Applied Intelligence (2022) 52:12630–12667
https://doi.org/10.1007/s10489-021-02862-w

A hybrid salp swarm algorithm based on TLBO for reliability redundancy allocation problems

Tanmay Kundu[1] · **Deepmala**[1] · **Pramod K. Jain**[2]

Accepted: 17 September 2021 / Published online: 10 February 2022

Abstract
A novel optimization algorithm called hybrid salp swarm algorithm with teaching-learning based optimization (HSSATLBO) is proposed in this paper to solve reliability redundancy allocation problems (RRAP) with nonlinear resource constraints. Salp swarm algorithm (SSA) is one of the newest meta-heuristic algorithms which mimic the swarming behaviour of salps. It is an efficient swarm optimization technique that has been used to solve various kinds of complex optimization problems. However, SSA suffers a slow convergence rate due to its poor exploitation ability. In view of this inadequacy and resulting in a better balance between exploration and exploitation, the proposed hybrid method HSSATLBO has been developed where the searching procedures of SSA are renovated based on the TLBO algorithm. The good global search ability of SSA and fast convergence of TLBO help to maximize the system reliability through the choices of redundancy and component reliability. The performance of the proposed HSSATLBO algorithm has been demonstrated by seven well-known benchmark problems related to reliability optimization that includes series system, complex (bridge) system, series-parallel system, overspeed protection system, convex system, mixed series-parallel system, and large-scale system with dimensions 36, 38, 40, 42 and 50. After illustration, the outcomes of the proposed HSSATLBO are compared with several recently developed competitive meta-heuristic algorithms and also with three improved variants of SSA. Additionally, the HSSATLBO results are statistically investigated with the wilcoxon sign-rank test and multiple comparison test to show the significance of the results. The experimental results suggest that HSSATLBO significantly outperforms other algorithms and has become a remarkable and promising tool for solving RRAP.

Keywords Salp swarm algorithm · TLBO · Reliability redundancy allocation problem · Constrained optimization

1 Introduction

Since 1950, reliability optimization plays a progressively decisive role because of its critical requirements on several engineering and industrial applications, and has become a hot research topic in the engineering field. To be more competitive in daily life, the basic goal of a reliability engineer is always to improve the reliability of product components or manufacturing systems. Obviously, an excellent reliability design facilitates a system to run more safely and reliably. In general, reliability optimization problems can be classified into two classes: integer reliability problems (IRP) and mixed-integer reliability problems (MIRP). In IRP, the components reliability of the system is known and the main task is only to allocate the redundant components number. In case of MIRP, both the component reliability and the redundancy allocation of the system are to be designed simultaneously. This kind of problem in which the reliability of the system is maximized through the choices of redundancy and component reliability is also known as the reliability-redundancy allocation problem (RRAP). To optimize a RRAP, redundancy levels and component reliabilities of the system are considered as integer values and continuous values lies between zero and one respectively. Several researchers works on this field to solve RRAP with the objective of maximizing system reliability under constraints such as the system cost, volume, and weight etc., [44–48, 66, 76]. RRAP has been considered to be an NP-hard combinatorial optimization problem because of its complexity and it has been considered as

✉ Deepmala
dmrai23@gmail.com

Extended author information available on the last page of the article.

the subject of much prior research work over various optimization approaches. Recently, meta-heuristics algorithms (MHAs) have been successfully applied in dealing with the computational difficulty to solve a wide range of practical optimization problems. Solving optimization problems, these methods apply probabilistic rules and also approximate the optimal solution by using a random population in the search space that makes it more flexible to find better solutions compared to deterministic methods.

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 –

(i) Swarm intelligence algorithms: These types of algorithms mimic the behaviours or intelligence of animals or plants in nature, such as artificial bee colony (ABC) [7], ant colony optimization (ACO) [18], grey wolf optimization (GWO) [60], particle swarm optimization (PSO) [42], whale optimization algorithm (WOA) [59], bat algorithm (BA) [85], cuckoo search (CS) [86], jellyfish search (JS) [16], mayfly algorithm (MA) [92] and salp swarm algorithm (SSA) [58] etc.
(ii) Evolutionary algorithms: These algorithms mimic the mechanism of biological evolution, such as genetic algorithm (GA) [29], differential evolution (DE) [73], biogeography-based optimization (BBO) [71], and evolutionary programming (EP) [88] etc.
(iii) Human-related algorithms: These algorithms are inspired by some human nature/activities, such as passing vehicle search (PVS) [69], sine cosine algorithm (SCA) [57], teaching-learning based optimization (TLBO) [65], and coronavirus herd immunity optimizer (CHIO) [4] etc.

According to the "No Free Launch (NFL)" theorem [83], 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 another set of problems. Thus, no MHAs are perfect and its limitation affects the performance of the algorithm. Therefore, NFL provokes researchers to develop new MHAs or upgrade some original methods for solving a wider range of complex optimization problems (COPs). The hybridization of two algorithms is a remarkable and better choice between all strategies to upgrade an existing algorithm and to overcome shortcomings. In this process, two different operators are merged to get better solutions. For example, an improved HHO method named HHO-DE, based on the hybridization with DE algorithm is proposed for the multilevel image thresholding task by Bao et al., [11]. Later, Ibrahim et al., [38] present a hybrid optimization method that combines the salp swarm algorithm (SSA) with the particle swarm optimization for solving the feature selection problem. Again, in 2020, a hybrid method GNNA combining grey wolf optimization (GWO) and neural network algorithm (NNA) is proposed by Zhang [94]. Note that all above cited hybrid algorithms have been shown to be more competitive compared to the corresponding original methods. Considering the efficiency of the hybrid methods, this paper introduces a new hybrid algorithm based on SSA and TLBO to solve different kinds of reliability optimization problems.

Mirjalili et al. (2017) [58] first proposed an innovative population-based optimization method named salp swarm algorithm (SSA) that mimics the swarming behaviour of salps. The important characteristics like, simple structure, robustness, and scalability, makes SSA an efficient method for solving various kinds of real world problems (e.g., engineering design and optimization [80], feature selection [38], job shop scheduling [75], optimal power flow problem [22], parameter optimization of power system stabilizer [21], power generation [68], image segmentation [39, 84], parameter estimation for soil water retention curve [95], PID controller for AVR system [20], target localization [54]). Also, SSA shows the following outstanding features like: (1) It can be easily applied to different optimization problems without adjusting other parameters except population size and stopping criterion and it is worth mentioning that these parameters are essential for all MHAs; (2) It has a powerful neighbourhood search ability and it can easily fitted for wide search space [8]. Therefore, these advantages make SSA an efficient technique and a rapid growth of the SSA studies has also been noticed recently. Despite of these efficiency, the basic SSA has some major drawbacks in solving some optimization problems. Firstly, the SSA algorithm suffers from the problem of local optima stagnation. Secondly, the SSA experience is adequate in exploration but lacking of exploitation forces an improper balance between exploration and exploitation. Finally, SSA has poor convergence tendency and sometimes, it need more time to evaluate a new solution for some optimization problems. To address these issues, researchers have applied different search mechanisms and adopted modified operators to upgrade the original SSA. To mention a few- Gupta et al., [30] have introduced a new variant of the SSA called m-SSA. In this work, two different search strategies levy-fight search and opposition-based learning are utilized to increase the convergence speed and also, to establish an appropriate balance between exploration and exploitation. Abasi et al., [1] proposed a new hybrid algorithm named H-SSA combining the SSA and the $\beta-$hill climbing algorithm (βHC) to enhance the convergence speed as well as the local searching ability of the conventional SSA. A new type improved SSA based on inertia weight search mechanism

is introduced by Hegazy et al. [33] to maximize reliability, optimizing accuracy, and convergence speed. Kassaymeh et al., [41] have embedded backpropagation neural network (BPNN) in the salp swarm algorithm (SSA) to solve the software fault prediction (SFP) problem that helps to enhance the prediction accuracy. Sayed et al., [70] proposed a new hybrid algorithm CSSA that is based on the chaos theory and the basic SSA. Here, ten different types of chaotic maps are utilized to maximize the convergence speed and to get more accurate optimization results. A simplex method based salp swarm algorithm is introduced by Wang et al., (2018) [80] that improves the local searching ability of the algorithm. To maintain a proper balance between exploration and exploitation, Wu et al., [82] introduced an improved SSA based on a dynamic weight factor and an adaptive mutation searching strategy. Further, Tubishat et al., [77] proposed an improved version of SSA based on the concepts of opposition based learning and new local search strategy. These two improvement helps to enhance the exploitation capability of SSA. Singh et al., (2020) [72] developed a hybrid algorithm named HSSASCA that combines the sine-cosine algorithm (SCA) and the SSA to improve the convergence efficiency in both local and global search. Also, an enriched review of recent variants of SSA and its applications has been discussed by Abualigah et al., [3].

Apart from the SSA, inspired by the conventional teaching circumstances of the classroom, Rao et al., (2011) [65] proposed another significant algorithm for solving the optimization problems named 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 classroom teaching-learning environment is motivational process that allows students to improve their cognitive levels. After their appearance, several researchers [2, 12, 13, 15, 36, 62, 80, 87] used this algorithm to solve the real-world optimization problems. However, in order to solve large complex global optimization problems, it often falls to the local optimum. The main advantages of the TLBO algorithm is that without any effort for calibrating initial parameters, it leads to first convergence speed and also, the computational complexity of the algorithm is much better than several existing algorithms like GA, ABC, CS, SCA and PSO etc.

Motivated by the advantages of SSA and TLBO, a hybrid algorithm called HSSATLBO has been developed in this paper. Generally, searching processes with similar nature may lead to the loss of diversity in the search space, and also there is a chance of getting trapped into a local optimum. But, the different searching techniques of two different algorithms can maximize the capacity of escaping from the local optimal. In this algorithm, the basic structure of the SSA has been renovated by embedding the features of the TLBO. In this context, a probabilistic selection strategy is defined, which helps to determine whether to apply the basic SSA or the TLBO to construct a new solution. In the searching process, TLBO helps to accelerate the convergence speed of HSSATLBO, whereas the excellent global exploration ability of SSA helps to find a better global optimal solution. Therefore, in the search process of HSSATLBO, TLBO aims at the local search, and SSA accentuates the global search, which may help to maintain a convenient balance between exploration and exploitation and produces efficient and effective results for solving RRAP. Therefore, in this study, a population diversity definition of the proposed method is introduced and also, performed the exploration-exploitation evaluation for investigating the search behaviour of both HSSATLBO and the conventional SSA. To reduce the computational complexity and improve the searching abilities, HSSATLBO can reduce its population by keeping the diversity too low. The measurement of exploration and exploitation also helps to identify how the proposed HSSATLBO performs better on an optimization problem. Keeping all of the above points in mind, the basic objective of the study is to present an efficient and effective algorithm to solve various types of reliability optimization problems. The main contributions of the paper are listed as follows:

- A hybrid algorithm HSSATLBO is developed by combining the features of SSA and TLBO algorithms. Proposed algorithm mainly contains the structure of the basic SSA algorithm and, meantime, it has been reconstructed by embedding the searching strategy of TLBO.
- The proposed method makes a proper balance between exploration and exploitation in which the basic SSA looks after the exploration part and the presence of the searching strategy of TLBO increases the exploitation capability of the algorithm. Again, to generate a new solution, a new probabilistic selection strategy is introduced to determine whether to apply the original SSA or the TLBO algorithm.
- To validate the effectiveness and efficiency of the HSSATLBO algorithm, it is examined against seven well-known reliability redundancy optimization problems [9, 14, 23, 24, 28, 36, 43, 51, 56, 63, 78, 81, 89]. For a fair comparison, the test problems are also examined by the conventional SSA and its three different variants (LSSA, CSSA and GSSA). Finally, a comparative study between HSSATLBO and the three different variants of SSA are also performed in this study.
- In order to state the statistically significant results or not, a number of tests have been carried out, such as rank-tie, Wilcoxon-rank test, Kruskal Wallis test, and multiple

comparison tests, on the results obtained from the proposed and the existing algorithms. From these computed results, it is verified that the proposed algorithm produces an effective result and also delivers superior performance compared to other existing algorithms in terms of the best optimal solutions.

The rest of the papers is organized as follows: Section 2 briefly describes the traditional SSA, three variants of SSA (i.e., LSSA, CSSA & GSSA), and TLBO algorithms. Section 3 presents the proposed algorithm HSSATLBO, a probabilistic selection procedure, and exploration-exploitation measurement. The reliability redundancy allocation problems are described in Section 4. Section 5 presents the computational results of the proposed HSSATLBO algorithm and compares them with several existing algorithms. The results obtained have also been validated through statistical test analysis. Finally, Section 6 concludes the paper.

2 Basis algorithms

In this section, basic SSA, three improved variants of SSA, and the conventional TLBO algorithms are briefly described.

2.1 Salp swarm algorithm (SSA)

Salp swarm algorithm (SSA) is one of the population based algorithm recently developed by Mirjalili et al.,(2017) [58] to solve numerous kinds of optimization problems. Salp is a kind of marine creature which belongs to the Salpidae family and has a thin barrel-shaped body with openings at the end in which the water is pumped through the body as propulsion to move forward. These marine creature shows an interesting behaviour which is of interest in the paper is their swarming behaviour. In oceans, Salps having a swarm behaviour called salp chain may support salps in exploring and a better movement may be possible using fast cordial changes.

Based on this conduct, a mathematical form for the salp chains is designed by the authors and examined in optimization problems. Firstly, the population of salp is divided into two groups: leaders and followers. The first salp of the chain is known as the leader, and rest of them are called the followers. The position of all salps (N_p) is stored in a two-dimensional matrix X given in equation (1). These salps looking for a food source that implies the target of the swarm.

$$X = \begin{pmatrix} x_1^1 & x_2^1 & \ldots & x_d^1 \\ x_1^2 & x_2^2 & \ldots & x_d^2 \\ \ldots & \ldots & \ldots & \ldots \\ x_1^{N_p} & x_2^{N_p} & \ldots & x_d^{N_p} \end{pmatrix} \tag{1}$$

Then the salp with the best fitness (i.e., leader) is find out by calculating the fitness value of each salp. The position of the leader should be refurbished in regular basis, so the following equation (2) is proposed-

$$x_j^1 = \begin{cases} F_j + D_1((ub_j - lb_j)D_2 + lb_j), & D_3 \geq 0.5 \\ F_j - D_1((ub_j - lb_j)D_2 + lb_j), & D_3 < 0.5 \end{cases} \tag{2}$$

Where x_j^1 is the position of the first salp (leader) in the j^{th} dimension and F_j is the food position in the j^{th} dimension. lb_j and ub_j represents the lower and upper bound of the j^{th} dimension respectively. D_1, D_2 and D_3 are random numbers lies between 0 and 1.

Equation (2) shows that the leader only updates its position concerning the food source. Here, D_1 is a very important parameter in this algorithm as it plays a vital role in balancing the exploration and exploitation phase and the is determined by the equation (3)

$$D_1 = 2\exp\left(-\left(\frac{4it}{T}\right)^2\right) \tag{3}$$

Where T and it represents the maximum number of iterations and the current iteration respectively. The parameter D_2 and D_3 controlled both the direction and the step size of the j^{th} dimension of the next position.

After updating the leader's position, the follower's position is updated using equation (4).

$$x_j^i = \frac{1}{2}\lambda t^2 + \mu_0 t \tag{4}$$

Where, $i \geq 2$, x_j^i shows the position of the i^{th} follower salp in the j^{th} dimension, μ_0 is the initial speed, t is the time and $\lambda = \frac{\mu_{final}}{\mu_0}$, where $\mu = \frac{(x - x_0)}{t}$.

In optimization, the time indicates the iteration, so the disparity between iterations is considered as 1 and, taking $\mu_0 = 0$, the following equation (5) is applied for this problem.

$$x_j^i = \frac{1}{2}(x_j^i + x_j^{i-1}), i \geq 2 \tag{5}$$

In may happen that some salps cross the search space, but, using equation (6) they can be bring back to the search space.

$$x_j^i = \begin{cases} lb_j & \text{if} x_j^i \leq lb_j \\ ub_j & \text{if} x_j^i \geq ub_j \\ x_j^i & \text{,otherwise} \end{cases} \tag{6}$$

The detailed steps of the basic SSA are explained in Algorithm 1.

Algorithm 1 Salp Swarm Algorithm (SSA).

1. Initialization: Maximum Iterations, Population size (N_p)
2. Initialize a random salp population x^i for $i = 1, 2, \ldots, N_p$
3. Evaluate the fitness value of the population
4. F = The global best solution
5. **repeat**
6. Update D_1 using (23)
7. **for** each slap $i \in N_p$
8. **if** $i == 1$
9. Update the position of leading salp using (2)
10. **else**
11. Update the position of follower salp using (5)
12. **end if**
13. **end for**
14. Update the position of salps based on their upper and lower bounds using (6)
15. **Until** (stop condition = false)
16. Return the best solution F.

2.2 Improved variants of SSA with mutation strategies

Although the conventional SSA is a highly competitive and effective method for solving different kinds of complex optimization problems, it may trap into local optima and also, suffers from the improper balance between exploration and exploitation which encounters slow convergence. To get rid of this difficulty and to explore the solution space more adequately, Nautiyal et al., (2021) [61] introduced improved variants of SSA named LSSA, CSSA, and GSSA using three new mutation operators Lévy flight based mutation, Cauchy mutation, and Gaussian mutation respectively to enhance the overall performance of SSA. These different mutation operators make the algorithm more competent in exploring and exploiting the search space. In these improved SSA, the mutation scheme is performed after the greedy search is completed between two consecutive position $x_j^i(t)$ at t^{th} iteration and $x_j^i(t+1)$ at $(t+1)^{th}$ iteration corresponding to each salp which is given by the (7).

$$x_j^{i,new} = \begin{cases} x_j^i(t) & \text{if} f(x_j^i(t)) < f(x_j^i(t+1)) \\ x_j^i(t+1) & \text{if} f(x_j^i(t)) > f(x_j^i(t+1)) \end{cases} \tag{7}$$

After the completion of the greedy search for each salp, the mutation strategy is performed with a mutation rate m_r. In this study, the value of this parameter m_r is taken as 0.7. During this process, the fitness value of the newly generated muted salps are compared with the original salps and if it found better, it replaces the original salps otherwise discarded. The detailed pseudo-code of the mutation-based SSA is presented in Algorithm 2.

