Wang [75] |
| 20. IABC   | Improved ABC algorithm | Ghambari and Rahati [81] |
| 21. NAFSA  | Novel artificial fish swarm algorithm | He et al. [84] |
| 22. MICA   | Modified imperialist competitive algorithm | Afonso et al. [85] |
algorithms SAA [72], IPSO [17], CPSO [75], CS1 [73], ICS [78], CS-GA [19], ABC2 [74], MPSO [76], INGHS [77], CS2 [18], DE [86], EBBO [21], and NAFSA [84], the proposed INNA gives better results with MPI 33.3% for each. Further, the optimal redundant component by INNA for this problem P2 is (3,2,2,4) which is completely different from the other approaches. It can be observed from Table 9 that the optimal solution for the complex system (P3) produced by INNA is 0.9998896373034 which is better than the best result given by the other compared algorithms and also have most symbolic improvement 8.67%, 1.78%, 47.6%, 0.259% and 66.4% over the results given by GA [14], SCA [70], SAA [72], IA [22], and PSO [71] respectively.

It can be noticed in Table 10, the best optimal solution for overspeed protection system (P4) is 0.9999546746307478 that is obtained by INNA. In fact, the proposed algorithm has succeeded to improve considerably the best known solution found so far by the ten competitive algorithms. By implementing the new search operator, INNA provide more accurate solutions as well as makes the search balance in favour of exploitation behaviour. Table 10 depicts that the improvement indices 1.76E-01%, 1.78E-04%, 1.02E-02%, 1.02E-0%, 7.30E-04%, 1.39E-03%, 1.39E-03%, 5.24E-01%, 1.02E-02%, and 3.60E-03% over the results by SAA [72], IA [22], IPSO [17], NMDE [80], INGHS [77], CS2 [18], EBBO [21], PSO [71], DE [86] and MICA [85] respectively.

Table 11 indicates that INNA executes the same or better than the other existing algorithms given in this literature for solving the convex quadratic reliability problem (P5) and the mixed series-parallel system (P6) in terms of best results. Table 12 reports the test results of the problem P7. It can be seen that the INNA algorithm gives equal or better results compared to other algorithms in terms of the best objective function value for the large-scale problems of dimensions 36, 38, 40, 42 and 50. But in case of dimension 40 and 42, it comes with weaker objective value than two existing algorithms INGHS and IABC. It should be observed that even very small up-gradation in the reliability is important and valuable for the efficiency of the system and it is generally a difficult task to do in real-life applications.