I. *Lévy flight based SSA (LSSA):* The concept of lévy flight based mutation is used to increase salps diversity in the SSA. When mutation rate allows, lévy-mutation can improve the global search ability more adequately by mutating the salps. Each muted salp in the LSSA is generated using (8) as follows

$$\widehat{x}_j^i = x_j^i \times (1 + LF(\delta)) \tag{8}$$

where $LF(\delta)$ corresponds to lévy distributed random number with δ variable size and that can be obtained using (9)

$$LF(\delta) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \quad \sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right)^{\frac{1}{\beta}} \tag{9}$$

where u and v are standard normal distribution. β is a default constant set to 1.5.

II. *Cauchy-SSA (CSSA):* In this Cauchy-SSA, a random number is generated, and if its value allows to generate the new salps using the mutation scheme based on the mutation rate m_r, then each muted salp of the swarm in CSSA is generated using the (10) as follows

$$\widehat{x}_j^i = x_j^i \times (1 + Cauchy(\delta)) \tag{10}$$

where Cauchy(δ) is a random number generated using the Cauchy distribution function given by the (11) as follows

$$y = \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{\alpha}{\eta}\right) \tag{11}$$

and the Cauchy density function is given by

$$f_{cauchy(0,\eta)}(\alpha) = \frac{1}{\pi} \frac{\eta}{\eta^2 + \alpha^2} \tag{12}$$

where, y is a uniformly distributed random number within (0,1) and η = 1 is a scale parameter.

III. *Gaussian-SSA (GSSA):* In GSSA, the mutation follows the (13)

$$\widehat{x}_j^i = x_j^i \times (1 + Gaussian(\delta)) \tag{13}$$

where Gaussian(δ) is a random number generated using the Gaussian distribution and the Gaussian density function is given by (14)

$$f_{gaussian(0,\sigma^2)}(\beta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{\alpha^2}{2\sigma^2}} \tag{14}$$

where σ^2 is a variance for each salp. To generate random numbers, the above equation is reduced by taking standard deviation σ as 1.

Algorithm 2 Pseudo-code of LSSA, CSSA and GSSA.

1. Initialize Maximum iterations, mutation rate m_r, Population size (N_p)
2. Initialize a random salp population x^i for $i = 1, 2, \ldots, N_p$
3. **repeat**
4. Calculate the fitness value of each salp and select the best one
5. **for** (each salp x^i in the population N_p) **do**
6. Update D_1 using (3)
7. Update the position of leading salp using (2)
8. Update the position of follower salp using (5)
9. Update the position of salps based on their upper and lower bounds using (6)
10. Apply the greedy search corresponding to each salp position
11. **if** $rand < m_r$ **then**
12. **for** (each salp x^i in the population N_p) **do**
13. Apply one of the following mutation strategy to generate mutant salp
14. Update the position of salps using (8) ▷ **LSSA**
15. Update the position of salps using (10) ▷ **CSSA**
16. Update the position of salps using (13) ▷ **GSSA**
17. Accept the newly generated mutant salp if it gives better fitness otherwise discarded.
18. **Until** (stop condition = false)
19. **Return** the salp with best fitness value

2.3 Teaching-Learning Based Optimization (TLBO)

Rao et al., (2011) [65] first developed an algorithm like other population-based algorithms, which reproduced 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 -

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). In this phase, the basic intention of a teacher is to transfer knowledge among the students and also to improvise the average result of the class. Here, the parameter $Mean_j(t)$ indicates the mean result of the learners in j^{th} subject ($j = 1, 2, \ldots, d$) and at generation t, it is given by (15).

$$Mean_j(t) = [M_1^T, M_2^T, \ldots, M_d^T] \tag{15}$$

$X_{Teacher}(t)$ 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 equation (16).

$$\Delta_j^m(t) = rand[X_{Teacher}(t) - T_F \cdot Mean_j(t)] \tag{16}$$

where $rand$ indicates a random number lies between 0 and 1. T_F denotes the teaching factor and its value is decided randomly as given in equation (17)

$$T_F = round(1 + rand) \tag{17}$$

The existing solution is now updated in the teacher phase and the updated solution is given by the following equation (18)

$$X_j^{m,new}(t) = X_j^m(t) + \Delta_j^m(t) \tag{18}$$

If the new learner $X_j^{m,new}(t)$ in generation t is found to be a better than $X_j^m(t)$, then it will replace $X_j^m(t)$ otherwise keeps the previous solution.

Interaction with other students is an effective way to enhance their knowledge. A learner can also gain new information from other learners having more knowledge than him or her. The learning circumstances of the learner phase is given below.

A student X^m randomly select classmate X^n ($\neq X^m$) to obtain more knowledge in the learner phase. If X^n performs better, X^m moves towards X^n ; if X^n performs worse, X^m moves away from it. The following formulas (19) and (20) can be used to describe this process:

$$X_j^{m,new}(t) = X_j^m(t) + rand(X_j^n(t) - X_j^m(t)), \quad if \ \ f(X_j^n(t) < f(X_j^m(t)) \tag{19}$$

$$X_j^{m,new}(t) = X_j^m(t) + rand(X_j^m(t) - X_j^n(t)), \quad if \ \ f(X_j^n(t)) > f(X_j^m(t)) \tag{20}$$

Where $f(X_j^n(t))$ and $f(X_j^m(t))$ are fitness values of $X_j^n(t)$ and $X_j^m(t)$ respectively. The pseudocode of the basic TLBO is given in Algorithm 3.

Algorithm 3 Teaching-Learning based optimization (TLBO).

1. **Initialization**: Maximum Iterations, Population size (N_p)
2. Initialize a random population including N_p solutions x^m for $m = 1, 2, \ldots, N_p$
3. Evaluate the fitness value of each solution.
4. Select the global best solution as the teacher ($X_{Teacher}$)
6. **repeat**
7. **for** each individual $m \in N_p$
5. Calculate the mean result $Mean_j(t)$ according to (15)
8. Perform the teacher phase according to (16) and (18)
9. Calculate the fitness value of individual x^m.
10. Replace the old solution x_j^m with the new one $x_j^{m,new}$ if $x_j^{m,new}$ is better than x_j^m.
11. **end for**
12. **for** each individual $m \in N_p$
13. Perform the learner phase according to (19) and (20)
14. Calculate the fitness value of individual x^m.
15. Evaluate the fitness and accept the new solution if it gives a better result than
the old one.
16. **end for**
17. Select the optimal solution as the teacher ($X_{Teacher}$)
18. **Until** (stop condition = false)
19. Post process results and visualization

3 Proposed method

3.1 Probabilistic selection procedure

It is very important to maintain a proper balance between exploration and exploitation for a well-organized and well-designed meta-heuristics optimization algorithm. Therefore, in this study, a hybrid algorithm HSSATLBO is introduced by modifying the basic structure of SSA. A probabilistic selection parameter (PSP) is implemented in the proposed algorithm to decide whether to apply the search equation of SSA (Algorithm 1) or TLBO (Algorithm 3) to generate the new solution. The formula for the parameter PSP is given by equation (21)

$$PSP = PSP_{max} \cdot \exp\left(\frac{ln(\frac{PSP_{min}}{PSP_{max}})}{MaxIter} iter\right) \tag{21}$$

Here, PSP_{min} and PSP_{max} denotes the minimum and maximum values of the parameter PSP, respectively. Again, $MaxIter$ indicates the maximum number of generations, and $iter$ shows the current generation. Equation (21) shows that, at the early stage of iterations, the value of the parameter PSP is very large, and its force to choose the search equation of SSA. However, as the number of iterations increases, the probability of electing the search equation of TLBO is also increased. In this study, PSP_{min} and PSP_{max} takes the value as 0.3 and 0.9 respectively, i.e., the value of the parameter PSP is a random number lies between [0.3, 0.9].

3.2 The proposed HSSATLBO

The framework of the proposed algorithm HSSATLBO that associates with the SSA and the TLBO algorithm, is demonstrated in this section. The conventional SSA algorithm shows excellent efficiency in exploration but undergoes poor exploitation, and as a result, it fails to manage the convenient balance between exploitation and exploration and also, most of the time it cannot generate a global optimum solution. To avoid this situation, the updating phase of the salps position is enhanced by reconstructing the basic formation of the SSA. During this modification the searching mechanism of TLBO is implemented into the main structure of the SSA. 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 adds more flexibility to the SSA and subsequently, the exploration and exploitation abilities of the SSA algorithm are also improved. The detailed framework of the HSSATLBO algorithm is presented in Fig. 1. The first step in the HSSATLBO is to initialize the parameters for both the SSA and TLBO and a random population is generated that represents a set of salp positions. Then the fitness value for each solution is computed to evaluate the performance and the best one is determined. After that, the current population of both the leader and follower position is to be updated either by using the searching technique of SSA or TLBO algorithm depending on a probabilistic selection parameter (PSP) (Section 3.1). This parameter is basically designed to control the probability of selecting the above searching strategies. If a random number lies between [0,1] is less than PSP, then the SSA, otherwise, the TLBO is used for updating the current salps position. After that, the fitness value of the current population is evaluated and the current best solution is compared with the previous best fitness value, and accordingly the best solution is need to be updated. This procedure is continued until the stopping criterion is satisfied. For the proposed HSSATLBO algorithm the maximum iteration number is considered as a stopping criterion.

3.3 Exploration and exploitation measurement

In this study, an in-depth empirical analysis is performed to examine the searching behaviour of the proposed HSSATLBO in terms of diversity. Through diversity measurement, it is possible to measure explorative and

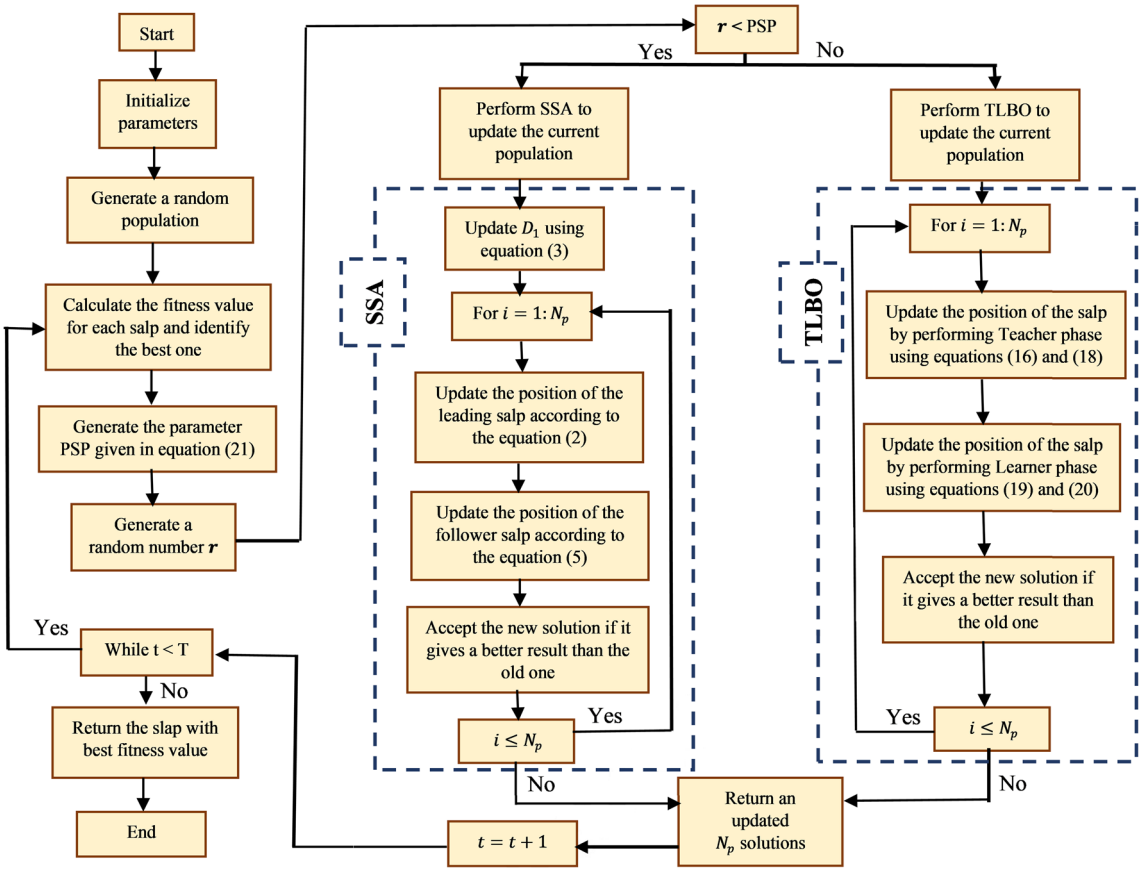


Fig. 1 The framework of HSSATLBO

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 this two abilities is a trade-off problem. For better illustration about the exploration and exploitation concept, see Fig. 2 . According to Hussain [37], diversity in population is measured mathematically, using the following equations (22) and (23):

$$Div_j^t = \frac{\sum_{i=1}^{N_p} \left[med(x_j^t) - x_j^{i,t} \right]}{N_p} \tag{22}$$

$$Div^t = \frac{\sum_{j=1}^{D} Div_j^t}{D} \tag{23}$$

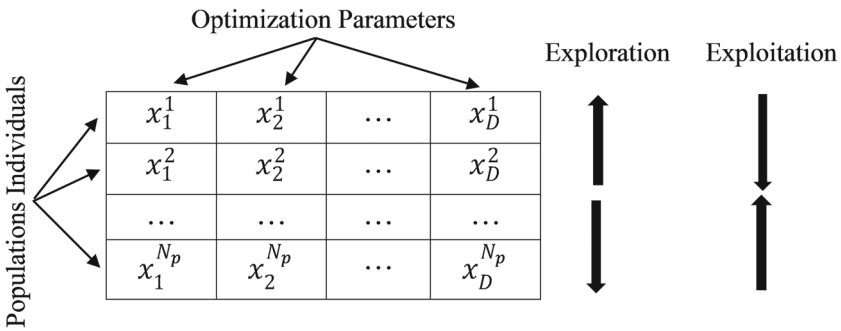


Fig. 2 Candidate population representation for exploration-exploitation

Where, $x_j^{i,t}$ 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^t for all the iterations, it is now possible to calculate the exploration and exploitation percentage ratios during search process, using equation (24) and equation (25) respectively:

$$Expl(\%) = \frac{Div^t}{Div_{max}^t} \times 100 \tag{24}$$

$$Expt(\%) = \frac{|Div^t - Div_{max}^t|}{Div_{max}^t} \times 100 \tag{25}$$

where Expl(%) and Expt(%) denotes exploration and exploitation percentages respectively for iteration t, whereas Div_{max}^t is the maximum population diversity in all iterations (T).
The MATLAB code for measuring population diversity and exploration-exploitation for MHAs has been made publicly available at https://github.com/usitsoft/Exploration-Exploitation-Measurement.

4 Problem formulation

Notations

n	$= (n_1, n_2, \dot{...}, n_m)$, the redundancy allocation vector for the system.
m	number of subsystems.
n_i	the number of components in subsystem i.
${n_i}^{max}$	maximum 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.
v_i	the component volume in subsystem i.
w_i	the component weight in subsystem i.
R_i	$= 1 - (1 - r_i)^{n_i}$, is the reliability of the i^{th} subsystem.
C	upper limit of the system's cost.
V	upper limit of the system's volume.
W	upper limit of the system's weight.
R_S	the system reliability.
g_j	the j^{th} constraint function.

4.1 Reliability-redundancy allocation problem

The requirement of reliability analysis to evaluate the performance of products, equipment, and several engineering systems is increasing day by day. Reliability optimization can figure out these issues and capable of finding a high-quality products and equipment that performs efficiently and safely in a given period. In this section, seven reliability optimization problems are discussed to examine the performance of the HSSATLBO algorithm.
The general form of the reliability redundancy problem is

$$\begin{aligned} &\max \quad R_S(r_1, r_2, ..., r_m; n_1, n_2, ..., n_m) \\ &s.t, \quad g(r_1, r_2, ..., r_m; n_1, n_2, ..., n_m) \le b, \\ &0.5 \le r_i \le 1, \quad 1 \le n_i \le {n_i}^{max}, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., m. \end{aligned} \tag{26}$$

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.

4.1.1 Series system [Fig. 3(a)] [6, 9, 23, 27, 31, 35, 36, 43, 81, 89, 91, 93]

The series system is a non-linear mixed-integer programming problem and the formulation is given as follows

$$\max R_S(n) = \prod_{i=1}^{5} R_i(n_i) \tag{27}$$

$$s.t, \quad g_1(r, n) = \sum_{i=1}^{5} v_i n_i^2 - V \le 0, \tag{28}$$

$$g_2(r, n) = \sum_{i=1}^{5} c_i [n_i + \exp\left(\frac{n_i}{4}\right)] - C \le 0, \tag{29}$$

$$g_3(r, n) = \sum_{i=1}^{5} w_i n_i \exp\left(\frac{n_i}{4}\right) - W \le 0, \tag{30}$$

$0.5 \le r_i \le 1, \quad 1 \le n_i \le 5, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., 5.$

where, $c_i = \alpha_i \left(\frac{-1000}{ln(r_i)}\right)^{\beta_i}, \quad i = 1, 2, ..., 5.$
The parameters β_i and α_i are physical features of system components. Constraints $g_1(r, n)$, $g_2(r, n)$, and $g_3(r, n)$ represents volume, cost and weight constraint respectively. The coefficients of the series system are shown in the literature (Garg, 2015a)[23] and **Table 1**.