In order to show the convergence performance of the stated algorithm over several existing algorithms like SSA, SCA, SMA, HHO, ABC, TLBO, and NNA, we vary the best solution for each considered problem and the results are plotted in Fig. 9. From this convergence analysis, we can conclude that, as the iteration number increases, the proposed INNA algorithm also shows better performance than the existing algorithms.
| Algorithm | \((x_1, x_2, x_3, x_4, x_5)\) | \(r_1\)   | \(r_2\)   | \(r_3\)   | \(r_4\)   | \(r_5\)   | \(R_5(R)\)  | \(\text{Slack}(g_1)\) | \(\text{Slack}(g_2)\) | \(\text{Slack}(g_3)\) | \(\text{MPI}(\%)\) |
|-----------|----------------|----------|----------|----------|----------|----------|----------------|----------------|----------------|----------------|----------------|
| GA        | (3, 3, 1, 2, 3) | 0.838193 | 0.855065 | 0.878859 | 0.911402 | 0.850355 | 0.99996875 | 53              | 0.0000E-00     | 7.110849       | 5.02E+01       |
| SCA       | (2, 2, 2, 2, 4) | 0.785452 | 0.842998 | 0.885333 | 0.917958 | 0.870318 | 0.99997418 | 40              | 1.1944E-00     | 1.609289       | 3.97E+01       |
| SAA       | (2, 2, 2, 2, 4) | 0.819596 | 0.845000 | 0.895514 | 0.895519 | 0.868456 | 0.99997665 | 40              | 7.0000E-06     | 1.609289       | 3.33E+01       |
| IA        | (2, 2, 2, 2, 4) | 0.812161 | 0.853346 | 0.897597 | 0.900710 | 0.866316 | 0.99997361 | 40              | 7.3000E-03     | 1.609289       | 3.42E+01       |
| IPSO      | (2, 2, 2, 2, 4) | 0.819746 | 0.845008 | 0.895458 | 0.900903 | 0.868407 | 0.99997311 | 40              | 1.4695E+00     | 1.609289       | 3.13E+01       |
| ABC1      | (2, 2, 2, 2, 4) | 0.819592 | 0.844951 | 0.895428 | 0.895522 | 0.868490 | 0.99997665 | 40              | 5.9846E-04     | 1.609289       | 3.33E+01       |
| CPSO      | (2, 2, 2, 2, 4) | 0.819185 | 0.843664 | 0.894730 | 0.895376 | 0.869127 | 0.99997664 | 40              | 5.6100E-04     | 1.609289       | 3.33E+01       |
| CS1       | (2, 2, 2, 2, 4) | 0.819027 | 0.845268 | 0.895492 | 0.895441 | 0.868319 | 0.99997665 | 40              | 1.6100E-06     | 1.609289       | 3.33E+01       |
| CS-GA     | (2, 2, 2, 2, 4) | 0.819660 | 0.844982 | 0.895519 | 0.895492 | 0.868447 | 0.99997665 | 40              | 1.7000E-08     | 1.609289       | 3.33E+01       |
| ABC2      | (2, 2, 2, 2, 4) | 0.819738 | 0.844991 | 0.895530 | 0.895434 | 0.868435 | 0.99997665 | 40              | 1.3915E-10     | 1.609289       | 3.33E+01       |
| MPSO      | (2, 2, 2, 2, 4) | 0.819660 | 0.844981 | 0.895506 | 0.895406 | 0.868448 | 0.99997665 | 40              | 1.9616E-07     | 1.609289       | 3.33E+01       |
| INGHS     | (2, 2, 2, 2, 4) | 0.819812 | 0.844951 | 0.895670 | 0.895233 | 0.868438 | 0.99997665 | 40              | 5.3054E-05     | 1.609289       | 3.33E+01       |
| CS2       | (2, 2, 2, 2, 4) | 0.819483 | 0.844783 | 0.895810 | 0.895220 | 0.868542 | 0.99997665 | 40              | 2.7217E-10     | 1.609289       | 3.33E+01       |
| DE        | (2, 2, 2, 2, 4) | 0.819660 | 0.844981 | 0.895506 | 0.895406 | 0.868448 | 0.99997665 | 40              | 1.9616E-07     | 1.609289       | 3.33E+01       |
| EBB1      | (2, 2, 2, 2, 4) | 0.819658 | 0.844910 | 0.895487 | 0.895158 | 0.868446 | 0.99997665 | 40              | 1.7485E-05     | 1.609289       | 3.33E+01       |
| PSO       | (2, 2, 2, 1, 2) | 0.840253 | 0.886651 | 0.623750 | 0.93850 | 0.751587 | 0.9985845 | 68              | 9.1691E-01     | 4.017704       | 8.90E+01       |
| NAFSA     | (2, 2, 2, 2, 4) | 0.819788 | 0.845672 | 0.894868 | 0.895908 | 0.868296 | 0.99997665 | 40              | 3.1248E-08     | 1.609289       | 3.33E+01       |
| INNA      | (2, 3, 2, 2, 4) | 0.84342538 | 0.79318760 | 0.89238731 | 0.89260221 | 0.86456512 | 0.99998442283 | 20              | 1.5645E-05     | 0.268197665 | –                |
| Algorithm | \((x_1, x_2, x_3, x_4, x_5)\) | \(r_1\) | \(r_2\) | \(r_3\) | \(r_4\) | \(r_5\) | \(R_5(R)\) | \(\text{Slack}(g_1)\) | \(\text{Slack}(g_2)\) | \(\text{Slack}(g_3)\) | MPI(%) |
|-----------|----------------|-------|-------|-------|-------|-------|---------|----------|----------|----------|--------|
| GA        | (3, 3, 3, 3, 1) | 0.814090 | 0.864614 | 0.890291 | 0.701190 | 0.734731 | 0.99987916 | 18 | 3.7634E-01 | 4.264770 | 8.67E+00 |
| SAA       | (3, 3, 3, 3, 1) | 0.868116 | 0.807263 | 0.872862 | 0.712667 | 0.751034 | 0.99988764 | 40 | 7.3000E-03 | 1.609289 | 1.78E+00 |
| SCA       | (3, 3, 2, 3, 2) | 0.814483 | 0.821383 | 0.896151 | 0.713091 | 0.814091 | 0.99978940 | 18 | 1.8540E+00 | 4.264770 | 4.76E+01 |
| NGHS      | (3, 3, 2, 4, 1) | 0.829840 | 0.857989 | 0.913339 | 0.646745 | 0.703109 | 0.99989000 | 5  | 5.9400E-06 | 1.560466 | 3.38E-02 |
| IA        | (3, 3, 3, 3, 1) | 0.816624 | 0.868767 | 0.858749 | 0.710279 | 0.753429 | 0.99988900 | 18 | 4.0421E-08 | 4.264770 | 2.59E-01 |
| EGHS      | (3, 3, 2, 4, 1) | 0.829840 | 0.857989 | 0.913339 | 0.646745 | 0.703110 | 0.99988960 | 5  | 5.9400E-06 | 1.560466 | 3.38E-02 |
| ABC1      | (3, 3, 2, 4, 1) | 0.828087 | 0.857805 | 0.704163 | 0.648146 | 0.914240 | 0.99988962 | 5  | 25.43E+00  | 1.560466 | 1.57E+02 |
| IPSO      | (3, 3, 2, 4, 1) | 0.828684 | 0.858026 | 0.913646 | 0.648034 | 0.702276 | 0.99988963 | 5  | 3.5900E-06 | 1.560466 | 6.62E-03 |
| ABC2      | (3, 3, 2, 4, 1) | 0.827970 | 0.857875 | 0.914186 | 0.648355 | 0.703575 | 0.99988963 | 5  | 3.7463E-04 | 1.560466 | 1.35E-03 |
| INGHS     | (3, 3, 2, 4, 1) | 0.827985 | 0.857680 | 0.914156 | 0.648481 | 0.704865 | 0.99988963 | 5  | 1.8900E-06 | 1.560466 | 8.19E-04 |
| CS2       | (3, 3, 2, 4, 1) | 0.827856 | 0.857626 | 0.914753 | 0.648217 | 0.702670 | 0.99988963 | 5  | 1.0672E-10 | 1.560466 | 4.90E-03 |
| EBBIO     | (3, 3, 2, 4, 1) | 0.828061 | 0.858040 | 0.914149 | 0.647969 | 0.704205 | 0.99988963 | 5  | 1.4541E-04 | 1.560466 | 8.19E-04 |
| PSO       | (3, 3, 2, 2, 3) | 0.770616 | 0.901092 | 0.892768 | 0.600830 | 0.73451  | 0.99967140 | 37  | 16.545713 | 1.41E+00 | 6.64E+01 |
| NAFSA     | (3, 3, 2, 4, 1) | 0.828322 | 0.857974 | 0.914221 | 0.647757 | 0.703007 | 0.99988963 | 5  | 1.5485E-05 | 1.560466 | 1.18E-03 |
| MICA      | (3, 3, 2, 4, 1) | 0.827642 | 0.857478 | 0.914197 | 0.649274 | 0.704092 | 0.99988963 | 5  | 4.4280E-05 | 1.560466 | 6.62E-03 |
| INNA      | (3, 3, 2, 4, 1) | 0.82800430 | 0.85775303 | 0.91433036 | 0.64825542 | 0.70382891 | 0.999889637303 | 5  | 1.1872E-04 | 1.56046628 | –       |
Table 10 Comparison of the best result for the Overspeed system (P4) with other results in the literature

| Algorithms | $(x_1, x_2, x_3, x_4)$ | $r_1$  | $r_2$  | $r_3$  | $r_4$  | $R_0(R)$  | $Slack(g_1)$ | $Slack(g_2)$ | $Slack(g_3)$ | MPI(%) |
|------------|------------------------|--------|--------|--------|--------|------------|--------------|--------------|--------------|--------|
| SAA        | (5, 5, 5, 5)           | 0.895644 | 0.885878 | 0.912184 | 0.887785 | 0.999945000 | 50           | 9.3800E-01  | 28.8037   | 1.76E+01 |
| IA         | (5, 5, 4, 6)           | 0.901589 | 0.888193 | 0.948167 | 0.849970 | 0.999954674 | 55           | 1.2495E-04  | 15.3634   | 1.78E-04 |
| IPSO       | (5, 5, 4, 5)           | 0.901631 | 0.849970 | 0.948218 | 0.888128 | 0.999954674 | 55           | 9.0000E-06  | 24.0818   | 1.02E-02 |
| NMDE       | (5, 6, 4, 5)           | 0.901615 | 0.849921 | 0.948141 | 0.888223 | 0.999954674 | 55           | 1.0570E-05  | 24.0818   | 1.02E-02 |
| INGHS      | (5, 5, 4, 6)           | 0.901556 | 0.888243 | 0.948111 | 0.849982 | 0.999954674 | 55           | 5.0540E-05  | 24.0818   | 7.30E-04 |
| CS2        | (5, 5, 4, 6)           | 0.901598 | 0.888226 | 0.948102 | 0.849981 | 0.999954674 | 55           | 8.8249E-10  | 15.3634   | 1.39E-03 |
| EBBBO      | (5, 5, 4, 6)           | 0.901563 | 0.888225 | 0.948156 | 0.849953 | 0.999954674 | 55           | 2.7021E-05  | 15.3634   | 1.39E-03 |
| PSO        | (4, 6, 5, 5)           | 0.929523 | 0.813703 | 0.886637 | 0.899872 | 0.999904000 | 37           | 11.52E+00  | 11.6447   | 5.24E+01 |
| DE         | (5, 6, 4, 5)           | 0.901615 | 0.849921 | 0.948141 | 0.888222 | 0.999954670 | 55           | 1.0051E-05  | 24.0818   | 1.02E-02 |
| MICA       | (5, 5, 4, 5)           | 0.901489 | 0.850035 | 0.948130 | 0.888238 | 0.999954673 | 55           | 2.1378E-03  | 24.0818   | 3.60E-03 |
| INNA       | (5, 5, 4, 6)           | 0.901588 | 0.8882576| 0.9481337| 0.8498978| 0.99995467463| 55           | 5.2747E-09  | 15.363463 | –        |