4.1.2 Complex (bridge) system [Fig. 3(b)] [6, 9, 23, 25, 27, 31, 35, 36, 43, 63, 81, 89, 91, 93, 96, 97]

Complex (bridge) system consists of five subsystems and the formulation of it is described as follows:

$$\max R_S(r, n) = R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 - R_1R_2R_3R_4 - R_1R_2R_3R_5 - R_1R_2R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5$$

subject to, the same constraint given by the equation equation (28), (29) and (30) respectively. And also,

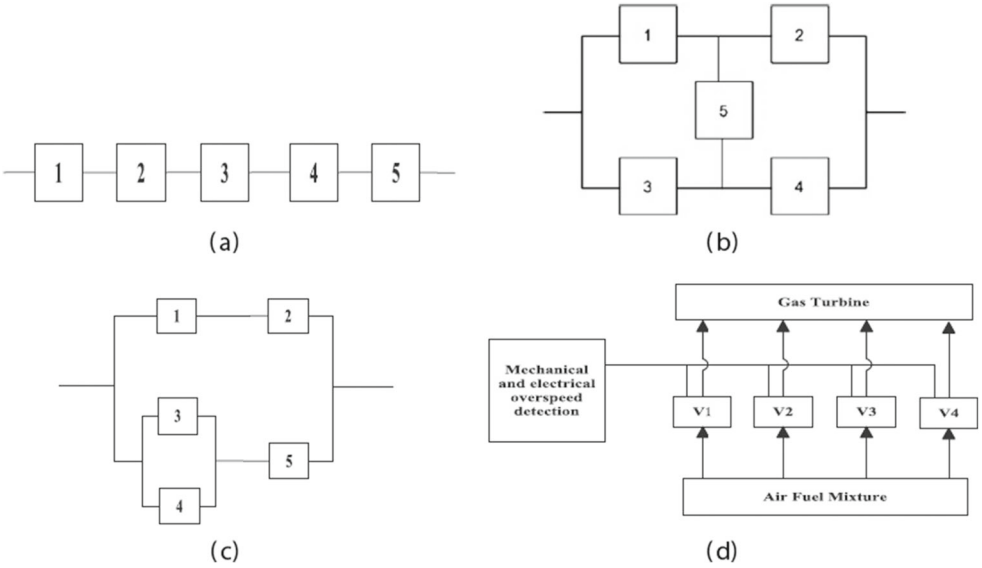


Fig. 3 The schematic diagram of (a) series system, (b) complex (bridge) system, (c) series-parallel system, and (d) overspeed system for a gas turbine

$0.5 \le r_i \le 1, \quad 1 \le n_i \le 5, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., 5.$
The coefficients of the complex system are shown in the literature (Garg, 2015a) [23] and Table 1.

4.1.3 Series-Parallel System [Fig. 3(c)] [23, 25, 27, 31, 32, 35, 36, 40, 43, 51, 53, 63, 78, 79, 81, 89, 91]

The mathematical formulation is as follows:

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

subject to, the same constraint given by the equation equation (28),(29) and (30) respectively. And also, $0.5 \le r_i \le 1, \quad 1 \le n_i \le 5, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., 5.$

The coefficients of the series-parallel system are shown in the literature (Garg, 2015a) [23] and Table 1.

4.1.4 Overspeed protection system for a gas turbine [Fig. 3(d)] [6, 19, 23, 24, 35, 36, 43, 51, 53, 63, 81, 93, 98]

This reliability problem is formulated as follows:

$$\max R_S(r, n) = \prod_{i=1}^{4} R_i(n_i) \tag{31}$$

$$s.t, \quad g_1(r, n) = \sum_{i=1}^{4} v_i n_i^2 - V \le 0, \tag{32}$$

Table 1 Values of parameters used in the literature

i	$10^5\alpha_i$	β_i	v_i	w_i	C	V	W
Parameter used for 4.1.1 and 4.1.2							
1	2.330	1.5	1	7			
2	1.450	1.5	2	8			
3	0.541	1.5	3	8	175	110	200
4	8.050	1.5	4	6			
5	1.950	1.5	2	9			
Parameter used for 4.1.3							
1	2.500	1.5	2	3.5			
2	1.450	1.5	4	4.0			
3	0.541	1.5	5	4.0	175	180	100
4	0.541	1.5	8	3.5			
5	2.100	1.5	4	3.5			
Parameter used for 4.1.4							
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			

$$g_2(r,n) = \sum_{i=1}^{4} c_i[n_i + \exp\left(\frac{n_i}{4}\right)] - C \leq 0, \tag{33}$$

$$g_3(r,n) = \sum_{i=1}^{4} w_i n_i \exp\left(\frac{n_i}{4}\right) - W \leq 0 \tag{34}$$

$0.5 \leq r_i \leq 1, \quad 1 \leq n_i \leq 5, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., 5.$

where, $c_i = \alpha_i \left(\frac{-1000}{ln(r_i)}\right)^{\beta_i}, \quad i = 1, 2, ..., 4.$

The coefficients of the overspeed protection system are shown in the literature (Garg, 2015a) [23] and Table 1.

4.1.5 Convex quadratic reliability problem [28, 50, 63, 91]

The mathematical formulation of this problem is as follows:

$$\max R_S(r,n) = \prod_{i=1}^{10}(1 - (1 - r_i)^{n_i}) \tag{35}$$

$$s.t, \quad g_j(r,n) = \prod_{i=1}^{10}(a_{ji}{n_i}^2 + C_{ji}n_i) \leq b_j \tag{36}$$

$n_i \in [1, 6], \quad i = 1, 2, ..., 10. \quad j = 1, 2, 3, 4.$

The parameters r_i, a_{ji} and C_{ji} are generated from uniform distributions that lies between [0.80, 0.99], [0,10] and [0,10] respectively. A randomly generated set of values of these coefficients are given as follows:

$r_i = [\ 0.81,\ 0.93,\ 0.92,\ 0.96,\ 0.99,\ 0.89,\ 0.85,\ 0.83,\ 0.94,\ 0.92];$

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

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

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

4.1.6 Mixed series-parallel system [28, 50, 63, 91]

The mathematical formulation of this problem is as follows:

$$\max R_S(r,n) = \prod_{i=1}^{15}(1 - (1 - r_i)^{n_i}) \tag{37}$$

$$s.t, \quad g_1(r,n) = \sum_{i=1}^{15} c_i n_i - 400 \leq 0 \tag{38}$$

$$g_2(r,n) = \sum_{i=1}^{15} w_i n_i - 414 \leq 0 \tag{39}$$

$n_i \geq 1, \quad n_i \in \mathbf{Z}^+, \quad i = 1, 2, ..., 15.$

Table 2 Parameter used for 4.1.6

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

The coefficients of the mixed series-parallel system are taken from the literature (Gen et al., 1999) [26] and are listed in Table 2.

4.1.7 Large-scale system reliability problem [27, 28, 63, 78, 79, 81, 97]

The mathematical formulation of this problem is as follows:

$$\max R_S(r,n) = \prod_{i=1}^{m} R_i(n_i) \tag{40}$$

$$s.t,\ g_1(r,n) = \sum_{i=1}^{m} \alpha_i n_i^2 - \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{m} \alpha_i l_i^2 \ \le 0 \tag{41}$$

$$g_2(r,n) = \sum_{i=1}^{m} \beta_i \exp\left(\frac{n_i}{2}\right) - \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{m} \beta_i \exp\left(\frac{l_i}{2}\right) \le 0 \tag{42}$$

$$g_3(r,n) = \sum_{i=1}^{m} \gamma_i n_i - \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{m} \gamma_i l_i \ \le 0 \tag{43}$$

$$g_4(r,n) = \sum_{i=1}^{m} \delta_i \sqrt{n_i} - \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{m} \delta_i \sqrt{l_i} \ \le 0 \tag{44}$$

$1 \le n_i \le 10,\ \ n_i \in \mathbf{Z}^+,\ \ i = 1, 2, ..., m.$

Here, l_i indicates the lower bound of n_i. The parameter θ 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 $\sum_{i=1}^{m} g_{ji}(l_i)$, $(j = 1, ..., 4)$ and the average values of which is given by $b_j = \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{m} g_{ji}(l_i)$. In this way, we set the available system resources (Zou et al., 2010) [97] for reliability systems with 36, 38, 40, 42, and 50 subsystems, respectively, as shown in Tables 3 and 4.

5 Results & discussions

In this section, we presented the results of all of the above-mentioned reliability optimization problems identified by the use of the proposed HSSATLBO algorithm. This section is divided into the following six 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 5. The performance

Table 3 Available system resources for each system for 4.1.7

n	i	1	2	3	4
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

Table 4 Constant coefficients for 4.1.7

i	$1-r_i$	α_i	β_i	γ_i	δ_i	i	$1-r_i$	α_i	β_i	γ_i	δ_i	i	$1-r_i$	α_i	β_i	γ_i	δ_i	i	$1-r_i$	α_i	β_i	γ_i	δ_i
1	0.005	8	4	13	26	11	0.028	6	5	14	28	21	0.030	6	2	15	30	31	0.021	7	5	15	28	41	0.023	10	5	17	33
2	0.026	10	4	16	32	12	0.021	10	3	15	30	22	0.027	6	2	12	24	32	0.023	9	5	11	22	42	0.040	8	3	18	35
3	0.035	10	4	12	23	13	0.039	9	1	17	34	23	0.018	7	2	20	40	33	0.030	6	3	15	29	43	0.012	8	1	18	35
4	0.029	6	3	12	24	14	0.013	10	4	20	39	24	0.013	8	5	19	38	34	0.026	7	3	14	27	44	0.026	6	4	19	38
5	0.032	7	1	13	26	15	0.038	7	4	14	28	25	0.006	9	5	15	29	35	0.009	6	5	15	29	45	0.038	6	4	13	26
6	0.003	10	4	16	31	16	0.037	10	2	13	25	26	0.029	8	1	18	35	36	0.019	10	5	17	33	46	0.015	8	1	19	37
7	0.020	9	2	19	38	17	0.021	10	1	15	29	27	0.022	8	3	16	32	37	0.005	9	5	19	37	47	0.036	7	4	14	28
8	0.018	9	3	15	29	18	0.023	8	3	19	38	28	0.017	9	3	15	29	38	0.019	10	5	11	22	48	0.032	10	2	19	37
9	0.004	7	4	12	23	19	0.027	10	5	18	36	29	0.002	10	1	18	35	39	0.002	6	2	17	34	49	0.038	8	3	15	30
10	0.038	6	4	16	31	20	0.028	7	4	13	26	30	0.031	9	2	19	37	40	0.015	8	3	17	33	50	0.013	10	2	11	22

Table 5 Some existing meta-heuristic algorithms from the literature for solving reliability optimization problems (4.1.1 to 4.1.7)

	Algorithms	Methods	Authors & published year
1.	SCa	Soft computing approach	(Gen and Yun, 2006) [27]
2.	SAA	Simulated annealing algorithm	(Kim et al., 2006) [43]
3.	GA	Genetic algorithm (GA)	(Yokota et al., 1996) [91]
4.	IA	Immune based two-phase approach	(Hsieh and You, 2011) [35]
5.	ABC1	Artificial bee colony algorithm	(Yeh and Hsieh, 2011) [89]
6.	IPSO	Improved particle swarm optimization	(Wu et al., 2011) [81]
7.	CS1	Cuckoo search (CS) algorithm	(Valian and Valian, 2013) [79]
8.	CS2	Cuckoo search algorithm	(Garg, 2015a) [23]
9.	PSO/SSO/PSSO	Particle-based swarm optimization algorithm	(Huang, 2015) [36]
10.	ICS	Improved CS algorithm	(Valian et al., 2013) [78]
11.	CS-GA	Hybrid CS and genetic algorithm	(Kanagaraj et al., 2013) [40]
12	ABC2	Artificial bee colony	(Garg et al., 2013) [25]
13.	TS-DE	Hybrid TS–DE algorithm	(Liu and Qin, 2014b)[52]
14.	INGHS	Improved novel global harmony search	(Ouyang et al., 2015) [63]
15.	MPSO	Modified particle swarm optimization	(Liu and Qin, 2014a) [51]
16.	EBBO	Efficient biogeography-based optimization	(Garg, 2015b) [24]
17.	EGHS	Effective global harmony search algorithm	(Zou et al., 2011) [96]
18.	NMDE	Novel modified DE	(Zou et al., 2011) [98]
19.	NGHS	Novel global HS algorithm	(Zou et al., 2010) [97]
20.	CPSO	Co-evolutionary PSO	(He and Wang, 2007a) [32]
21.	IABC	Improved ABC algorithm	(Ghambari and Rahati,2018) [28]
22.	NAFSA	Novel artificial fish swarm algorithm	(He et al., 2015) [31]
23.	MICA	Modified imperialist competitive algorithm	(Afonso et al., 2013) [6]
24.	GA-SRS	RRAP with cold-standby redundancy strategy	(Ardakan and Hamadani,2014) [9]
25.	LXPM-IPSO-GS	IPSO-based hybrid approache	(Zhang et al.,2013) [93]
26.	PSFSA	Penalty guided stochastic fractal search approach	(Mellal and Zio, 2016) [56]
27.	GA-PSO	Hybrid GA-PSO approach	(Duan et al., 2010) [19]
28.	DE	DE algorithm combined with levy flight	(Liu and Qin, 2015) [53]
29.	HDE	Hybrid DE algorithm	(Liao, 2010) [50]
30.	NNA	Neural network algorithm	(Sadollah et al., 2018) [67]
31.	HHO	Harris hawks optimization	(Heidari et al., 2019) [34]
32.	SMA	Slime mould algorithm	(Li et al., 2020) [49]
33.	SCA	Sine cosine algorithm	(Mirjalili, 2016) [57]

comparisons between HSSATLBO, SSA, and variants of SSA are presented in Section 5.3. A parameter sensitivity analysis for the parameter PSP is performed in Section 5.4. The performance in terms of population diversity and the exploation-exploitation measurement of HSSATLBO, SSA, and variants of SSA are described in Section 5.5. Finally, the statistical analysis of the proposed algorithm and all compared algorithms are illustrated in section Section 5.6.

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. The initial population sizes of ABC, NNA, TLBO, SSA, HHO, SMA, SCA and HSSATLBO were set as 100 for each and also the parameters of these compared algorithms are considered as: ABC (Maximum number of trials i.e., limit = 100), NNA (modification factor, β = 1), TLBO (teaching factor, TF = 1 or 2), HHO (β = 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 HSSATLBO for solving every reliability optimization problems. In our experiment, for every independent run, the maximum number of iterations for each algorithm is taken as 300.

Table 6 Comparison of the best result for the series system (4.1.1) with other results in the literature

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SCa	(3, 2, 2, 3, 3)	0.779427	0.869482	0.902674	0.714038	0.786896	0.931680	27	1.21454E-01	7.518918	3.4940E-03
SAA	(3, 2, 2, 3, 3)	0.777143	0.867514	0.896696	0.717739	0.793889	0.931363	27	0.00000	7.518918	4.6533E-01
GA	(3, 2, 2, 3, 3)	0.782391	0.866712	0.901747	0.717266	0.783795	0.931460	27	5.3194E-02	7.518918	3.2446E-01
IA	(3, 2, 2, 3, 3)	0.77946230	0.87188345	0.90280087	0.71135016	0.78786158	0.93168234	27	5.284E-07	7.518918	6.8943E-05
ABC1	(3, 2, 2, 3, 3)	0.779399	0.871837	0.902885	0.711403	0.787800	0.931682	27	2.1836E-04	7.518918241	5.6662E-04
IPSO	(3, 2, 2, 3, 3)	0.78037307	0.87178343	0.90240890	0.71147356	0.78738760	0.931680	27	1.0100E-04	7.518918	3.4940E-03
CS2	(3, 2, 2, 3, 3)	0.779439734	0.871995212	0.902873050	0.711127088	0.787986374	0.931682106	27	4.42986E-07	7.518918241	4.1061E-04
PSO	(2, 3, 2, 4, 2)	0.80059281	0.74049316	0.82914384	0.63686144	0.88704276	0.8885037	4	16.7757	4.8146147	3.8727E+01
NAFSA	(3, 2, 2, 3, 3)	0.779388413	0.871720982	0.903033391	0.711418362	0.787789288	0.931682268	27	6.7347E-09	7.518918	1.7353E-04
SSO	(3, 2, 2, 3, 3)	0.78271484	0.8735199	0.90264893	0.71313477	0.77729797	0.93150199	27	1.82140E-03	7.51891824	2.6336E-01
PSSO	(3, 2, 2, 3, 3)	0.77946645	0.87173278	0.90284951	0.71148780	0.78781644	0.93168229721	27	4.9081E-5	7.51891824	1.3157E-04
MICA	(3, 2, 2, 3, 3)	0.779874	0.872057	0.903426	0.710960	0.786902	0.93167939	27	9.9E-05	7.518918	4.3868E-03
GA-SRS	(3, 2, 2, 3, 3)	0.76459335	0.88752892	0.91539527	0.69350544	0.77603145	0.929082263568	27	2.2694E-05	7.5189182411	3.6664E+00
LXPM-IPSO-GS	(3, 2, 2, 3, 3)	0.779509	0.871859	0.902891	0.711345	0.787739	0.93168236	27	2.20E-07	7.518918	3.9668E-05
HSSATLBO	(3, 2, 2, 3, 3)	0.779382894	0.871833757	0.902885037	0.711416829	0.7877965964	0.93168238710	27	4.949952767E-07	7.5189182411	–

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 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 HSSATLBO and is expressed by the (45)

$$MPI = \frac{R_S(HSSATLBO) - R_S(Others)}{1 - R_S(Others)} \tag{45}$$

Where $R_S(HSSATLBO)$ denotes the best optimal solution obtained by the proposed algorithm and $R_S(Others)$ implies the best result obtained by the other compared approaches and the greater MPI indicates greater improvement.