Table 11 Comparison of the best result for the Convex system (P5) and Mixed series-parallel system (P6) with other results in the literature

| Problems | Methods | $n$ | $R_0(r, n)$ | $Slack(g_1)$ | $Slack(g_2)$ | $Slack(g_3)$ | $Slack(g_4)$ |
|----------|---------|----|-------------|--------------|--------------|--------------|--------------|
| P5       | GA      | (2, 2, 1, 1, 2, 3, 2, 1, 2) | 0.808844 | –           | –            | –            | –            |
|          | HDE     | (2, 2, 2, 1, 1, 2, 3, 2, 1, 2) | 0.808844 | –           | –            | –            | –            |
|          | INGHS   | (2, 2, 2, 1, 1, 2, 3, 2, 1, 2) | 0.8088441896 | 0.9649E+13 | 0.0203E+13  | 4.3632E+13  | 0.0872E+13  |
|          | IABC    | (2, 2, 2, 1, 1, 2, 3, 2, 1, 2) | 0.8088441896 | 0.9649E+13 | 0.0203E+13  | 4.3632E+13  | 0.0871E+13  |
|          | INNA    | (2, 2, 2, 1, 1, 2, 3, 2, 1, 2) | 0.8088441896 | 9.6493E+12 | 2.0299E+11  | 4.3632E+13  | 8.7149e+11  |
| P6       | GA      | (3, 4, 5, 3, 2, 4, 5, 3, 4, 5, 5) | 0.9202 | –           | –            | –            | –            |
|          | HDE     | (3, 4, 6, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4) | 0.945613 | –           | –            | –            | –            |
|          | INGHS   | (3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4) | 0.945613574 | 8           | 0            | –            | –            |
|          | IABC    | (3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4) | 0.945613574 | 8           | 0            | –            | –            |
|          | INNA    | (3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4) | 0.945613574 | 8           | 0            | –            | –            |
5.3 Diversity and exploration-exploitation analysis

For an effective in-depth performance analysis, the population diversity and the exploration-exploitation measurement in INNA and the conventional NNA are presented in Table 13 while solving the reliability optimization problems. A graphical presentation on comparison of diversity measurement and the exploration-exploitation phases between the proposed INNA and the basic NNA are also given in Figs. 10 and 11 respectively. According to Table 13, the proposed INNA algorithm keeps the population diversity high compared to the conventional NNA for all the reliability problems. In case of solving P1 to P6, INNA maintained population diversity value 14.4026, 14.2185, 14.4711, 12.5628, 20.1557, and 38.7695 which is

| Dim | Methods | VTV  | \(R_5(r,n)\) | Slack\(g_1\) | Slack\(g_2\) | Slack\(g_3\) | Slack\(g_4\) |
|-----|---------|------|----------------|----------------|----------------|----------------|----------------|
| 36  | SCA     | (5, 10, 15, 21, 33) | 0.519976 | – | – | – | – |
|     | NGHS    | (5, 10, 15, 21, 33) | 0.519976 | – | – | – | – |
|     | IPSO    | (5, 10, 15, 21, 33) | 0.519976 | – | – | – | – |
|     | ICS     | (5, 10, 15, 21, 33) | 0.519976 | – | – | – | – |
|     | CS1     | (5, 10, 15, 21, 33) | 0.51997597 | – | – | – | – |
|     | INGHS   | (5, 10, 15, 21, 33) | 0.51997597 | 1 | 49.125763 | 109 | 301.353247 |
|     | IABC    | (5, 10, 15, 21, 33) | 0.51997597 | 1 | 49.125763 | 109 | 301.353247 |
|     | INNA    | (5, 10, 15, 21, 33) | 0.51997597 | 1 | 49.125763 | 109 | 291.353247 |
| 38  | SCA     | (10, 13, 15, 21, 33) | 0.510989 | – | – | – | – |
|     | NGHS    | (10, 13, 15, 21, 33) | 0.510989 | – | – | – | – |
|     | IPSO    | (10, 13, 15, 21, 33) | 0.510989 | – | – | – | – |
|     | ICS     | (10, 13, 15, 33) | 0.510989 | – | – | – | – |
|     | INGHS   | (10, 13, 15, 21, 33) | 0.510985965 | 1 | 53.638551 | 115 | 317.039538 |
|     | IABC    | (10, 13, 15, 21, 33) | 0.510985965 | 1 | 53.638551 | 115 | 317.039538 |
|     | INNA    | (10, 13, 15, 21, 33) | 0.510985965 | 1 | 53.638551 | 115 | 317.039538 |
| 40  | SCA     | (5, 10, 13, 15, 33) | 0.503292 | – | – | – | – |
|     | NGHS    | (5, 10, 13, 15, 33) | 0.503292 | – | – | – | – |
|     | IPSO    | (5, 10, 13, 15, 33) | 0.503292 | – | – | – | – |
|     | ICS     | (5, 10, 13, 33) | 0.503292 | – | – | – | – |
|     | CS1     | (4, 10, 11, 21, 22, 33) | 0.50599242 | – | – | – | – |
|     | INGHS   | (4, 10, 11, 21, 22, 33) | 0.50599242 | 0 | 51.047142 | 119 | 333.240549 |
|     | IABC    | (4, 10, 11, 21, 22, 33) | 0.50599242 | 0 | 51.047142 | 119 | 333.240549 |
|     | INNA    | (4, 10, 11, 21, 22, 33) | 0.50599242 | 0 | 51.047142 | 119 | 333.240549 |
| 42  | SCA     | (4, 10, 11, 15, 21, 33) | 0.479664 | – | – | – | – |
|     | NGHS    | (4, 10, 11, 15, 21, 33) | 0.479664 | – | – | – | – |
|     | IPSO    | (4, 10, 11, 15, 21, 33) | 0.479664 | – | – | – | – |
|     | ICS     | (4, 10, 11, 15, 21, 33) | 0.479664 | – | – | – | – |
|     | CS1     | (4, 10, 11, 15, 21, 33) | 0.47966355 | – | – | – | – |
|     | INGHS   | (4, 10, 11, 15, 21, 33) | 0.47966355 | 2 | 52.718250 | 129 | 354.583694 |
|     | IABC    | (4, 10, 11, 15, 21, 33) | 0.47966355 | 2 | 52.718250 | 129 | 354.583694 |
|     | INNA    | (4, 10, 11, 15, 21, 33) | 0.47966355 | 2 | 52.718250 | 129 | 354.583694 |
| 50  | SCA     | (4, 10, 15, 21, 33, 45, 47) | 0.405390 | – | – | – | – |
|     | NGHS    | (4, 10, 15, 21, 33, 45, 47) | 0.405390 | – | – | – | – |
|     | IPSO    | (4, 10, 15, 21, 33, 45, 47) | 0.405390 | – | – | – | – |
|     | ICS     | (4, 10, 15, 21, 33, 42, 45) | 0.40695474 | – | – | – | – |
|     | CS1     | (4, 10, 15, 21, 33, 42, 45) | 0.40695474 | 0 | 61.955982 | 154.0 | 433.914647 |
|     | INGHS   | (4, 10, 15, 21, 33, 42, 45) | 0.40695474 | 0 | 61.955982 | 154.0 | 433.914647 |
|     | IABC    | (4, 10, 15, 21, 33, 42, 45) | 0.40695474 | 0 | 61.955982 | 154.0 | 433.914647 |
|     | INNA    | (4, 10, 15, 21, 33, 42, 45) | 0.40695474 | 0 | 61.955982 | 154.0 | 433.914647 |
Fig. 9 Comparison of Convergence curve of INNA with SSA, SCA, SMA, HHO, ABC, TLBO and NNA
In view of the above two quality parameters, the statistical results achieved for INNA and all other existing algorithms (like SSA, SCA, SMA, HHO, ABC, TLBO, NNA) are computed and summarized in Table 14 for the considered problems. In this table, the mean, Std and median of the best fitness value after the 30 independent runs of each algorithm is reported. From this table, it is observed that the proposed algorithm is rank 1 followed by the other algorithm, which shows its stability and convergence for all of the reliability issues. Also, we can sort the ranking, as per their achievement, in the order: INNA, TLBO, SMA, SSA, NNA/ABC, HHO and SCA.