5.2 HSSATLBO comparison with existing optimizers

This section describes the performance evaluation of proposed HSSATLBO 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 Table 6 to Table 11.

For the series system (4.1.1), Table 6 shows that the best optimal solution obtained by the proposed method is 0.93168238710, which is preferable to all compared algorithms SCA(Gen & Yun, 2006), SAA (Kim et al., 2006), GA (Yokota et al., 1996), IA (Hsieh & You, 2011), ABC1 (Yeh & Hsieh, 2011), IPSO (Wu et al., 2011), CS2 (Garg, 2015a), PSO (Huang, 2015), NAFSA (He et al., 2015), SSO (Huang, 2015), PSSO (Huang, 2015), MICA (Afonso et al., 2013), GA-SRS (Ardakan & Hamadani, 2014), and LXPM-IPSO-GA (Zhang et al., 2013) with the improvements 3.4940E-03%, 4.6533E-01%, 3.2446E-01%, 6.8943E-05%, 5.6662E-04%, 3.4940E-03%, 4.1061E-04%, 3.8727E+01%, 1.7353E-04%, 2.6336E-01%, 1.3157E-04%, 4.3868E-03%, 3.6664E+00%, and 3.9668E-05% respectively.

It can be observed from Table 7 that the optimal solution for the complex system (4.1.2) produced by HSSATLBO is 0.9998896373815054 which is better than the best result given by the other compared algorithms and also have most symbolic improvement 4.7596E+01%, 1.7777E+00%, 8.6705E+00%, 2.5927E-01%, 6.6414E+01%, 9.1343E-01%, and 2.5695E+00%, over the results given by SCA

Table 7 Comparison of the best result for the complex system (4.1.2) with other results in the literature

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SCa	(3, 3, 2, 3, 2)	0.814483	0.821383	0.896151	0.713091	0.814091	0.9997894	18	1.854075	4.264770	4.7596E+01
SAA	(3, 3, 3, 3, 1)	0.868116	0.807263	0.872862	0.712667	0.751034	0.99988764	40	0.007300	1.609289	1.7777E+00
GA	(3, 3, 3, 3, 1)	0.814090	0.864614	0.890291	0.701190	0.734731	0.99987916	18	0.376347	4.264770	8.6705E+00
NGHS	(3, 3, 2, 4, 1)	0.82983999	0.85798911	0.91333926	0.64674479	0.70310972	0.99988960	5	0.00000594	1.56046629	3.3860E-02
IA	(3, 3, 3, 3, 1)	0.81662417	0.86876739	0.85874878	0.71027937	0.75342920	0.9998893505	18	4.0420871E-08	4.264770	2.5927E-01
EGHS	(3, 3, 2, 4, 1)	0.82983999	0.85798911	0.91333926	0.64674479	0.70310972	0.99988960	5	0.00000594	1.56046629	3.3860E-02
ABC1	(3, 3, 2, 4, 1)	0.828087	0.857805	0.704163	0.648146	0.914240	0.99988962	5	25.43392577	1.56046628	1.5747E-02
IPSO	(3, 3, 2, 4, 1)	0.82868361	0.85802567	0.91364616	0.64803407	0.70227595	0.99988963	5	0.00000359	1.56046629	6.6880E-03
ABC2	(3, 3, 2, 4, 1)	0.827970276	0.857874758	0.914186404	0.648355386	0.703575311	0.999889635809	5	3.7463676E-04	1.56046628	1.4248E-03
INGHS	(3, 3, 2, 4, 1)	0.8279847911	0.8576796813	0.9141564522	0.6484814055	0.7048654988	0.9998896364	5	0.00000189	1.56046628	8.8934E-04
CS2	(3, 3, 2, 4, 1)	0.82785565	0.8576261054	0.914752916	0.648217208	0.702670374	0.9998896319	5	1.06721E-10	1.5604662	4.8959E-03
EBBO	(3, 3, 2, 4, 1)	0.8280606892	0.8580404545	0.9141487486	0.6479689012	0.7042048796	0.9998896364	5	1.4541E-04	1.560466	8.8934E-04
PSO	(3, 3, 2, 2, 3)	0.77061588	0.90109253	0.89278651	0.60083008	0.73451002	0.99967140	37	16.54571312	1.41180322	6.6414E+01
NAFSA	(3, 3, 2, 4, 1)	0.8283217918	0.8579745073	0.9142209882	0.6477571701	0.7030066618	0.9998896360	5	1.5485E-05	1.56046629	1.2427E-03
PSSO	(3, 3, 2, 4, 1)	0.82783292	0.85771241	0.91437458	0.64861002	0.70287554	0.999889635738	5	2.502902E-05	1.560466288	1.4891E-03
SSO	(3, 3, 2, 4, 1)	0.82008362	0.85119629	0.91854858	0.66072083	0.70275879	0.99988862	5	0.00437599	1.560466288	9.1343E-01
GA-SRS	(3, 3, 3, 3, 1)	0.80457234	0.85717305	0.86734683	0.72759162	0.76416666	0.999886726842	18	6.7785491E-05	4.264769804	2.5695E+00
LXPM-IPSO-GS	(3, 3, 3, 3, 1)	0.827974	0.857818	0.914166	0.648348	0.704427	0.999889636852	5	3.351252E-05	1.560466288	4.7989E-04
MICA	(3, 3, 2, 4, 1)	0.82764257	0.85747845	0.91419677	0.64927379	0.70409200	0.99988963	5	0.00004428	1.56046629	6.6880E-03
HSSATLBO	(3, 3, 2, 4, 1)	0.8280051677	0.8578130972	0.9142533044	0.6482662731	0.7038807118	0.9998896373815054	5	1.1074633E-06	1.56046628802	–

Table 8 Comparison of the best result for the series-parallel system (4.1.3) with other results in the literature

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SCa	(2, 2, 2, 2, 4)	0.785452	0.842998	0.885333	0.917958	0.870318	0.99997418	40	1.194440	1.609289	4.7085E+01
GA	(3, 3, 1, 2, 3)	0.838193	0.855065	0.878859	0.911402	0.850355	0.99996875	53	0.00000	7.110849	5.6280E+01
SAA	(2, 2, 2, 2, 4)	0.819596	0.845000	0.895514	0.895519	0.868456	0.99997665	40	0.000007	1.609289	4.1488E+01
ABC1	(2, 2, 2, 2, 4)	0.819591561	0.844951068	0.895428548	0.895522339	0.868490229	0.999976649036	40	5.9845376E-04	1.609288966	4.1490E+01
IA	(2, 2, 2, 2, 4)	0.812161	0.853346	0.897597	0.900710	0.866316	0.99997631	40	0.007300	1.609289	4.2327E+01
CPSO	(2, 2, 2, 2, 4)	0.81918526	0.84366421	0.89472992	0.89537628	0.86912724	0.99997664	40	0.000561	1.609289	4.1513E+01
CS1	(2, 2, 2, 2, 4)	0.819927087	0.845267657	0.895491554	0.895440692	0.868318775	0.999976649	40	0.0000161	1.6092890	4.1490E+01
ICS	(2, 2, 2, 2, 4)	0.819927087	0.845267657	0.895491554	0.895440692	0.868318775	0.999976649	40	0.0000161	1.6092890	4.1490E+01
CS-GA	(2, 2, 2, 2, 4)	0.819660256	0.844981615	0.895519305	0.895492245	0.868447587	0.99997665	40	0.000000017	1.60928897	4.1488E+01
IPSO	(2, 2, 2, 2, 4)	0.8197457	0.8450080	0.8954581	0.9009032	0.8684069	0.99997731	40	1.469522338	1.609288966	3.9786E+01
ABC2	(2, 2, 2, 2, 4)	0.819737753469	0.844991099776	0.895529543820	0.895433687206	0.868434824469	0.999976649054	40	1.39152E-10	1.609288966	4.1490E+01
TS-DE	(2, 2, 2, 2, 4)	0.819659	0.844981	0.895507	0.895506	0.868448	0.9999766491	40	2.66935542E-04	1.609288966	4.1490E+01
INGHS	(2, 2, 2, 2, 4)	0.8198118626	0.8449506842	0.8956701585	0.8952327069	0.868438057445	0.9999766489	40	0.00005305414	1.609288966	4.1490E+01
CS2	(2, 2, 2, 2, 4)	0.819483232488	0.844783084455	0.895810553887	0.895220216915	0.868542486973	0.999976648818	40	2.7216628E-10	1.609288966	4.1491E+01
MPSO	(2, 2, 2, 2, 4)	0.81965932	0.84498074	0.89550642	0.89550643	0.86844775	0.9999766490660	40	1.961642794E-07	1.609288966	4.1490E+01
EBBO	(2, 2, 2, 2, 4)	0.8196583448	0.8449101406	0.8954871713	0.8955148963	0.8684681613	0.9999766490488	40	1.748541669E-05	1.609288966	4.1490E+01
PSO	(4, 3, 2, 1, 2)	0.84025282	0.88865099	0.62375055	0.93984950	0.75158691	0.99985845	68	0.916915088	4.017703641	9.0348E+01
PSFS	(2, 2, 2, 2, 4)	0.81965939118	0.84498085296	0.89550643076	0.89550645172	0.86844769346	0.9999766490661	40	1.85082E-10	1.6092889667	4.1490E+01
NAFSA	(2, 2, 2, 2, 4)	0.81978757527	0.84567194372	0.89486836331	0.89590826856	0.86829583055	0.999976648004	40	3.1248E-08	1.609289	4.1493E+01
PSSO	(2, 2, 2, 2, 4)	0.81958939	0.84458412	0.89534134	0.89581626	0.86852902	0.9999766487381	40	8.1179461E-05	1.609288966	4.1491E+01
SSO	(2, 2, 2, 2, 4)	0.81385803	0.83912659	0.89366150	0.89845276	0.87106323	0.99997657	40	0.0024	1.609288966	4.1687E+01
DE	(2, 2, 2, 2, 4)	0.81965932	0.84498074	0.89550642	0.89550643	0.86844775	0.9999766490660	40	1.96164279E-07	1.609288966	4.1490E+01
IABC	(3, 3, 2, 2, 3)	0.827743712027	0.847138611794	0.856891035304	0.856634348041	0.875976531093	0.999979829614	38	0.0000183067	0.705901612	3.2264E+01
HSSATLBO	(3, 2, 2, 2, 4)	0.7753618512628	0.8714241422773	0.8903702230415	0.8914438741116	0.8630261550595	0.9999863373757	30	1.26363261E-07	1.794965001	–

Table 9 Comparison of the best result for the overspeed protection system (4.1.4) with other results in the literature

Algorithms	(x_1, x_2, x_3, x_4)	r_1	r_2	r_3	r_4	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SAA	(5, 5, 5, 5)	0.895644	0.885878	0.912184	0.887785	0.999945	50	0.9380	28.8037	1.759E+01
IA	(5, 5, 4, 6)	0.901588628	0.888192380	0.948166022	0.849969792	0.99995467455	55	1.249537E-04	15.3634630	2.421E-04
IPSO	(5, 5, 4, 5)	0.90163164	0.84997020	0.94821828	0.88812885	0.99995467	55	0.000009	24.081883	1.029E-02
NMDE	(5, 6, 4, 5)	0.90161480	0.84992111	0.94814139	0.88822286	0.99995467	55	1.057E-05	24.80188272	1.029E-02
TS-DE	(5, 6, 4, 5)	0.901615	0.849921	0.948141	0.888223	0.9999546746081	55	1.896705E-04	24.80188272	1.240E-04
INGHS	(5, 5, 4, 6)	0.901556583	0.888243885	0.948111097	0.849981737	0.9999546743	55	0.00005054	24.801882722	8.038E-04
CS2	(5, 5, 4, 6)	0.901598077	0.888226184	0.948101861	0.849980778	0.999954674	55	8.824940E-10	15.36346308	1.750E-04
EBBO	(5, 5, 4, 6)	0.90156292	0.88822494	0.94815595	0.849952895	0.999954674	55	2.7021E-05	15.3634631	1.419E-04
PSO	(4, 6, 5, 5)	0.92952331	0.81370356	0.88663747	0.89987183	0.99990474	37	11.5265677	11.6447077	5.242E+01
PSSO	(5, 5, 4, 6)	0.90166461	0.88817296	0.94821033	0.84987084	0.99995467	55	3.28722E-05	15.3634630	1.029E-02
SSO	(5, 6, 4, 5)	0.90208435	0.85472107	0.94606018	0.88633728	0.99995416	55	0.109233104	24.8018827	1.123E+00
LXPM-IPSO-GS	(5, 5, 4, 6)	0.90163317	0.888251065	0.948141377	0.849854043	0.999954674599	55	5.305493E-06	15.36346308	1.423E-04
DE	(5, 6, 4, 5)	0.90161482	0.84992114	0.94814139	0.88822284	0.99995467	55	1.005073E-05	24.80188272	1.029E-02
MICA	(5, 5, 4, 5)	0.90148988	0.85003526	0.94812952	0.88823833	0.999954673	55	0.00213782	24.8018827	3.672E-03
GA-PSO	(5, 5, 4, 6)	0.901628	0.888230	0.948121	0.849921	0.99995467	55	0.000006	15.363463	1.029E-02
HSSATLBO	(5, 6, 4, 5)	0.901623877	0.849936249	0.948146758	0.888204712	0.99995467466432	55	4.96040115E-07	24.80188272	–

Table 10 Comparison of the best result for the Convex quadratic (4.1.5) and Mixed series-parallel system (4.1.6) with other results in the literature

Problems	Methods	n	$R_S(r, n)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	$Slack(g_4)$
4.1.5	GA	(2, 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)	0.80884418963273	0.9649307648E+13	0.0202998323E+13	4.3632406548E+13	0.0871498795E+13
	IABC	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.80884418963273	0.9649307648E+13	0.02029983232E+13	4.36324065484E+13	0.0871498795E+13
	HSSATLBO	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.8088441896327347	9.649307648E+12	2.029983232E+11	4.36324065484E+13	8.71498795E+11
4.1.6	GA	(3, 4, 5, 3, 3, 2, 4, 5, 4, 3, 3, 4, 5, 5, 5)	0.9202	–	–	–	–
	HDE	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.945613	–	–	–	–
	INGHS	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.94561335745814	8	0	–	–
	IABC	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.94561335745814	8	0	–	–
	HSSATLBO	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.9456133574581371	8	0	–	–

(Gen & Yun, 2006), SSA(Kim et al., 2006), GA (Yokota et al., 1996), IA (Hsieh & You, 2011), PSO/SSO (Huang, 2015), and GA-SRS (Ardakan & Hamadani, 2014) respectively.

Table 8 presents that the best result for the series-parallel system (4.1.3) obtained by the proposed method is 0.9999863373757 and also better than the algorithms given by SCA (Gen & Yun, 2006), SAA (Kim et al., 2006), GA (Yokota et al., 1996), IA (Hsieh & You, 2011), ABC1 (Yeh & Hsieh, 2011), IPSO (Wu et al., 2011), CPSO (He & Wang, 2007a), CS1 (Valian & Valian, 2013), ICS (Valian et al., 2013), CS-GA (Kanagaraj et al., 2013), ABC2 (Garg et al., 2013), TS-DE (Liu & Qin, 2014), INGHS (Ouyang et al., 2015), CS2 (Garg, 2015a), MPSO (Liu & Qin, 2015), EBBO (Garg, 2015b), PSO (Huang, 2015), PSFSA (Mellal & Zio, 2016), NAFSA (He et al., 2015), PSSO (Huang, 2015), SSO (Huang, 2015), DE (Liu & Qin, 2015), IABC(Ghambari & Rahati, 2018) by the improvement 4.7085E+01%, 4.1488E+01%, 5.6280E+01%, 4.2327E+01%, 4.1490E+01%, 3.9786E+01%, 4.1513E+01%, 4.1490E+01%, 4.1490E+01%, 4.1488E+01%, 4.1490E+01%, 4.1490E+01%, 4.1490E+01%, 4.1491E+01%, 4.1490E+01%, 4.1490E+01%, 9.0348E+01%, 4.1490E+01%, 4.1493E+01%, 4.1491E+01%, 4.1687E+01%, 4.1490E+01%, and 3.2264E+01% respectively. Also, the optimal redundant component by HSSATLBO for this sires-parallel system is (3,2,2,2,4) which is completely different from the other approaches.

It can be noticed in Table 9, the best solution achieved by HSSATLBO for the overspeed protection system (4.1.4) is 0.99995467466432. The proposed algorithm dominates 15 competitive algorithms in terms of the best-known solution found so far. Table 9 depicts that the proposed method has symbolic improvement indices 1.759E+03%, 2.421E-02%, 1.029E+00%, 1.029E+00%, 1.240E-02%, 8.038E-02%, 1.750E-02%, and 1.419E-02% over the results by SAA (Kim et al., 2006), IPSO (Wu et al., 2011), NMDE (Zou et al., 2011), PSO (Huang, 2015), PSSO (Huang, 2015), SSO (Huang, 2015), DE (Liu & Qin, 2015) and GA-PSO (Duan et al., 2010) respectively. Table 10 indicates that HSSATLBO executes the same or better than the other existing algorithms given in this literature for solving the convex quadratic reliability problem (4.1.5) and the mixed series-parallel system (4.1.6) in terms of best results. Table 11 reports the test results of the problem (4.1.7). It can be seen that the HSSATLBO algorithm gives equal or better results compare to other algorithms in terms of the best objective function value for the large-scale problems of dimensions 36,38,40,42 & 50. But in the case of dimension 40, it comes with weaker objective value than two existing algorithms INGHS and IABC.

In order to show the convergence performance of the stated algorithm over several existing algorithms like ABC, NNA, TLBO, SSA etc, we vary the best solution for each considered problem and the results are plotted in Fig. 4. This analysis shows that the HSSATLBO has a better convergence rate compared to other algorithms.

5.3 HSSATLBO comparison with other variants of SSA

This section details about the comparative study of results for the conventional SSA, the proposed HSSATLBO and the three SSA variants namely Levy flight based SSA (LSSA), Cauchy salp swarm algorithm (CSSA) and Gaussian salp swarm algorithm (GSSA). The results are presented in Tables 12–16 in terms of best obtained value and the MPI values. Table 12 shows that the best optimal solution obtained by HSSATLBO for solving series system (4.1.1) is better than the original SSA and the three variants LSSA, CSSA and GSSA with the improvements 1.2529E-07%, 2.7509E-06%, 1.2566E-06%, and 6.8942E-07% respectively. It can be observed from Table 13, the optimal solution achieved by HSSATLBO for solving the complex system (4.1.2) is better than the compared algorithms SSA, LSSA, CSSA and GSSA with significant improvement percentage 2.6556E-03%, 2.3872E-03%, 3.2317E-04%, and 1.1439E-05% respectively. Again, in case solving the series-parallel system (4.1.3) and overspeed system (4.1.4), Tables 14 and 15 shows that, HSSATLBO dominated all compared algorithms effectively with the best optimal value as well as in MPI values. Table 16 shows that HSSATLBO executes the same or better optimal value with the other compared algorithms in case of solving both convex system (4.1.5) and mixed series-parallel system (4.1.6).

Again, the convergence graphs of HSSATLBO are also compared with LSSA, CSSA and GSSA for solving problems 4.1.1 to 4.1.6 and are given in Fig. 5. From these convergence graphs, we can conclude that, as the iteration number increases, the proposed HSSATLBO algorithm also shows better performance than the existing algorithm.

5.4 Parameter sensitivity analysis

In this section, parameter sensitivity analysis is performed to evaluate the impact of the probabilistic parameter PSP on the proposed algorithm. Under other conditions retained, different values of the parameter PSP are tested on reliability problems (4.1.1 - 4.1.6) and the results are presented in Table 17. PSP1 indicates that PSP_{min} and PSP_{max} takes the value 0.05 and 0.95 respectively, i.e., PSP1 lies between [0.05, 0.95]. Similarly, PSP2, PSP3, PSP4 and PSP5 are lies between [0.1, 0.9], [0.2, 0.9], [0.3, 0.9] and [0.4, 0.9] respectively. The mean values obtained

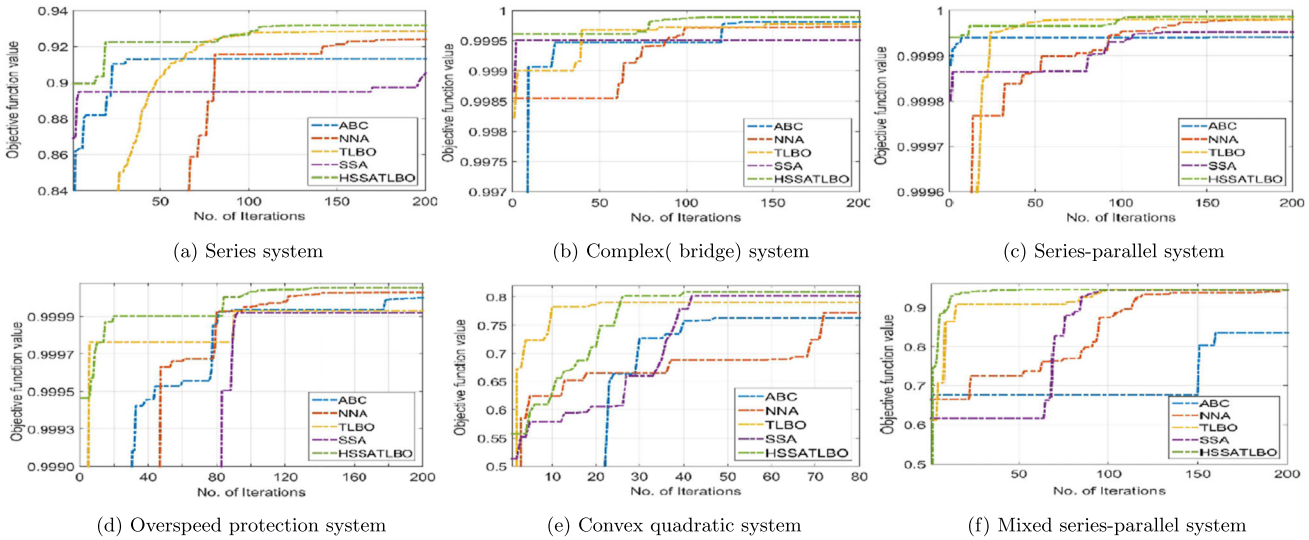
Table 11 Comparison of results for Large Scale systems (4.1.7) with other results in the literature

Dim	Methods	VTV	$R_S(r,n)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	$Slack(g_4)$
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)	0.51997596538026	1	49.125763519460	109	301.353247018274
	IABC	(5, 10, 15, 21, 33)	0.5199759653802567	1	49.125763519460179	109	301.35324701827426
	HSSATLBO	(5, 10, 15, 21, 33)	0.519975965380256	1	49.12576351946018	109	291.3532470182740
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	(10,13,15,21 ,33)	0.51098860	–	–	–	–
	INGHS	(10,13,15,21 ,33)	0.51098859649712	1	53.638550812459	115	317.039538519290
	IABC	(10,13,15,21 ,33)	0.5109885964971198	1	53.6385508124589020	115	317.039538519289640
	HSSATLBO	(10,13,15,21 ,33)	0.5109885964971198	1	53.6385508124589020	115	317.039538519289640
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, 15, 33)	0.5032926	–	–	–	–
	CS1	(4, 10, 11, 21, 22, 33)	0.50599242	–	–	–	–
	INGHS	(4, 10, 11, 21, 22, 33)	0.505992421241597	0	51.04714167016368	119	333.24054864606615
	IABC	(4, 10, 11, 21, 22, 33)	0.5059924212415972	0	51.047141670163683	119	333.24054864606615
	HSSATLBO	(5, 10, 13, 15, 33)	0.5032924930631358	3	58.53406557447610	128	330.2821792064087

Table 11 (continued)

Dim	Methods	VTV	$R_S(r,n)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	$Slack(g_4)$
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)	0.47966355148656	2	52.718250389045	129	354.583694396574
	IABC	(4, 10, 11, 15, 21, 33)	0.4796635514865568	2	52.7182503890448400	129	354.583694396573720
	HSSATLBO	(4, 10, 11, 15, 21, 33)	0.4796635514865568	2	52.7182503890448400	129	354.583694396573720
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)	0.40695474513707	0	61.955982588824	154.0	433.914646838262
	INGHS	(4, 10, 15, 21, 33, 42, 45)	0.40695474513707	0	61.955982588824	154.0	433.914646838262
	IABC	(4, 10, 15, 21, 33, 42, 45)	0.4069547451370713	0	61.9559825888243270	154.0	433.914646838261660
	HSSATLBO	(4, 10, 15, 21, 33, 42, 45)	0.4069547451370713	0	61.9559825888243270	154.0	433.914646838261660

Here, VTV denotes the variables that received the value 2 in optimum stage



(a) Series system (b) Complex(bridge) system (c) Series-parallel system

(d) Overspeed protection system (e) Convex quadratic system (f) Mixed series-parallel system

Fig. 4 Comparison of convergence curves of HSSATLBO with existing optimizers

by HSSATLBO and their corresponding ranking for each cases are given in that table. Also, as per their achievement in terms of mean value, we can sort their ranking, in the order: PSP4, PSP5, PSP3, PSP2, and PSP1. From the ranking order in Table 17, it can be observed that the result of the proposed algorithm is superior when PSP lies between [0.3, 0.9] i.e., for the case of PSP4. Researchers can also choose different values for PSP according to other set of problems. Figure 6 provides a better visualization of the ranking for each cases of HSSATLBO for solving reliability optimization problems.

5.5 Diversity and exploration-exploitation analysis

For an effective in-depth performance analysis, the population diversity and the exploration-exploitation measurement in HSSATLBO, SSA and SSA variants (i.e., LSSA, CSSA and GSSA) are presented in Table 18 while solving reliability optimization problems. A graphical presentation on comparison of diversity measurement between the proposed HSSATLBO and the SSA variants are given in Fig. 7. The exploration-exploitation phases of the proposed algorithm is also given in Fig. 8. According to Table 18, the proposed hybrid method mostly reduced population diversity compared to SSA, LSSA, CSSA and GSSA for all the reliability problems. For example, on series system (4.1.1), HSSATLBO maintained population diversity value 0.12618 which is relatively lesser than diversity values 1.17742, 0.64185, 0.65301, and 0.65110 in SSA, LSSA, CSSA and GSSA respectively. Similarly, diversity measurement in HSSATLBO for all other problems (4.1.2 - 4.1.6) remained lower than original SSA and its variants. Moreover, Table 18 also reveals that mostly HSSATLBO maintained exploration percentage lower than exploitation on all of the reliability problems. For instance, HSSATLBO maintained exploitation percentage as 81% 88% 72% 85% 67% and 65% for series, complex, series-parallel, overspeed, convex and mixed series-parallel system respectively; and these values are higher than exploitation measurements recorded for the compared algorithms. This discussion can be further assimilated via Fig. 7 for diversity measurement and Fig. 8 for exploration and exploitation behaviours in the proposed algorithm.

5.6 Statistical analysis

In addition, to analyze whether or not the results obtained by the proposed HSSATLBO algorithm are statistically significant, here we consider the following quality indices described below:

5.6.1 The statistical results by Value-based method and tied ranking

The solution quality in terms of standard deviation and mean value is described here. The lower mean value and standard deviation indicates that the algorithm has a stronger global optimization capability and more stability. Also, Tied rank (TR) (Rakhshani & Rahati, 2017) [64] is used here to compare intuitively the performance between the considered methods. In this study, the algorithm with the best mean value is assigned to rank 1; the second-best

Table 12 Comparison of the best result for the series system (4.1.1) with SSA variants

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SSA	(3, 2, 2, 3, 3)	0.77946493	0.87181012	0.90289318	0.71139277	0.78778637	0.93168237854	27	8.9782981E-06	7.5189182	1.2529E-07
LSSA	(3, 2, 2, 3, 3)	0.77932026	0.87186227	0.90273431	0.71160561	0.78767127	0.93168219916	27	6.4467091E-06	7.51891824	2.7509E-06
CSSA	(3, 2, 2, 3, 3)	0.77959711	0.87181360	0.90292374	0.71134023	0.78769105	0.93168230125	27	7.5055857E-06	7.51891824	1.2566E-06
GSSA	(3, 2, 2, 3, 3)	0.77938544	0.87191850	0.90286873	0.71130499	0.78786502	0.93168234000	27	9.8571517E-06	7.51891824	6.8942E-07
HSSATLBO	(3, 2, 2, 3, 3)	0.779382894	0.871833757	0.902885037	0.711416829	0.7877965964	0.93168238710	27	4.9499527E-07	7.5189182411	–

Table 13 Comparison of the best result for the complex system (4.1.2) with SSA variants

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SSA	(3, 3, 3, 3, 1)	0.81596668	0.86859325	0.85853828	0.71089508	0.75535162	0.999889343515	18	2.4358456E-07	4.2647698	2.6556E-03
LSSA	(3, 3, 2, 4, 1)	0.82903789	0.85736946	0.91436033	0.65047059	0.67604199	0.999889373288	5	5.7066553E-06	1.5604662	2.3872E-03
CSSA	(3, 3, 2, 4, 1)	0.82831707	0.85747467	0.91437566	0.64931145	0.69435926	0.999889601704	5	2.8128656E-06	1.5604662	3.2317E-04
GSSA	(3, 3, 2, 4, 1)	0.82816367	0.85782025	0.91432496	0.64813395	0.70237521	0.999889636119	5	1.3054309E-06	1.5604662	1.1439E-05
HSSATLBO	(3, 3, 2, 4, 1)	0.82800516	0.85781309	0.91425330	0.64826627	0.70388071	0.9998896373815	5	1.1074633E-06	1.5604662	–

Table 14 Comparison of the best result for the series-parallel system (4.1.3) with SSA variants

Algorithm	$(x_1, x_2, x_3, x_4, x_5)$	r_1	r_2	r_3	r_4	r_5	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SSA	(3, 2, 2, 2, 4)	0.77494709	0.87147146	0.89201825	0.89022079	0.86300277	0.999986336949	30	2.4360319E-06	1.7949650	3.1230E-05
LSSA	(3, 2, 2, 2, 4)	0.77509882	0.87027453	0.89110882	0.89228253	0.86319730	0.999986335794	30	5.4134949E-07	1.7949650	1.1575E-04
CSSA	(3, 2, 2, 2, 4)	0.77585180	0.87166330	0.89161410	0.89004200	0.86283118	0.999986336878	30	2.9153687E-07	1.7949650	3.6426E-05
GSSA	(3, 2, 2, 2, 4)	0.77104400	0.86680377	0.89522995	0.89169037	0.86470216	0.99998630333	30	3.4680749E-06	1.7949650	2.4856E-03
HSSATLBO	(3, 2, 2, 2, 4)	0.77536185	0.871424142	0.890370223	0.891443874	0.863026155	0.9999863373757	30	1.2636326E-07	1.7949650	–

Table 15 Comparison of the best result for the overspeed protection system (4.1.4) with SSA variants

Algorithms	(x_1, x_2, x_3, x_4)	r_1	r_2	r_3	r_4	$R_S(R)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	MPI(%)
SSA	(5, 6, 4, 5)	0.90158641	0.84985533	0.94815323	0.88826987	0.9999546746609	55	3.6119283E-06	2.4801882E+01	2.3015E-06
LSSA	(5, 5, 4, 6)	0.90164911	0.88819858	0.94814849	0.84991872	0.999954674643	55	1.5440297E-06	1.5363463E+01	4.7037E-07
CSSA	(5, 5, 4, 6)	0.90167880	0.88823897	0.94810473	0.84987261	0.99995467454	55	5.2320308E-07	1.5363463E+01	2.7428E-06
GSSA	(5, 6, 4, 5)	0.90159816	0.84992802	0.94815898	0.88821623	0.9999546746613	55	4.4184018E-06	2.4801882E+01	6.6630E-08
HSSATLBO	(5, 6, 4, 5)	0.901623877	0.849936249	0.948146758	0.888204712	0.99995467466432	55	4.96040115E-07	2.480188272E+01	–

Table 16 Comparison of the best result for the Convex quadratic (4.1.5) and Mixed series-parallel system (4.1.6) with SSA variants

Problems	Methods	n	$R_S(r,n)$	$Slack(g_1)$	$Slack(g_2)$	$Slack(g_3)$	$Slack(g_4)$
4.1.5	SSA	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.808844189632734	9.649307648E+12	2.029983232E+11	4.363240654 84E+13	8.71498795E+11
	LSSA	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.808844189632734	9.649307648E+12	2.029983232E+11	4.363240654 84E+13	8.71498795E+11
	CSSA	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.808844189632734	9.649307648E+12	2.029983232E+11	4.363240654 84E+13	8.71498795E+11
	GSSA	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.808844189632734	9.649307648E+12	2.029983232E+11	4.363240654 84E+13	8.71498795E+11
	HSSATLBO	(2, 2, 2, 1, 1, 2, 3, 2, 1, 2)	0.808844189632734	9.649307648E+12	2.029983232E+11	4.363240654 84E+13	8.71498795E+11
4.1.6	SSA	(3, 4, 5, 4, 3, 2, 4, 6, 4, 2, 3, 4, 5, 4, 5)	0.9452180086299	8	0	–	–
	LSSA	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.9456133574581	8	0	–	–
	CSSA	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.9456133574581	8	0	–	–
	GSSA	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.9456133574581	8	0	–	–
	HSSATLBO	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0.945613357458 1371	8	0	–	–

get rank 2, and so on. Besides, two algorithms having same results share the average of ranks. The algorithm with the smaller rank indicates that it is better than the compared algorithms.

In view of the above two quality parameters, the statistical results achieved for HSSATLBO and all other existing algorithms (like ABC, NNA, TLBO, SSA, HHO, SMA and SCA) and the three variants of SSA (LSSA, CSSA, and GSSA) are computed and summarized in Tables 19 and 20 for the considered problems. In this table, the mean, SD 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 benchmark issues. Also, we can sort the ranking, as per their achievement, in the order:

i. HSSATLBO, TLBO, SMA, SSA, NNA, ABC, HHO and SCA.
ii. HSSATLBO, GSSA, LSSA, CSSA and SSA.