The ranking order in Table 14 indicates that the TLBO algorithm shows strong competitiveness and is the second-best on all test issues except Overspeed system. It can therefore be argued that INNA is an efficient and effective method for solving various kinds of optimization problems. Figure 12 provides a better visualization of the ranking of all compared algorithms for solving RRAPs.

Apart from this analysis, a statistical test named Wilcoxon signed-rank test is performed to check the statistical significance of the results obtained from the proposed algorithm.

II. Wilcoxon signed-rank test This statistical test-based method [88] is used to compare the performance of the proposed INNA with the other algorithms. Also, it has several advantages, compared to the t-test, such as: (1) normal distributions is not considered here for the sample tested; (2) It’s less affected and more responsive than the t-test. This advantages makes it more powerful test for comparing two algorithms [89–91]. Wilcoxon signed-rank test is performed here with a significance level $\alpha = 0.05$ and the obtained results are shown in Table 15. In this table, “H” scored “1” if there is a symbolic difference between INNA and the existing algorithm and also “H” is labelled as “0” if there is no significant difference. Again, the sign of “S” is taken as “+” if the proposed algorithm is superior to the compared algorithm and “−” is assigned to “S” if INNA is inferior to the compared algorithm. It is noted that the proposed algorithm INNA dominates all compared algorithms on all reliability problems. Thus, from this analysis, we conclude that the proposed INNA can obtain better solutions than the comparative algorithms, which means that the proposed method has a better global performance optimization capability than the comparable algorithms.

III. Kruskal-Wallis and Multiple Comparison Test The multiple comparison test (MCT) test is performed here to justify whether the proposed INNA algorithm is better than the other optimizers (e.g., SSA, SCA, SMA, HHO, ABC, TLBO, NNA). For this purpose, we perform a non-parametric Kruskal-Wallis test (KWT) between the best values obtained for each problem considered. This test was used to investigate the hypothesis that the different independent samples of the distributions had or did not have the same

| Problems | INNA | NNA |
|----------|------|-----|
|          | Diversity | Expl%:Exp% | Diversity | Expl%:Exp% |
|          | P1 14.4026 48:52 9.5045 35:65 |
|          | P2 14.2185 48:52 9.6495 36:64 |
|          | P3 14.4711 48:52 9.5714 35:65 |
|          | P4 12.5628 46:54 8.5150 36:64 |
|          | P5 20.1557 49:51 13.2548 35:65 |
|          | P6 38.7695 51:49 21.6344 35:65 |
estimates. On the other hand, the MCT is used to determine the significant difference between the different estimates by performing multiple comparisons using one-way ANOVA. To addressed this, the significance of the proposed INNA algorithm results are compared with the compared algorithms results. The optimized results

Fig. 10 Comparison on Diversity measurement between NNA and INNA on reliability problems (P1–P7)
Fig. 11 Exploration-exploitation measurement of NNA and INNA on reliability problems (P1–P7)
between the pairs of the different algorithms are summarized in Table 16. In this table, the first column represents the problem considered, while the second column indicates the indices between the pairs of the different samples. The third and fifth column describes the boundary of the true mean difference between the samples considered at a 5% level of significance. At the end of the last column, the p-value of the test obtained by KWT corresponds to the null hypothesis of equal means.

The box-plot and the MCT graphs for the problems (P1 - P6) considered are shown in Fig. 13. In this figure, the left graph describes the boxes with the values of the 1st, 2nd and 3rd quarters, while the vertical lines that extend the boxes are called the whisker lines that provide information on the re-imagining values. On the other hand, on the right side of this figure, the MCT makes a multiple comparison between the different pairs and makes a significant difference between them. The blue line on these graphs represents the proposed INNA results and the red line indicates which algorithm results (such as SSA, SCA, SMA, HHO, ABC, TLBO, NNA) are statistically significant from the proposed INNA. For example, in case of series system (P1), as shown in Fig. 13 we calculate that the existing algorithms (SSA, SCA, SMA, HHO, and ABC) have statistically significant resources from the INNA algorithm. Furthermore, the vertical lines (right/left, shown in black colour) shown around the INNA results (displayed in blue colour) describe the marginal area to show which method is statistically better or not considered to be problematic. From this analysis and the results are shown in Fig. 13 and Table 16, we conclude that the performance of the proposed algorithm is statistically significant with the other algorithms. The best results are therefore provided by the INNA.

6 Conclusions and future work

In practical life, we have to handle various types of complex optimization problems that appears in the field of engineering and as a result, an efficient and accurate methods are required to deal with them. This paper introduces an improved neural network algorithm (INNA) for
| Problems | INNA  | NNA   | TLBO  | ABC   | HHO   | SMA   | SCA   | SSA   |
|----------|-------|-------|-------|-------|-------|-------|-------|-------|
| P1       | 0.9316823879 | 0.9316823658 | 0.931682380 | 0.9303824905 | 0.923216599 | 0.9316562324 | 0.9197764827 | 0.9316823793 |
| Mean     | 0.9309717985 | 0.9274360928 | 0.929505661 | 0.9243814737 | 0.8972295012 | 0.927729201 | 0.8944273654 | 0.9245305903 |
| Std      | 1.19E-03 | 3.13E-03 | 3.11E-03 | 4.66E-03 | 1.93E-02 | 3.36E-03 | 1.83E-02 | 5.74E-03 |
| Median   | 0.9316777002 | 0.9271935041 | 0.9316810357 | 0.9258462866 | 0.9062971655 | 0.928363319 | 0.8961627775 | 0.9246467135 |
| Rank     | 1      | 6      | 2      | 5      | 7      | 3      | 8      | 4      |
| P2       | 0.9999844228 | 0.9999833840 | 0.9999863211 | 0.9999844896 | 0.9999851850 | 0.9999862453 | 0.9999852690 | 0.9999863373 |
| Mean     | 0.9999829421 | 0.99993171890 | 0.9999813410 | 0.9999756365 | 0.9999770319 | 0.9999796647 | 0.9999764160 | 0.9999692476 |
| Std      | 1.81E-06 | 1.25E-04 | 2.52E-06 | 7.35E-06 | 3.15E-05 | 1.21E-05 | 3.32E-05 | 1.76E-05 |
| Median   | 0.9999838413 | 0.9999794560 | 0.9999804569 | 0.9999773879 | 0.9999721157 | 0.999979231 | 0.999921671 | 0.9999798144 |
| Rank     | 1      | 7      | 2      | 4      | 6      | 3      | 8      | 5      |
| P3       | 0.998896373 | 0.998985729 | 0.9998824392 | 0.9998752654 | 0.9998919292 | 0.9999882589 | 0.9999565706 | 0.9999512153 |
| Mean     | 0.998892381 | 0.9985312241 | 0.998631093 | 0.9998427879 | 0.9996770319 | 0.9998767824 | 0.9999673706 | 0.9999851215 |
| Std      | 3.05E-07 | 6.83E-05 | 3.49E-05 | 1.86E-05 | 1.55E-04 | 7.11E-05 | 9.39E-05 | 2.14E-05 |
| Median   | 0.9998893081 | 0.9989498128 | 0.9989857341 | 0.9989429582 | 0.9997219627 | 0.998542653 | 0.996852794 | 0.9998513151 |
| Rank     | 1      | 6      | 2      | 4      | 7      | 5      | 8      | 3      |
| P4       | 0.9999546746 | 0.9999546746 | 0.9999546746 | 0.9999536832 | 0.999941232 | 0.9999546645 | 0.9998727414 | 0.9999546746 |
| Mean     | 0.999950830 | 0.9999456290 | 0.9999432315 | 0.9999437241 | 0.999752034 | 0.9999427318 | 0.9999568798 | 0.9999407833 |
| Std      | 1.20E-05 | 1.06E-05 | 7.36E-05 | 6.53E-06 | 2.15E-04 | 2.20E-05 | 2.34E-04 | 1.60E-05 |
| Median   | 0.9999546742 | 0.9999461511 | 0.9999443586 | 0.9998946633 | 0.999943821 | 0.9999543821 | 0.9999547068 | 0.9999461343 |
| Rank     | 1      | 6      | 2      | 4      | 7      | 5      | 8      | 3      |
| P5       | 0.8088441896 | 0.8088441896 | 0.8088441896 | 0.7857228761 | 0.808441896 | 0.808441896 | 0.808441896 | 0.808441896 |
| Mean     | 0.8088441896 | 0.7796532244 | 0.8044835709 | 0.647076562 | 0.803624111 | 0.8002047093 | 0.796651734 | 0.7924251753 |
| Std      | 5.65E-16 | 2.60E-02 | 7.64E-03 | 7.74E-02 | 9.36E-03 | 1.00E-02 | 1.02E-02 | 1.76E-02 |
| Median   | 0.8088441896 | 0.7808355252 | 0.8088441896 | 0.6400778320 | 0.8088441896 | 0.8088441896 | 0.805473903 | 0.794475387 |
| Rank     | 1      | 2      | 6      | 3      | 7      | 5      | 8      | 4      |
| P6       | 0.9456133574 | 0.9456133574 | 0.9456133574 | 0.8347527546 | 0.9447484845 | 0.9456133574 | 0.9210481246 | 0.9452180086 |
| Mean     | 0.9454201515 | 0.9424717453 | 0.9443294149 | 0.714608865 | 0.9420684305 | 0.9426414224 | 0.8895852425 | 0.9409797892 |
| Std      | 2.90E-04 | 2.32E-03 | 1.25E-03 | 6.33E-02 | 2.12E-03 | 2.37E-03 | 1.97E-02 | 5.92E-03 |
| Median   | 0.9456133574 | 0.9432527335 | 0.9447484846 | 0.7008594171 | 0.9401279010 | 0.9432527335 | 0.8930756445 | 0.9425524293 |
| Rank     | 1      | 4      | 2      | 8      | 6      | 3      | 7      | 5      |

Average ranking 1 5.33 2.67 5.33 6 3.83 7.33 4.5
Ranking 1 5.5 2 5.5 7 3 8 4
solving some RRAP with non-linear resource constraints. In this study, INNA is proposed by implementing a new logarithmic spiral search operator and the searching strategy of the learner phase of TLBO to the basic NNA to make a proper balance between exploration and exploitation. Here, the basic NNA looks after the exploration part and the presence of a new search operator and the searching strategy of TLBO increases the exploitation capability of the algorithm, which makes the proposed algorithm an efficient, effective, and more acceptable for solving COPs.

A comprehensive set of seven well-known reliability optimization problems consist of series system, series-parallel system, complex systems, overspeed protection systems, convex quadratic system, mixed series-parallel system and large scale system are employed to examine the performance of INNA and compared with several reported algorithms in the literature. All of these problems considered are mixed variables – discrete, continuous and integer. The results are also compared statistically with 7 competitive MHAs including SSA, SCA, SMA, HHO, ABC, TLBO, and NNA. According to the experimental results, the proposed INNA outperforms the compared algorithms for all reliability issues in terms of best and mean values. In addition, in order to eliminate the stochastic nature of the algorithm, we perform several statistical tests, namely a Tied-rank test and a Wilcoxon signed-rank test for P1 to P6. All of the above discussions and evaluations in this
Table 15 The comparison results of the applied algorithms by Wilcoxon signed-rank test (a level of significance \( \alpha = 0.05 \))