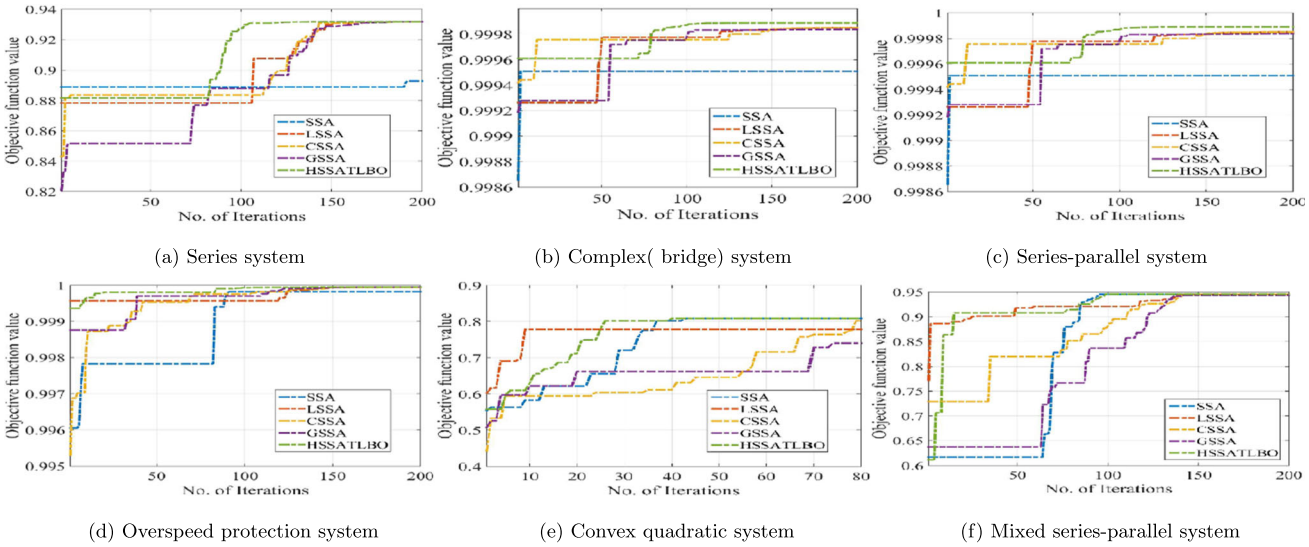
The ranking order in Table 19 indicates that the TLBO algorithm shows strong competitiveness and is the second-best on all test issues except Overspeed system. Also, in Table 20, the Gaussian variant of SSA (GSSA) occupied second best position on most of the cases except for solving the series-parallel system and the mixed system. It can therefore be argued that HSSATLBO is an efficient and effective method for solving various kinds of optimization problems.

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.

5.6.2 The results analysis by Wilcoxon signed-rank test

This statistical test-based method [17] is used to compare the performance of the proposed HSSATLBO 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 (Mafarja et al., 2018 [55]; Sun et al., 2018 [74]; Yi et al., 2019 [90]).

Wilcoxon signed-rank test is performed here with a significance level $\alpha = 0.05$ and the obtained results are shown in Tables 21 and 22 . In this table, "H" scored "1" if there is a symbolic difference between HSSATLBO 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 HSSATLBO is inferior to the compared algorithm. It is noted that the



(a) Series system (b) Complex(bridge) system (c) Series-parallel system

(d) Overspeed protection system (e) Convex quadratic system (f) Mixed series-parallel system

Fig. 5 Comparison of convergence curves of HSSATLBO with SSA variants

Table 17 Ranking of results with different values of parameter PSP

Problems		PSP1	PSP2	PSP3	PSP4	PSP5
4.1.1	Mean	0.930607050	0.931293414	0.930909699	**0.931549670**	0.930698573
	Rank	5	2	3	1	4
4.1.2	Mean	0.999889220	0.999889278	0.999889346	**0.999889391**	0.999889348
	Rank	5	4	3	1	2
4.1.3	Mean	0.999984725	0.999984748	0.999984869	**0.999985542**	0.999985240
	Rank	5	4	3	1	2
4.1.4	Mean	0.999949560	0.999949559	0.999952968	**0.999954106**	0.999953538
	Rank	4	5	3	1	2
4.1.5	Mean	**0.808844189633**	**0.808844189633**	**0.808844189633**	**0.808844189633**	**0.808844189633**
	Rank	1	1	1	1	1
4.1.6	Mean	0.944738729	0.944725261	0.945334852	**0.945356368**	0.945214695
	Rank	4	5	2	1	3
Average ranking		4	3.5	2.5	1	2.33
Ranking		5	4	3	1	2

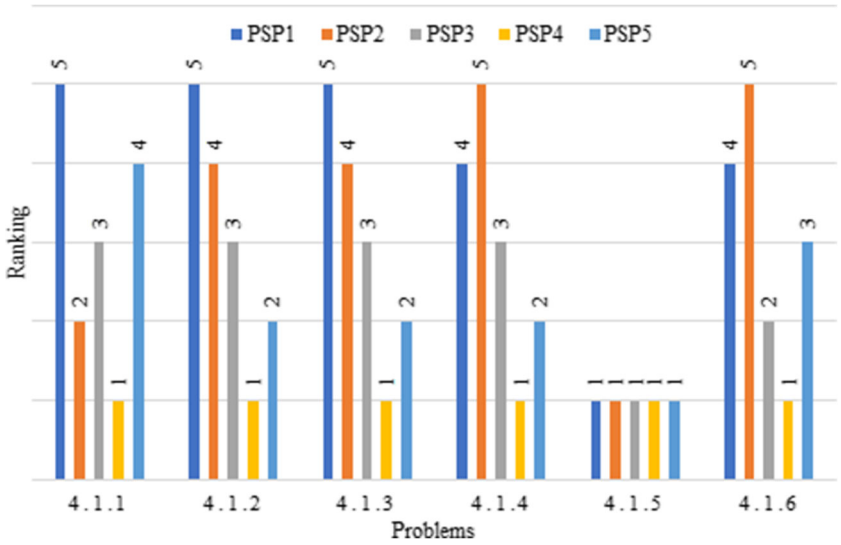


Fig. 6 Ranking of results with different values of parameter PSP

Table 18 Comparison on Diversity and Exploration-Exploitation measurement with SSA and its variants

Problems	Measurement	Compared algorithms				
		SSA	LSSA	CSSA	GSSA	HSSATLBO
4.1.1	Diversity	1.17742	0.64185	0.65301	0.65110	0.12618
	Expl% : Expt%	66 : 34	40 : 60	41 : 59	41 : 59	19 : 81
4.1.2	Diversity	1.19247	0.63969	0.65583	0.66395	0.08692
	Expl% : Expt%	66 : 33	47 : 53	41 : 59	42 : 58	12 : 88
4.1.3	Diversity	1.20481	0.70349	0.67796	0.66555	0.19484
	Expl% : Expt%	68 : 31	44 : 56	43 : 57	42 : 58	28 : 72
4.1.4	Diversity	1.02167	0.54170	0.55105	0.56604	0.07521
	Expl% : Expt%	63 : 37	38 : 63	38 : 62	39 : 60	15 : 85
4.1.5	Diversity	2.67162	1.39743	1.37574	1.40417	0.30378
	Expl% : Expt%	90 : 9	61 : 39	60 : 40	62 : 38	33 : 67
4.1.6	Diversity	3.54413	1.69871	1.67273	1.72212	0.43632
	Expl% : Expt%	81 : 18	45 : 51	45 : 55	46 : 54	35 : 65

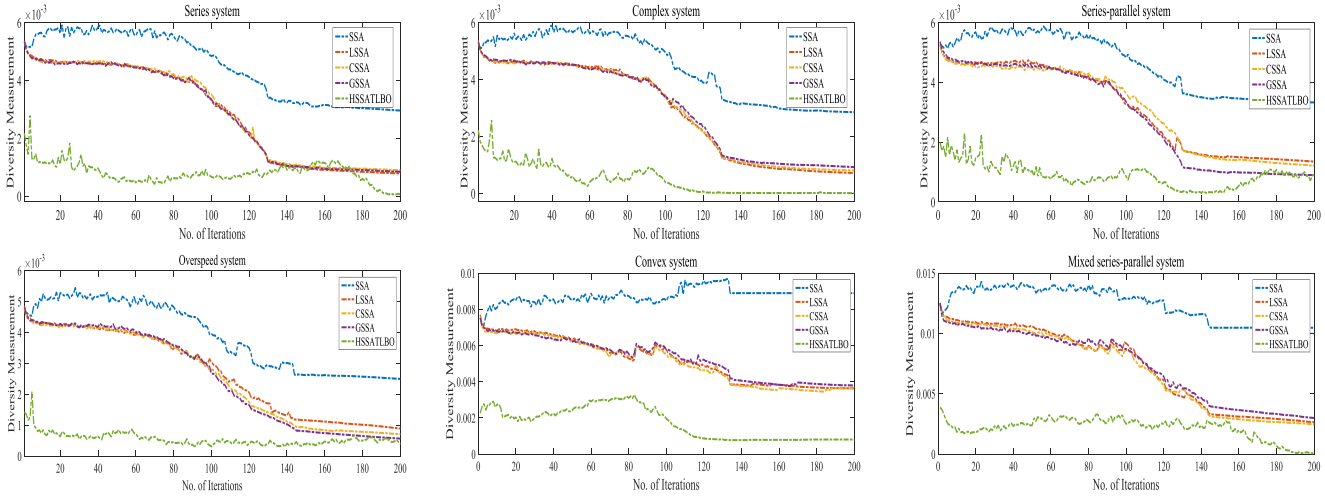


Fig. 7 Comparison of Diversity Measurement of HSSATLBO with SSA, LSSA, CSSA and GSSA

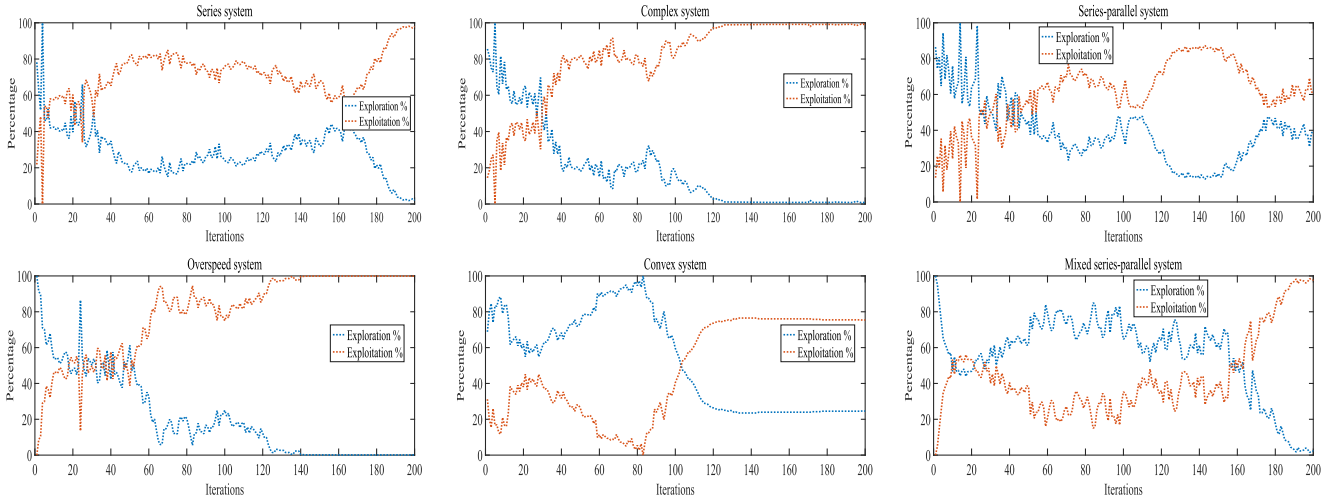


Fig. 8 Exploration-Exploitation measurement of HSSATLBO for solving 4.1.1 to 4.1.6

Table 19 Comparison of the statistical results obtained by HSSATLBO and the existing optimizers

Problems		HHO	SMA	SCA	ABC	NNA	TLBO	SSA	HSSATLBO
4.1.1	Best	0.9232316599627	0.9316562323884	0.9197764827240	0.930382490513	0.931682365783	0.931682337961	0.93168237854	0.931682387100
	Mean	0.8972295012520	0.9277272900921	0.8944273654145	0.924381473736	0.927436092852	0.929590566068	0.924530590320	**0.931379775783**
	Std	1.9334365E-02	3.3659236E-03	1.8268221E-02	4.66145E-03	3.136224E-03	3.10656E-03	5.741864E-03	8.026681E-04
	Median	0.9062971654865	0.9283863318909	0.8961627775126	0.925846286649	0.927193504074	0.931681035689	0.924646713520	0.931680940554
	Rank	7	3	8	6	4	2	5	1
4.1.2	Best	0.999857265455	0.999889192967	0.999798858923	0.999882439262	0.999889619976	0.999889572955	0.999889343515	0.999889637382
	Mean	0.999677703132	0.999835962457	0.999673706031	0.999848278911	0.999831224137	0.999863109311	0.999851215328	**0.999889356835**
	Std	1.55695E-04	7.11217E-05	9.38780E-05	1.85770E-05	6.82623E-05	3.49079E-05	2.14278E-05	1.54451E-07
	Median	0.999721962723	0.999854265310	0.999685279461	0.999849258203	0.999844812853	0.999885734109	0.999851315177	0.999889331930
	Rank	7	5	8	4	6	2	3	1
4.1.3	Best	0.999985518490	0.999986245353	0.999963269496	0.999984489590	0.999984756234	0.999986321134	0.9999863369493	0.999986337376
	Mean	0.999957711886	0.999976964757	0.999915641671	0.999975636531	0.999931755895	0.999981341020	0.999969247658	**0.999984950098**
	Std	3.15903E-05	1.21419E-05	3.32342E-05	7.35266E-06	1.25126E-04	2.52450E-06	1.75994E-05	2.28012E-06
	Median	0.999972115709	0.999979723190	0.999922167085	0.999977378889	0.999979456599	0.999980456985	0.999979814445	0.999986321573
	Rank	6	3	8	4	7	2	5	1
4.1.4	Best	0.999948123242	0.999954664575	0.999872741449	0.999953683180	0.999954674620	0.999954674556	0.999954674661	0.999954674664323
	Mean	0.999795203405	0.999942731867	0.999568789845	0.999943724103	0.999945629094	0.999934231510	0.999940788377	**0.999954104675**
	Std	2.15848E-04	2.19727E-05	2.34285E-04	6.52910E-06	1.05988E-05	7.36309E-05	1.60499E-05	2.16403E-06
	Median	0.999889663319	0.999954382137	0.999594706829	0.999944385656	0.999946151065	0.999946151194	0.999946134330	0.999954674328
	Rank	7	4	8	3	2	6	5	1
4.1.5	Best	0.808844189633	0.808844189633	0.808844189633	0.785722876111	0.808844189633	0.808844189633	0.808844189633	0.808844189633
	Mean	0.803652411107	0.800204709330	0.799655173430	0.647607656192	0.779653224368	0.804485730959	0.792425175286	**0.808844189633**
	Std	9.35732E-03	1.00297E-02	1.02335E-02	7.73926E-02	2.06347E-02	7.64016E-03	1.76271E-02	5.64601E-16
	Median	0.808844189633	0.808844189633	0.805407390333	0.640077832082	0.780835652533	0.808844189633	0.794475538781	0.808844189633
	Rank	3	4	5	8	7	2	6	1
4.1.6	Best	0.944748484568	0.945613357458	0.921041824697	0.834752754620	0.945613357458	0.945613357458	0.945218008630	0.945613357458
	Mean	0.940268430460	0.942641142180	0.889585424484	0.714460886500	0.942471745286	0.944324914872	0.940979789237	**0.945368142124**
	Std	2.12239E-03	2.37562E-03	1.97222E-02	6.32905E-02	2.32059E-03	1.25456E-03	5.91813E-03	3.76312E-04
	Median	0.940127901001	0.943252733491	0.893075644595	0.700895471149	0.943252733491	0.944748484568	0.942552429269	0.945613357458
	Rank	6	3	7	8	4	2	5	1
Average ranking		6	3.67	7.33	5.5	5	2.67	4.83	1
Ranking		7	3	8	6	5	2	4	1

Table 20 Comparison of the statistical results obtained by HSSATLBO and the SSA variants

Problems		SSA	LSSA	CSSA	GSSA	HSSATLBO
4.1.1	Best	0.931682378549	0.931681598466	0.931681812332	0.931681957580	0.931682387100
	Mean	0.924530590294	0.921589927882	0.921580640950	0.925015928556	**0.931379775784**
	Std	5.74186E-03	7.52792E-03	7.09996E-03	6.61257E-03	8.02668E-04
	Median	0.924646713521	0.921405053083	0.923357326677	0.928483055614	0.931680940554
	Rank	3	4	5	2	1
4.1.2	Best	0.999889343515	0.999889373288	0.999889636119	0.999889601704	0.999889637382
	Mean	0.999851215328	0.999836304693	0.999857891541	0.999858100930	**0.999889356835**
	Std	2.14278E-05	4.79475E-05	2.00489E-05	1.55662E-05	1.54451E-07
	Median	0.999851315177	0.999851302379	0.999864570775	0.999851368670	0.999889331930
	Rank	4	5	3	2	1
4.1.3	Best	0.999986336949	0.999986335794	0.999986336878	0.999986303326	0.999986337376
	Mean	0.999969247658	0.999976040265	0.999973838913	0.999971544907	**0.999984950098**
	Std	1.75994E-05	1.20043E-05	9.64372E-06	1.66063E-05	2.28012E-06
	Median	0.999979814445	0.999979814849	0.999979803291	0.999979776596	0.999986321573
	Rank	5	2	3	4	1
4.1.4	Best	0.999954674661	0.999954674643	0.999954674661	0.999954674540	0.999954674664
	Mean	0.999940503716	0.999941627064	0.999938408345	0.999941853887	**0.999954104695**
	Std	1.58699E-05	2.05199E-05	3.65211E-05	1.57548E-05	2.16403E-06
	Median	0.999946134330	0.999946124021	0.999946117229	0.999946114350	0.999954674352
	Rank	4	3	5	2	1
4.1.5	Best	0.808844189633	0.808844189633	0.808844189633	0.808844189633	0.808844189633
	Mean	0.792425175286	0.807698589866	0.807469469913	0.807698589866	**0.808844189633**
	Std	1.76271E-02	2.60543E-03	2.79644E-03	2.60543E-03	5.64601E-16
	Median	0.794475538781	0.808844189633	0.808844189633	0.808844189633	0.808844189633
	Rank	5	2.5	4	2.5	1
4.1.6	Best	0.945218008630	0.945613357458	0.945613357458	0.945613357458	0.945613357458
	Mean	0.940979789237	0.943723468748	0.942826393320	0.940895208312	**0.945368142124**
	Std	5.91813E-03	1.40407E-03	2.78831E-03	4.43258E-03	3.76312E-04
	Median	0.942552429269	0.943943950607	0.943404680133	0.941624946698	0.945613357458
	Rank	4	2	3	5	1
Average ranking		4.17	3.09	3.83	2.92	1
Ranking		5	3	4	2	1

proposed algorithm HSSATLBO dominates all compared algorithms on all reliability problems. Thus, from this analysis, we conclude that the proposed HSSATLBO 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.