| Problems | INNA vs | NNA | TLBO | ABC | HHO | SMA | SCA | SSA |
|----------|---------|-----|------|-----|-----|-----|-----|-----|
|          | p-value            | H   | S   | p-value | H | S | p-value | H | S | p-value | H | S | p-value | H | S | p-value | H | S | p-value | H | S | p-value | H | S | p-value | H | S |
| P1       | 7.5137e-05 | 1   | +   | 2.8948e-01 | 0 | + | 2.8786e-06 | 1 | + | 1.7344e-06 | 1 | + | 1.1265e-05 | 1 | + | 7.6909e-06 | 1 | + |
| P2       | 4.0715e-05 | 1   | +   | 1.2453e-02 | 1 | + | 3.8822e-06 | 1 | + | 1.2381E-05 | 1 | + | 2.3534e-06 | 1 | + | 9.8421e-03 | 1 | + | 1.7344E-06 | 1 | + |
| P3       | 6.8923e-05 | 1   | +   | 4.4493e-05 | 1 | + | 1.7344e-06 | 1 | + | 1.7344e-06 | 1 | + | 1.73440e-06 | 1 | + | 6.1035E-05 | 1 | + | 7.6909e-06 | 1 | + |
| P4       | 2.3038e-02  | 1   | +   | 1.3595e-04 | 1 | + | 9.7110e-05 | 1 | + | 6.9838e-06 | 1 | + | 1.4839e-03 | 1 | + | 1.7344E-06 | 1 | + | 2.4375E-02 | 1 | + |
| P5       | 8.1462E-06  | 1   | +   | 6.3103E-01 | 0 | + | 1.7344E-06 | 1 | + | 3.6671E-01 | 0 | + | 4.2859E-02 | 1 | + | 6.1035E-05 | 1 | + | 2.4375E-02 | 1 | + |
| P6       | 6.5226E-06  | 1   | +   | 1.2255E-04 | 1 | + | 1.7344E-06 | 1 | + | 1.7344E-06 | 1 | + | 5.7060E-06 | 1 | + | 1.7333E-06 | 1 | + | 1.7333E-06 | 1 | + |
| Problems       | Comparing            | Lower bound | Group mean | Upper bound | p-value  | Problems       | Comparing            | Lower bound | Group mean | Upper bound | p-value  |
|----------------|----------------------|-------------|------------|-------------|----------|----------------|----------------------|-------------|------------|-------------|----------|
| P1 INNA vs NNA | 6.3305               | 48.0000     | 102.3305   | 1.29E-01    |          | P4 INNA vs NNA | 13.8786              | 40.4500     | 94.7786    | 3.18E-01    |          |
| INNA vs TLBO   | 41.2139              | 13.1167     | 67.4472    | 9.96E-01    |          | INNA vs TLBO   | 9.0619               | 45.2667     | 99.5953    | 1.85E-01    |          |
| INNA vs ABC    | 29.5695              | 83.9000     | 138.2305   | 7.79E-05    |          | INNA vs ABC    | 17.9381              | 72.2667     | 126.5953   | 1.43E-03    |          |
| INNA vs HHO    | 106.5361             | 160.8667    | 215.1972   | 5.99E-08    |          | INNA vs HHO    | 83.6381              | 137.9667    | 192.2953   | 5.99E-08    |          |
| INNA vs SMA    | 1.0361               | 55.3667     | 109.6972   | 4.21E-02    |          | INNA vs SMA    | -0.7953              | 53.5333     | 107.8619   | 5.69E-02    |          |
| INNA vs SCA    | 111.5361             | 165.8667    | 220.1972   | 5.99E-08    |          | INNA vs SCA    | 116.1381             | 170.4667    | 224.7953   | 5.99E-08    |          |
| INNA vs SSA    | 18.8195              | 73.1500     | 127.4805   | 1.17E-03    |          | INNA vs SSA    | -3.0786              | 51.2500     | 105.5786   | 8.11E-02    |          |
| P2 INNA vs NNA | 13.6696              | 68.0000     | 122.3304   | 3.71E-03    |          | P5 INNA vs NNA | 32.9676              | 86.9000     | 140.8323   | 2.86E-05    |          |
| INNA vs TLBO   | -26.4637             | 27.8667     | 82.1971    | 7.77E-01    |          | INNA vs TLBO   | -82.7323             | -28.8000    | 25.1323    | 7.39E-01    |          |
| INNA vs ABC    | 27.5029              | 81.8333     | 136.1637   | 1.35E-04    |          | INNA vs ABC    | 86.4676              | 140.4000    | 194.3323   | 5.99E-08    |          |
| INNA vs HHO    | 58.3196              | 112.6500    | 166.9804   | 6.85E-08    |          | INNA vs HHO    | -67.4323             | -13.5000    | 40.4323    | 9.95E-08    |          |
| INNA vs SMA    | -9.6137              | 44.7167     | 99.0471    | 1.98E-01    |          | INNA vs SMA    | -37.9990             | 15.9333     | 69.8657    | 9.86E-01    |          |
| INNA vs SCA    | 110.1529             | 164.4833    | 218.8137   | 5.99E-08    |          | INNA vs SCA    | -15.7990             | 38.1333     | 92.0657    | 3.87E-01    |          |
| INNA vs SSA    | 15.3196              | 69.6500     | 123.9804   | 2.59E-03    |          | INNA vs SSA    | 3.0009               | 56.9333     | 110.8657   | 2.99E-01    |          |
| P3 INNA vs NNA | 22.8860              | 77.2167     | 131.5474   | 4.38E-04    |          | P6 INNA vs NNA | 23.1277              | 77.4333     | 131.7389   | 4.12E-04    |          |
| INNA vs TLBO   | -3.0474              | 51.2833     | 105.6140   | 8.07E-02    |          | INNA vs TLBO   | -14.0389             | 40.2667     | 94.5722    | 3.23E-01    |          |
| INNA vs ABC    | 32.5526              | 86.8833     | 141.2140   | 3.44E-05    |          | INNA vs ABC    | 147.3278             | 201.6333    | 255.9389   | 5.99E-08    |          |
| INNA vs HHO    | 111.8860             | 166.2167    | 220.5474   | 5.99E-08    |          | INNA vs HHO    | 61.6278              | 115.9333    | 170.2389   | 6.24E-08    |          |
| INNA vs SMA    | 33.9360              | 88.2667     | 142.5974   | 2.33E-05    |          | INNA vs SMA    | 21.6278              | 75.9333     | 130.2389   | 5.95E-04    |          |
| INNA vs SCA    | 124.7860             | 179.1167    | 233.4474   | 5.99E-08    |          | INNA vs SCA    | 117.2611             | 171.5667    | 225.8722   | 5.99E-08    |          |
| INNA vs SSA    | 25.3526              | 79.6833     | 134.0140   | 2.36E-04    |          | INNA vs SSA    | 35.9944              | 90.3000     | 144.6056   | 1.29E-05    |          |
Fig. 13 Box plot of objective function using the reported optimizers
study ensure that the proposed algorithm is a competitive approach, not only that it performs well but also that it can effectively achieve the best results for all reliability problems.

In future work, we will broaden the proposed INNA to solve more complex design and constraint optimization problems. We are also trying to expand the approach by using the neural network and applying it to solve different kinds of problems, such as the stock market, finance, decision-making, etc. Further, we will try to improve the other MHAs by the proposed reinforcement searching mechanism.

Declarations

Conflicts of interest The authors declared that they have no conflicts of interest to this work.

References

1. Tillman FA, Hwang CL, Kuo W (1977) Optimization techniques for system reliability with redundancy-a review. IEEE Trans Reliab 26(3):148–155
2. Kundu T, Islam S (2018) Neutrosophic goal geometric programming problem and its application to multi-objective reliability optimization model. Int J Fuzzy Syst 20(6):1986–1994
3. Kundu T, Islam S (2019) A new interactive approach to solve entropy based fuzzy reliability optimization model. Int J Interact Des Manuf 13(1):137–146
4. Kundu T, Islam S (2019) An interactive weighted fuzzy goal programming technique to solve multi-objective reliability optimization problem. J Ind Eng Int 15:95–104
5. Kuo W, Rajendra Prasad V (2000) An annotated overview of system-reliability optimization. IEEE Trans Reliab 49(2):176–187
6. Kuo W, Wan R (2007) Recent advances in optimal reliability allocation. IEEE Trans Syst Man Cybern Part A Syst Hum 37(2):143–156
7. Ravi V, Reddy PJ, Zimmermann HJ (2000) Fuzzy global optimization of complex system reliability. IEEE Trans Fuzzy Syst 8(3):241–248
8. Islam S, Kundu T (2018) Neutrosophic goal geometric problem based geometric mean method and its application. Neutrosophic Sets Syst 19:80–90
9. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
10. Kennedy J, Eberhart R. (1995). Particle swarm optimization. Proceedings of ICNN95 - International Conference on Neural Networks, 4, 1942–1948
11. Coit DW, Smith AE (1996) Penalty guided genetic search for reliability design optimization. Comput Ind Eng 30(4):895–904
12. Gen M, Ida K, Lee CY (1999) Hybridized neural network and genetic algorithms for solving nonlinear integer programming problem. Lecture Notes in Computer Science (Including
Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics 1585(April):421–429

13. Hsieh YC, Chen TC, Bricker DL (1998) Genetic algorithms for reliability design problems. Microelectron Reliab 38(10):1599–1605

14. Yokota T, Gen M, Li YX (1996) Genetic algorithm for non-linear mixed integer programming problems and its applications. Comput Ind Eng 30(4):905–917

15. Beji N, Jarboui B, Eddaly M, Chabchoub H (2010) A hybrid particle swarm optimization algorithm for the redundancy allocation problem. J Comput Sci 1(3):159–167

16. Coelho L, Dos S (2009) An efficient particle swarm approach for mixed-integer programming in reliability-redundancy optimization applications. Reliab Eng Syst Saf 94(4):830–837

17. Wu P, Gao L, Zou D, Li S (2011) An improved particle swarm optimization algorithm for reliability problems. ISA Trans 50(1):71–81

18. Garg H (2015) An approach for solving constrained reliability-redundancy allocation problems using cuckoo search algorithm. Beni-Suef Univ J Basic Appl Sci 4:14–25

19. Kanagaraj G, Ponnambalam SG, Jawahar N (2013) A hybrid cuckoo search and genetic algorithm for reliability-redundancy allocation problems. Comput Ind Eng 66(4):1115–1124

20. Meziane R, Massim Y, Zebalh A, Ghoraif A, Rahli R (2005) Reliability optimization using ant colony algorithm under performance and cost constraints. Electric Power Syst Res 76(1–3):1–8

21. Garg H (2015) An efficient biogeography-based optimization algorithm for solving reliability optimization problems. Swarm Evol Comput 24:1–10

22. Hsieh YC, You PS (2011) An effective immune based two-phase approach for the optimal reliability-redundancy allocation problem. Appl Math Comput 218(4):1297–1307

23. Nahas N, Thien-My D (2010) Harmony search algorithm: application to the redundancy optimization problem. Eng Optim 42(9):845–861

24. Krishna GJ, Ravi V (2016) Modified harmony search algorithm applied to reliability optimization of complex systems. Adv Intell Syst Comput 382:169–180

25. Zou D, Gao L, Li S, Wu J (2011) An effective global harmony search algorithm for reliability problems. Expert Syst Appl 38(4):4642–4648

26. Zou D, Gao L, Wu J, Li S, Li Y (2010) A novel global harmony search algorithm for reliability problems. Comput Ind Eng 58(2):307–316

27. Wang L, Li LP (2012) A coevolutionary differential evolution with harmony search for reliability-redundancy optimization. Expert Syst Appl 39(5):5271–5278

28. Kumar A, Pant S, Ram M (2017) System reliability optimization using gray wolf optimizer algorithm. Qual Reliab Eng Int 33(7):1327–1335

29. Kumar A, Pant S, Singh SB (2017) Reliability optimization of complex systems using cuckoo search algorithm. In Mathematical Concepts and Applications in Mechanical Engineering and Mechatronics (pp. 94–110). IGI global

30. He Q, Xiangtao H, Ren H, Zhang H (2015) A novel artificial fish swarm algorithm for solving large-scale reliability-redundancy application problem. ISA Trans 59:105–113

31. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359

32. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713

33. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102

34. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39

35. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61

36. Akay B, Karaboga D (2012) Artificial bee colony algorithm for large-scale problems and engineering design optimization. J Intell Manuf 23(4):1001–1014

37. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67

38. Yang XS (2010) A new metaheuristic bat-inspired algorithm. Stud Comput Intell 284:65–74

39. Yang X, Deb S (2009). Cuckoo search via Levy flights. In: 2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings, 210–214

40. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Futur Gener Comput Syst 111:300–323

41. Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12

42. Chou JS, Truong DN (2021) A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. Appl Math Comput 389:125535

43. Zervoudakis K, Tsafarakis S (2020) A mayfly optimization algorithm. Comput Ind Eng 145:106559

44. Heidari AA, Mirjalili S, Farsi H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872

45. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Farsi H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191

46. Savsani P, Savsani V (2016) Passing vehicle search (PVS): a novel metaheuristic algorithm. Appl Math Model 40(5–6):3951–3978

47. Tan Y, Zhu Y. (2010). Fireworks algorithm for optimization. In International conference in swarm intelligence (pp. 355-364). Springer, Berlin, Heidelberg

48. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133

49. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. CAD Comput Aided Des 43(3):303–315

50. Al-Betar MA (2017) β-Hill climbing: an exploratory local search. Neural Comput Appl 28(1):153–168

51. Al-Betar MA, Alyasserzi ZAA, Awadallah MA, Doush IA (2021) Coronavirus herd immunity optimizer (CHIO). Neural Comput Appl 33(10):5011–5042

52. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82

53. Ghavidel S, Azizivahed A, Li L (2018) A hybrid Jaya algorithm for reliability-redundancy allocation problems. Eng Optim 50(4):698–715

54. Juybari MN, Abouei Ardakan M, Davari-Ardakani H (2019) A penalty-guided fractal search algorithm for reliability-redundancy allocation problems with cold-standby strategy. Proc Inst Mech Eng O J Risk Reliab 233(5):775–790

55. Mellal MA, Zio E (2020) System reliability-redundancy optimization with cold-standby strategy by an enhanced nest cuckoo optimization algorithm. Reliab Eng Syst Saf 201:106973

56. Ouyang Z, Liu Y, Ruan SJ, Jiang T (2019) An improved particle swarm optimization algorithm for reliability-redundancy allocation problem with mixed redundancy strategy and heterogeneous components. Reliab Eng Syst Saf 181:62–74
57. Devi S, Garg D (2020) Hybrid genetic and particle swarm algorithm: redundancy allocation problem. Int J Syst Assur Eng Manag 11(2):313–319
58. Gupta S, Deep K, Assad A. (2020). Reliability–redundancy allocation using random walk gray wolf optimizer. In Soft computing for problem solving (pp. 941-959). Springer, Singapore
59. Kundu T, Deepmala, Jain PK (2022) A hybrid salp swarm algorithm based on TLBO for reliability redundancy allocation problems. Appl Intell. https://doi.org/10.1007/s10489-021-02862-w
60. Sadollah A, Sayyaadi H, Yadav A (2018) A dynamic meta-heuristic optimization model inspired by biological nervous systems: neural network algorithm. Appl Soft Comput J 71:747–782
61. Zhang Y, Jin Z, Chen Y (2020) Hybridizing grey wolf optimization with neural network algorithm for global numerical optimization problems. Neural Comput Appl 32(14):10451–10470
62. Zhang Y, Jin Z, Chen Y (2020) Hybrid teaching-learning-based optimization and neural network algorithm for engineering design optimization problems. Knowl-Based Syst 187:104836
63. Kundu T, Garg H (2022) A hybrid TLNABC algorithm for reliability optimization and engineering design problems. Eng Comput. https://doi.org/10.1007/s10489-021-01572-8
64. Birashk A, Kazemi Kordestani J, Meybodi MR (2018) Cellular teaching-learning-based optimization approach for dynamic multi-objective problems. Knowl-Based Syst 141:148–177
65. Chen X, Mei C, Xu B, Yu K, Huang X (2018) Quadratic interpolation based teaching-learning-based optimization for chemical dynamic system optimization. Knowl-Based Syst 145:250–263
66. Wang D, Zhou Y, Jiang S, Liu X (2018) A simplex method-based salp swarm algorithm for numerical and engineering optimization. IFIP Adv Inf Commun Technol 538:150–159
67. Yang Z, Li K, Guo Y, Ma H, Zheng M (2018) Compact real-valued teaching-learning based optimization with the applications to neural network training. Knowl-Based Syst 159:51–62
68. Kundu T, Garg H (2021) A hybrid ITLHHO algorithm for numerical and engineering optimization problems. Int J Intell Syst 37(7):3900–3980. https://doi.org/10.1002/int.22707
69. Hussain K, Salleh MNM, Cheng S, Shi Y (2019) On the exploitation and exploitation in popular swarm-based metaheuristic algorithms. Neural Comput Appl 31(11):7665–7683
70. Gen M, Yun YS (2006) Soft computing approach for reliability optimization: State-of-the-art survey. Reliab Eng Syst Saf 91(9):1008–1026
71. Huang CL (2015) A particle-based simplified swarm optimization algorithm for reliability redundancy allocation problems. Reliab Eng Syst Saf 142:221–230
72. Kim HG, Bae CO, Park DJ (2006) Reliability-redundancy optimization using simulated annealing algorithms. J Qual Maint Eng 12(4):354–363
73. Valian E, Valian E (2013) A cuckoo search algorithm by Lavy flights for solving reliability redundancy allocation problems. Eng Optim 45(11):1273–1286
74. Garg H, Rani M, Sharma SP (2013) An efficient two-phase approach for solving reliability-redundancy allocation problem using artificial bee colony technique. Comput Op Res 40(12):2961–2969
75. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99
76. Liu Y, Qin G (2014) A modified particle swarm optimization algorithm for reliability redundancy optimization problem. J Comput 9(9):2024–2031
77. Ouyang HB, Gao LQ, Kong XY, Zou DX, Li S (2015) Teaching-learning based optimization with global crossover for global optimization problems. Appl Math Comput 265:533–556
78. Valian E, Tavakoli S, Mohanna S, Haghi A (2013) Improved cuckoo search for reliability optimization problems. Comput Ind Eng 64(1):459–468
79. Yeh WC, Hsieh TJ (2011) Solving reliability redundancy allocation problems using an artificial bee colony algorithm. Comput Oper Res 38(11):1465–1473
80. Zou D, Liu H, Gao L, Li S (2011) A novel modified differential evolution algorithm for constrained optimization problems. Appl Soft Comput J 62:736–767
81. Ghanbari S, Rahati A (2018) An improved artificial bee colony algorithm and its application to reliability optimization problems. Appl Soft Comput J 10(4):1188–1199
82. Ouyang HB, Gao LQ, Li S, Kong XY (2015) Improved novel global harmony search with a new relaxation method for reliability optimization problems. Inform Sci 305:14–55
83. He Q, Hu X, Ren H, Zhang H (2015) A novel artificial fish swarm algorithm for solving large-scale reliability-redundancy-application problem. ISA Trans 59:105–113
84. Afonso LD, Mariani VC, Dos Santos Coelho L (2013) Modified imperialist competitive algorithm based on attraction and repulsion concepts for reliability-redundancy optimization. Expert Syst Appl 40(9):3794–3802
85. Liu Y, Qin G (2015) A DE algorithm combined with levy flight for reliability redundancy allocation problems. Int J Hybrid Inf Technol 8(5):113–118
86. Rakhshani H, Rahati A (2017) Snap-drift cuckoo search: a novel cuckoo search optimization algorithm. Appl Soft Comput J 62:736–767
87. Derrac J, Garca S, Molina D, Herrera F (2011) 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
88. Mafarja M, Aljarah I, Heidari AA, Faris H, Fournier-Viger P, Li X, Mirjalili S (2018) Binarydragonfly optimization for feature selection using time-varying transfer functions. Knowl-Based Syst 187:104836
89. Sun G, Ma P, Ren J, Zhang A, Jia X (2018) A stability constrained adaptive alpha for gravitational search algorithm. Knowl-Based Syst 139:200–213
90. Yi J, Gao L, Li X, Shoemaker CA, Lu C (2019) An on-line variable-fidelity surrogate-assisted harmony search algorithm with multi-level screening strategy for expensive engineering design optimization. Knowl-Based Syst 170:1–19

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.