5.6.3 Kruskal-Wallis and multiple comparison test

The MCT test is performed here to justify whether the proposed HSSATLBO algorithm is better than the other optimizers (e.g., SSA, NNA, TLBO, ABC, HHO, SMA and SCA) and the other variants of SSA (LSSA, CSSA, and GSSA). 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 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 HSSATLBO algorithm results are compared with the compared algorithms results. The optimized results between the pairs of the different algorithms are summarized in Tables 23 and 24. 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.

Table 21 The comparison results of the applied algorithms by Wilcoxon signed-rank test (a level of significance $\alpha = 0.05$)

Problems	HSSATLBO vs																			
	ABC			NNA			TLBO			SSA			HHO			SMA			SCA		
	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
4.1.1	2.35342E-06	1	+	4.44933E-05	1	+	3.68261E-02	1	+	9.31565E-06	1	+	1.73440E-06	1	+	3.8822e-06	1	+	1.73440E-06	1	+
4.1.2	1.73439E-06	1	+	4.86026E-05	1	+	2.59671E-05	1	+	1.92092E-06	1	+	1.73440E-06	1	+	1.92092E-06	1	+	1.73440E-06	1	+
4.1.3	3.18167E-06	1	+	4.28568E-06	1	+	4.86026E-05	1	+	1.23807E-05	1	+	3.18168E-06	1	+	8.46608E-06	1	+	1.73440E-06	1	+
4.1.4	3.18167E-06	1	+	4.19550E-04	1	+	1.49356E-05	1	+	2.61343E-04	1	+	1.73440E-06	1	+	1.73440E-06	1	+	1.12654E-05	1	+
4.1.5	1.73439E-06	1	+	8.1463E-06	1	+	6.3103E-01	0	+	2.4375E-05	1	+	3.667E-01	1	+	4.2859E-02	1	+	6.1035E-05	1	+
4.1.6	1.73439E-06	1	+	7.2533E-06	1	+	2.7931E-04	1	+	2.6017E-06	1	+	2.5631E-06	1	+	1.1181E-05	1	+	1.7333E-06	1	+

Table 22 The comparison results of the SSA variants by Wilcoxon signed-rank test (a level of significance $\alpha = 0.05$)

Problems	HSSATLBO vs									
	SSA			LSSA			CSSA			GSSA		
	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S
4.1.1	9.32E-06	1	+	5.22E-06	1	+	1.92E-06	1	+	4.73E-06	1	+
4.1.2	1.92E-06	1	+	1.73E-06	1	+	2.88E-06	1	+	3.18E-06	1	+
4.1.3	1.24E-05	1	+	8.47E-06	1	+	2.88E-06	1	+	1.49E-05	1	+
4.1.4	2.61E-04	1	+	1.89E-04	1	+	1.97E-05	1	+	4.07E-05	1	+
4.1.5	2.44E-05	1	+	4.04E-02	1	+	1.38E-01	0	+	4.04E-02	1	+
4.1.6	2.60E-06	1	+	6.91E-06	1	+	1.22E-05	1	+	1.18E-05	1	+

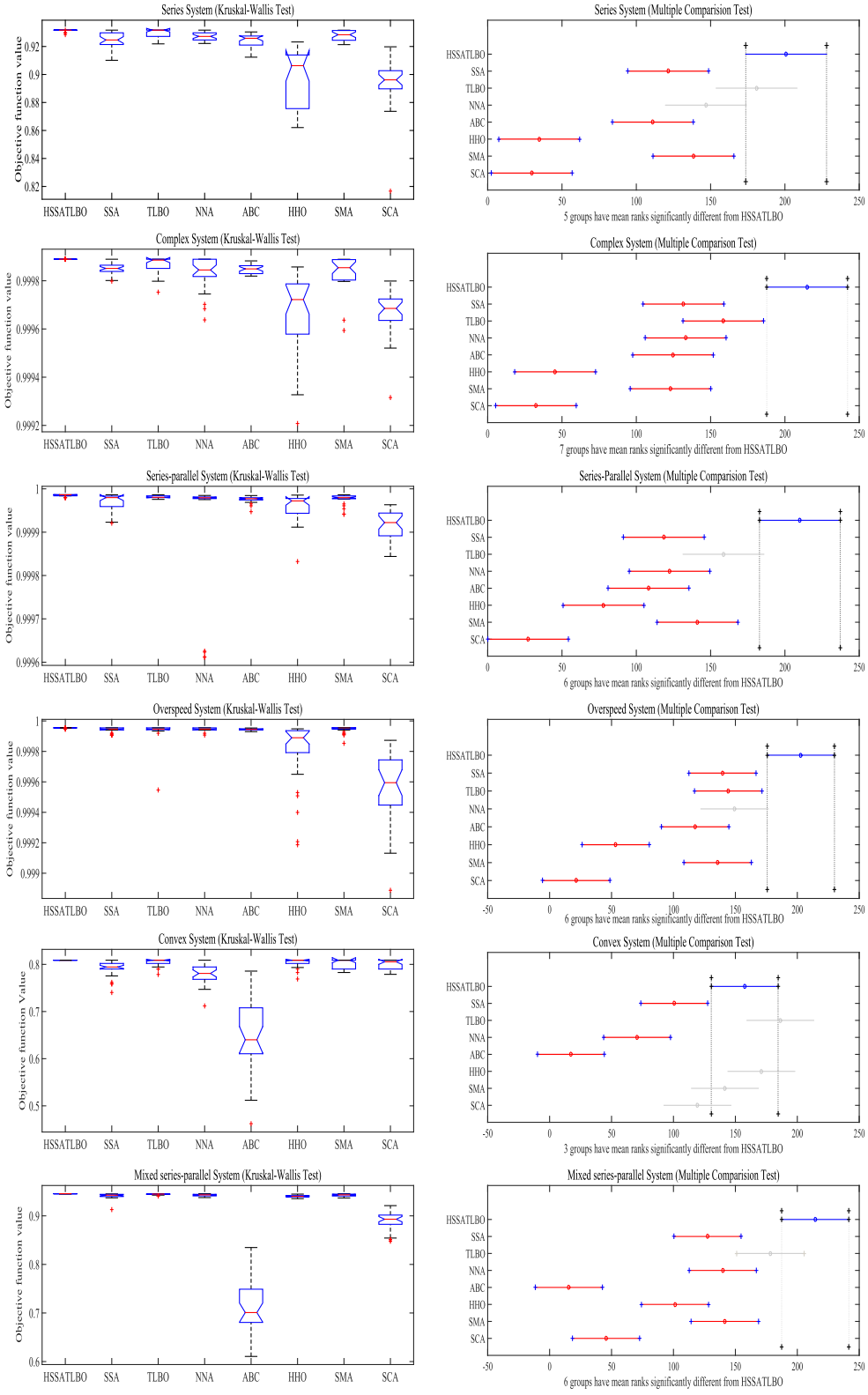


Fig. 9 Box plot of objective function using the reported optimizers

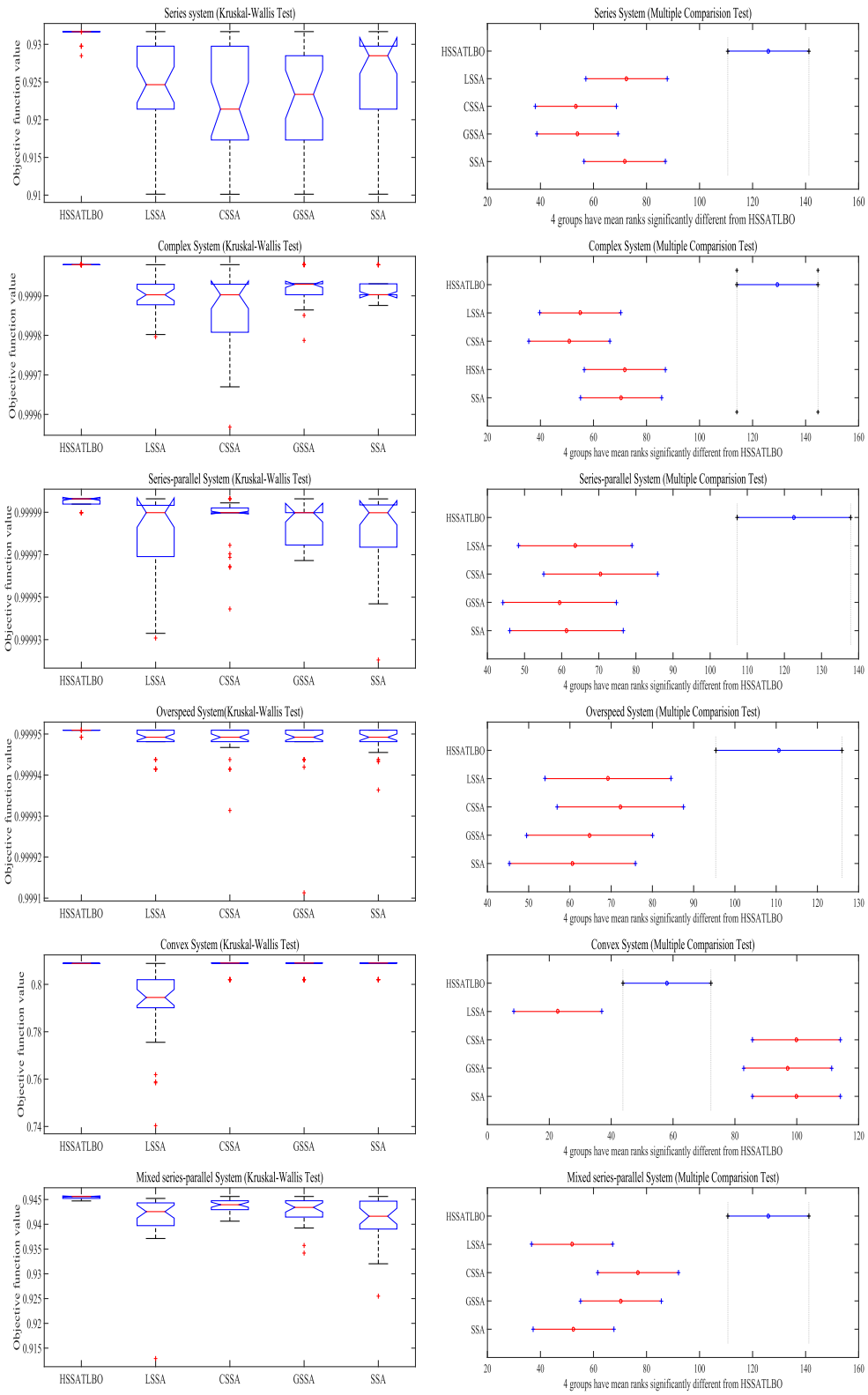


Fig. 10 Box plot of objective function using the SSA variants

Table 23 Statistical results of the existing optimizers using MCT analysis

Problems	Comparing	Lower bound	Group mean	Upper bound	p-value	Problems	Comparing	Lower bound	Group mean	Upper bound	p-value
4.1.1	HSSATLBO vs SSA	22.86833	53.46667	84.06500	1.85E-05	4.1.4	HSSATLBO vs SSA	10.83540	41.43333	72.03126	2.06E-03
	HSSATLBO vs TLBO	-34.414125	19.916667	74.247459	9.547E-01		HSSATLBO vs TLBO	4.287419	58.616667	112.945914	2.390E-02
	HSSATLBO vs NNA	-0.347459	53.983333	108.314125	5.291E-02		HSSATLBO vs NNA	-0.845914	53.483333	107.8125807	5.733E-02
	HSSATLBO vs ABC	35.402541	89.733333	144.064125	1.535E-05		HSSATLBO vs ABC	30.97075	85.30000	139.62925	5.323E-05
	HSSATLBO vs HHO	111.835875	166.166667	220.497459	5.988E-08		HSSATLBO vs HHO	95.17075	149.50000	203.82925	5.988E-08
	HSSATLBO vs SMA	8.002541	62.333333	116.664125	1.189E-02		HSSATLBO vs SMA	12.80409	67.13333	121.46258	4.468E-03
	HSSATLBO vs SCA	116.835875	171.166667	225.497459	5.988E-08		HSSATLBO vs SCA	127.07075	181.4	235.7292473	5.988E-08
4.1.2	HSSATLBO vs SSA	43.80368	74.40000	104.99632	1.02E-08	4.1.5	HSSATLBO vs SSA	6.83147	35.26667	63.70186	6.44E-03
	HSSATLBO vs TLBO	2.102494	56.433333	110.764172	3.514E-02		HSSATLBO vs TLBO	-82.732372	-28.800000	25.132372	7.393E-01
	HSSATLBO vs NNA	27.402494	81.733333	136.064172	1.386E-04		HSSATLBO vs NNA	32.967628	86.900000	140.832372	2.860E-05
	HSSATLBO vs ABC	35.935828	90.266667	144.597506	1.316E-05		HSSATLBO vs ABC	86.467628	140.400000	194.332372	5.988E-08
	HSSATLBO vs HHO	115.269161	169.600000	223.930839	5.988E-08		HSSATLBO vs HHO	-67.432372	-13.500000	40.432372	9.951E-01
	HSSATLBO vs SMA	37.602494	91.933333	146.264172	8.103E-06		HSSATLBO vs SMA	-37.999038	15.933333	69.865705	9.866E-01
	HSSATLBO vs SCA	128.169161	182.500000	236.830839	5.988E-08		HSSATLBO vs SCA	-15.799038	38.133333	92.065704	3.870E-01
4.1.3	HSSATLBO vs SSA	28.33467	58.93333	89.53200	1.49E-06	4.1.6	HSSATLBO vs SSA	43.47869	74.00000	104.52131	1.03E-08
	HSSATLBO vs TLBO	-2.947164	51.383333	105.713831	7.950E-02		HSSATLBO vs TLBO	-18.10360	36.20000	90.50360	4.678E-01
	HSSATLBO vs NNA	33.369503	87.700000	142.030497	2.736E-05		HSSATLBO vs NNA	20.32974	74.63333	128.93693	8.143E-04
	HSSATLBO vs ABC	47.486169	101.816667	156.147164	4.293E-07		HSSATLBO vs ABC	144.69640	199.00000	253.30360	5.988E-08
	HSSATLBO vs HHO	77.769503	132.100000	186.430497	5.988E-08		HSSATLBO vs HHO	58.89640	113.20000	167.50360	6.680E-08
	HSSATLBO vs SMA	14.536169	68.866667	123.197164	3.075E-03		HSSATLBO vs SMA	18.72974	73.03333	127.33693	1.188E-03
	HSSATLBO vs SCA	128.536169	182.866667	237.197164	5.988E-08		HSSATLBO vs SCA	114.62974	168.93333	223.23693	5.988E-08

The box-plot and the MCT graphs for the problems (4.1.1-4.1.6) considered are shown in Figs. 9 and 10. 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 HSSATLBO results and the red line indicates which algorithm results (such as HHO, SMA, SCA, ABC, NNA, TLBO, and SSA) or (LSSA, CSSA, GSSA and SSA) are statistically significant from the proposed HSSATLBO. For example, in case of series system (4.1.1), as shown in Fig. 9 we calculate that all existing algorithms (HHO, SMA, SCA, ABC, NNA, TLBO, and SSA) have statistically significant resources from the HSSATLBO algorithm. Furthermore, the vertical lines (right/left, shown in black colour) shown around the HSSATLBO 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 Figs. 9-10 and Tables 23-24, we conclude that the performance of the proposed algorithm is statistically significant with the other algorithms. The best results are therefore provided by the HSSATLBO.

6 Conclusions & Future work

In order to solve the reliability-redundancy allocation problems (RRAP) with non-linear resource constraints, this paper introduces a hybrid algorithm HSSATLBO combining the SSA and TLBO algorithms. SSA has been successfully tasted to solve various kinds of complex optimization problems due to its simple structure and outstanding performance. Although the SSA experience is adequate in exploration but lacking of exploitation, which forces slow convergence and reduces the optimizing accuracy. To address these issues, the basic formation of the SSA has been renovated by embedding the features of the TLBO. In this context, a probabilistic selection strategy is defined, which helps to determine whether to apply the basic SSA or the TLBO to construct a new solution. To demonstrate the application of the HSSATLBO algorithm, we have considered several benchmark issues in the areas of reliability optimization. All of these problems considered are mixed variables – discrete, continuous and integer. The core idea of the proposed HSSATLBO algorithm is to make full use of the good global search ability of SSA and fast convergence of TLBO that helps to maximize the system reliability through the choices of redundancy and component reliability. The results obtained from the

Table 24 Statistical results of SSA variants using MCT analysis

Problems	Comparing	Lower bound	Group mean	Upper bound	p-value	Problems	Comparing	Lower bound	Group mean	Upper bound	p-value
4.1.1	HSSATLBO vs SSA	22.86833	53.46667	84.06500	1.85E-05	4.1.4	HSSATLBO vs SSA	10.83540	41.43333	72.03126	2.06E-03
	HSSATLBO vs LSSA	41.93500	72.53333	103.13167	1.09E-08		HSSATLBO vs LSSA	7.835405	38.43333	69.03126	5.53E-03
	HSSATLBO vs CSSA	41.40166	72.00000	102.59834	1.12E-08		HSSATLBO vs CSSA	19.46873	50.06666	80.664594	7.90E-05
	HSSATLBO vs GSSA	23.56833	54.16667	84.76500	1.36E-05		HSSATLBO vs GSSA	15.30207	45.90000	76.49793	4.12E-04
4.1.2	HSSATLBO vs SSA	43.80368	74.40000	104.99632	1.02E-08	4.1.5	HSSATLBO vs SSA	6.83147	35.26667	63.70186	6.44E-03
	HSSATLBO vs LSSA	47.97034	78.56667	109.16299	9.94E-09		HSSATLBO vs LSSA	-70.26853	-41.83333	-13.39814	5.74E-04
	HSSATLBO vs CSSA	28.40368	59.00000	89.59632	1.44E-06		HSSATLBO vs CSSA	-67.53519	-39.10000	-10.66481	1.65E-03
	HSSATLBO vs GSSA	26.93701	57.53333	88.12966	2.89E-06		HSSATLBO vs GSSA	-70.26853	-41.83333	-13.39814	5.74E-04
4.1.3	HSSATLBO vs SSA	28.33467	58.93333	89.53200	1.49E-06	4.1.6	HSSATLBO vs SSA	43.47869	74.00000	104.52131	1.03E-08
	HSSATLBO vs LSSA	21.46800	52.06667	82.66533	3.40E-05		HSSATLBO vs LSSA	18.62869	49.15000	79.67131	0.000109304
	HSSATLBO vs CSSA	32.50134	63.10000	93.69866	1.94E-07		HSSATLBO vs CSSA	25.07869	55.60000	86.12131	6.67E-06
	HSSATLBO vs GSSA	30.63467	61.23333	91.83200	4.86E-07		HSSATLBO vs GSSA	42.97869	73.50000	104.02131	1.04E-08

proposed algorithm have been tested and compared with a number of existing algorithms and conclude that they also perform well. Also, the best, mean, median and SD of the problems considered are reports that indicate that the proposed algorithm has better results with less SD and is therefore reliable and optimal. In addition, in order to eliminate the stochastic nature of the algorithm, we perform several statistical tests, namely a ranking test and a Wilcoxon signed-rank test for each problem. All of the above discussions and evaluations in this study ensure that the proposed algorithm is a competitive approach, not only that it performs well but also that it has a better global performance optimization capability than the comparable algorithms to solve the reliability problems.

In future works, one can attempt to use the proposed algorithm in other applications such as airline recovery problems, integrated aircraft, and passenger recovery problems, flight perturbation problems, etc. Flight irregularity is a well-known and widespread problem all over the world which creates a serious impact on the performance of the airlines' company. All through the present decades, different models for dealing with the aircraft recovery issue have been recommended that hope to optimize the task upon different conditions. Airlines are attempting to locate the best timetables that are steady with their different objectives; namely, minimize the number of interrupted passengers and the total number of aircraft to recuperate from the interruption, decrease the number of irrecoverable flights, minimize the interrupted passenger's cost, thus the ultimate goal of airlines to maximize their overall profit (Andersson, 2006 [5]; Arıkan et al., 2016 [10]). Analysts suggest that it is better to consider recapture problems jointly with all essential constraints instead of considering only one recovery. Real situations including erratic interruptions require a more sensible arrangement that can retain changes in a better way. Based on our study in that field of disruption administration, our proposed algorithm may be considered for further research to solve the problem. Being a dynamic field for exploration, the researchers may extend the models to handle more complex variants of the combined recovery problem.

Acknowledgements The authors wish to acknowledge the Council of Scientific and Industrial Research (C.S.I.R), India, for financial support through the CSIR Grants-in-aid No: 25(0287)/18/ EMR-II.

References

1. Abasi AK, Khader AT, Al-Betar MA, Alyasseri ZAA, Makhadmeh SN, Al-laham M, Naim S (2021) A Hybrid Salp Swarm Algorithm with β-Hill Climbing Algorithm for Text Documents Clustering. Evolutionary Data Clustering: Algorithms and Applications, 129. https://doi.org/10.1016/j.jksuci.2021.06.015
2. Abirami M, Ganesan S, Subramanian S, Anandhakumar R (2014) Source and transmission line maintenance outage scheduling in a power system using teaching-learning based optimization algorithm. Applied Soft Computing Journal 21:72–83. https://doi.org/10.1016/j.asoc.2014.03.015
3. Abualigah L, Shehab M, Alshinwan M, Alabool H (2020) Salp swarm algorithm: a comprehensive survey. Neural Computing and Applications 32(15):11195–11215. https://doi.org/10.1007/s00521-019-04629-4
4. Al-Betar MA, Alyasseri ZAA, Awadallah MA, Doush IA (2021) Coronavirus herd immunity optimizer (CHIO). Neural Computing and Applications 33(10):5011–5042. https://doi.org/10.1007/s00521-020-05296-6
5. Andersson T (2006) Solving the flight perturbation problem with meta heuristics. Journal of Heuristics 12(1–2):37–53. https://doi.org/10.1007/s10732-006-4833-4
6. Afonso LD, Mariani VC, Dos Santos Coelho L (2013) Modified imperialist competitive algorithm based on attraction and repulsion concepts for reliability-redundancy optimization. Expert Systems with Applications 40(9):3794–3802. https://doi.org/10.1016/j.eswa.2012.12.093
7. Akay B, Karaboga D (2012) Artificial bee colony algorithm for large-scale problems and engineering design optimization. Journal of Intelligent Manufacturing 23(4):1001–1014. https://doi.org/10.1007/s10845-010-0393-4
8. Alkoffash MS, Awadallah MA, Alweshah M, Zitar RA, Assaleh K, Al-Betar MA (2021) A Non-convex Economic Load Dispatch Using Hybrid Salp Swarm Algorithm. Arabian Journal for Science and Engineering, 1-20. https://doi.org/10.1007/s13369-021-05646-z
9. Ardakan MA, Hamadani AZ (2014) Reliability-redundancy allocation problem with cold-standby redundancy strategy. Simulation Modelling Practice and Theory 42:107–118. https://doi.org/10.1016/j.simpat.2013.12.013
10. Arikan U, Gürel S, Aktürk MS (2016) Integrated aircraft and passenger recovery with cruise time controllability. Annals of Operations Research 236(2):295–317. https://doi.org/10.1007/s10479-013-1424-2
11. Bao X, Jia H, Lang C (2019) A novel hybrid harris hawks optimization for color image multilevel thresholding segmentation. IEEE Access 7:76529–76546. https://doi.org/10.1109/ACCESS.2019.2921545
12. Birashk A, Kazemi Kordestani J, Meybodi MR (2018) Cellular teaching-learning-based optimization approach for dynamic multi-objective problems. Knowledge-Based Systems 141:148–177. https://doi.org/10.1016/j.knosys.2017.11.016
13. Chen D, Zou F, Li Z, Wang J, Li S (2015) An improved teaching-learning-based optimization algorithm for solving global optimization problem. Information Sciences 297:171–190. https://doi.org/10.1016/j.ins.2014.11.001
14. Chen TC (2006) IAs based approach for reliability redundancy allocation problems. Applied Mathematics and Computation 182(2):1556–1567. https://doi.org/10.1016/j.amc.2006.05.044
15. Chen X, Mei C, Xu B, Yu K, Huang X (2018) Quadratic interpolation based teaching-learning-based optimization for chemical dynamic system optimization. Knowledge-Based Systems 145:250–263. https://doi.org/10.1016/j.knosys.2018.01.021
16. Chou JS, Truong DN (2021) A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. Applied Mathematics and Computation 389:125535. https://doi.org/10.1016/j.amc.2020.125535
17. Derrac J, García 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 and Evolutionary Computation 1(1):3–18. https://doi.org/10.1016/j.swevo.2011.02.002
18. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Computational Intelligence Magazine 1(4):28–39. https://doi.org/10.1109/MCI.2006.329691
19. Duan HB, Xu CF, Xing ZH (2010) A hybrid artificial bee colony optimization and quantum evolutionary algorithm for continuous optimization problems. International Journal of Neural Systems 20(1):39–50. https://doi.org/10.1142/S012906571000222X
20. Ekinci S, Hekimoğlu B, Kaya S (2018) Tuning of PID controller for AVR system using salp swarm algorithm. In: 2018 International Conference on Artificial Intelligence and Data Processing (IDAP) (pp. 1–6). IEEE. https://doi.org/10.1109/IDAP.2018.8620809
21. Ekinci S, Hekimoglu B (2018) Parameter optimization of power system stabilizer via salp swarm algorithm. In: 2018 5th international conference on electrical and electronic engineering (ICEEE) (pp. 143–147). IEEE. https://doi.org/10.1109/ICEEE2.2018.8391318
22. El-Fergany AA, Hasanien HM (2020) Salp swarm optimizer to solve optimal power flow comprising voltage stability analysis. Neural Computing and Applications 32(9):5267–5283. https://doi.org/10.1007/s00521-019-04029-8
23. Garg H (2015) An approach for solving constrained reliability-redundancy allocation problems using cuckoo search algorithm, vol 4. https://doi.org/10.1016/j.bjbas.2015.02.003
24. Garg H (2015) An efficient biogeography- based optimization algorithm for solving reliability optimization problems. Swarm and Evolutionary Computation 24:1–10. https://doi.org/10.1016/j.swevo.2015.05.001
25. Garg H, Rani M, Sharma SP (2013) An efficient two-phase approach for solving reliability-redundancy allocation problem using artificial bee colony technique. Computers and Operations Research 40(12):2961–2969. https://doi.org/10.1016/j.cor.2013.07.014
26. 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. https://doi.org/10.1007/3-540-48873-1-54
27. Gen M, Yun YS (2006) Soft computing approach for reliability optimization: State-of-the-art survey. Reliability Engineering and System Safety 91(9):1008–1026. https://doi.org/10.1016/j.ress.2005.11.053
28. Ghambari S, Rahati A (2018) An improved artificial bee colony algorithm and its application to reliability optimization problems. Applied Soft Computing Journal 62:736–767. https://doi.org/10.1016/j.asoc.2017.10.040
29. Goldberg DE, Holland JH (1988) Genetic Algorithms and Machine Learning. Machine Learning 3(2):95–99. https://doi.org/10.1023/A:1022602019183
30. Gupta S, Deep K, Heidari AA, Moayedi H, Chen H (2021) Harmonized salp chain-built optimization. Engineering with Computers, 1-31. https://doi.org/10.1007/s00366-019-00871-5
31. He Q, Hu X, Ren H, Zhang H (2015) A novel artificial fish swarm algorithm for solving large-scale reliability-redundancy application problem. ISA Transactions 59:105–113. https://doi.org/10.1016/j.isatra.2015.09.015
32. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Engineering Applications of Artificial Intelligence 20(1):89–99. https://doi.org/10.1016/j.engappai.2006.03.003
33. Hegazy AE, Makhlouf MA, El-Tawel GS (2020) Improved salp swarm algorithm for feature selection. Journal of King Saud University - Computer and Information Sciences 32(3):335–344. https://doi.org/10.1016/j.jksuci.2018.06.003
34. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: Algorithm and applications. Future generation computer systems 97:849–872. https://doi.org/10.1016/j.future.2019.02.028
35. Hsieh YC, You PS (2011) An effective immune based two-phase approach for the optimal reliability-redundancy allocation problem. Applied Mathematics and Computation 218(4):1297–1307. https://doi.org/10.1016/j.amc.2011.06.012
36. Huang CL (2015) A particle-based simplified swarm optimization algorithm for reliability redundancy allocation problems. Reliability Engineering and System Safety 142:221–230. https://doi.org/10.1016/j.ress.2015.06.002
37. Hussain K, Salleh MNM, Cheng S, Shi Y (2019) On the exploration and exploitation in popular swarm-based metaheuristic algorithms. Neural Computing and Applications 31(11):7665–7683. https://doi.org/10.1007/s00521-018-3592-0
38. Ibrahim RA, Ewees AA, Oliva D, Abd Elaziz M, Lu S (2019) Improved salp swarm algorithm based on particle swarm optimization for feature selection. Journal of Ambient Intelligence and Humanized Computing 10(8):3155–3169. https://doi.org/10.1007/s12652-018-1031-9
39. Ibrahim A, Ahmed A, Hussein S, Hassanien AE (2018) Fish image segmentation using salp swarm algorithm. In: International Conference on advanced machine learning technologies and applications (pp. 42–51). Springer, Cham. https://doi.org/10.1007/978-3-319-74690-6_5
40. Kanagaraj G, Ponnambalam SG, Jawahar N (2013) A hybrid cuckoo search and genetic algorithm for reliability-redundancy allocation problems. Computers and Industrial Engineering 66(4):1115–1124. https://doi.org/10.1016/j.cie.2013.08.003
41. Kassaymeh S, Abdullah S, Al-Betar MA, Alweshah M (2021) Salp swarm optimizer for modeling the software fault prediction problem. Journal of King Saud University-Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2021.01.015
42. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks 4:1942–1948. https://doi.org/10.1109/ICNN.1995.488968
43. Kim HG, Bae CO, Park DJ (2006) Reliability-redundancy optimization using simulated annealing algorithms. Journal of Quality in Maintenance Engineering 12(4):354–363. https://doi.org/10.1108/13552510610705928
44. Kundu T, Islam S (2018) Neutrosophic Goal Geometric Programming Problem and Its Application to Multi-objective Reliability Optimization Model. International Journal of Fuzzy Systems 20(6):1986–1994. https://doi.org/10.1007/s40815-018-0479-2
45. Kundu T, Islam S (2019) A new interactive approach to solve entropy based fuzzy reliability optimization model. International Journal on Interactive Design and Manufacturing 13(1):137–146. https://doi.org/10.1007/s12008-018-0484-6
46. Kundu T, Islam S (2019) An interactive weighted fuzzy goal programming technique to solve multi-objective reliability optimization problem. Journal of Industrial Engineering International 15:95–104. https://doi.org/10.1007/s40092-019-0321-y
47. Kuo W, Rajendra Prasad V (2000) An annotated overview of system-reliability optimization. IEEE Transactions on Reliability 49(2):176–187. https://doi.org/10.1109/24.877336
48. Kuo W, Wan R (2007) Recent advances in optimal reliability allocation. In: IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 37, no. 2, pp. 143–156. https://doi.org/10.1109/TSMCA.2006.889476
49. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: A new method for stochastic opti-

mization. Future Generation Computer Systems 111:300–323. https://doi.org/10.1016/j.future.2020.03.055
50. Liao TW (2010) Two hybrid differential evolution algorithms for engineering design optimization. Applied Soft Computing Journal 10(4):1188–1199. https://doi.org/10.1016/j.asoc.2010.05.007
51. Liu Y, Qin G (2014) A Modified Particle Swarm Optimization Algorithm for Reliability Redundancy Optimization Problem. Journal of Computers 9(9):2024–2031. https://doi.org/10.4304/jcp.9.9.2124-2131
52. Liu Y, Qin G (2014) A Hybrid TS-DE Algorithm for Reliability Redundancy Optimization Problem. Journal of Computers 9(9):2050–2057. https://doi.org/10.4304/jcp.9.9.2050-2057
53. Liu Y, Qin G (2015) A DE Algorithm Combined with Levy Flight for Reliability Redundancy Allocation Problems. International Journal of Hybrid Information Technology 8(5):113–118. https://doi.org/10.14257/ijhit.2015.8.5.12
54. Liu X, Xu H (2018) Application on target localization based on salp swarm algorithm. In: 37th Chinese Control Conference (CCC) (pp. 4542–4545). IEEE. https://doi.org/10.23919/ChiCC.2018.8482543
55. Mafarja M, Aljarah I, Heidari AA, Faris H, Fournier-Viger P, Li X, Mirjalili S (2018) Binary dragonfly optimization for feature selection using time-varying transfer functions. Knowledge-Based Systems 161:185–204. https://doi.org/10.1016/j.knosys.2018.08.003
56. Mellal MA, Zio E (2016) A penalty guided stochastic fractal search approach for system reliability optimization. Reliability Engineering and System Safety 152:213–227. https://doi.org/10.1016/j.ress.2016.03.019
57. Mirjalili S (2016) SCA: A Sine Cosine Algorithm for solving optimization problems. Knowledge-Based Systems 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022
58. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software 114:163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
59. Mirjalili S, Lewis A (2016) The Whale Optimization Algorithm. Advances in Engineering Software 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
60. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey Wolf Optimizer. Advances in Engineering Software 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
61. Nautiyal B, Prakash R, Vimal V, Liang G, Chen H (2021) Improved Salp Swarm Algorithm with mutation schemes for solving global optimization and engineering problems. Engineering with Computers, 1-23. https://doi.org/10.1007/s00366-020-01252-z
62. Ouyang HB, Gao LQ, Kong XY, Zou DX, Li S (2015) Teaching-learning based optimization with global crossover for global optimization problems. Applied Mathematics and Computation 265:533–556. https://doi.org/10.1016/j.amc.2015.05.012
63. Ouyang HB, Gao LQ, Li S, Kong XY (2015) Improved novel global harmony search with a new relaxation method for reliability optimization problems. Information Sciences 305:14–55. https://doi.org/10.1016/j.ins.2015.01.020
64. Rakhshani H, Rahati A (2017) Snap-drift cuckoo search: A novel cuckoo search optimization algorithm. Applied Soft Computing Journal 52:771–794. https://doi.org/10.1016/j.asoc.2016.09.048
65. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. CAD Computer Aided Design 43(3):303–315. https://doi.org/10.1016/j.cad.2010.12.015
66. Ravi V, Reddy PJ, Zimmermann HJ (2000) Fuzzy global optimization of complex system reliability. IEEE Transactions on Fuzzy Systems 8(3):241–248. https://doi.org/10.1109/91.855914
67. Sadollah A, Sayyaadi H, Yadav A (2018) A dynamic metaheuristic optimization model inspired by biological nervous systems: Neural network algorithm. Applied Soft Computing Journal 71:747–782. https://doi.org/10.1016/j.asoc.2018.07.039
68. Sahu PC, Mishra S, Prusty RC, Panda S (2018) Improved-salp swarm optimized type-II fuzzy controller in load frequency control of multi area islanded AC microgrid. Sustainable Energy, Grids and Networks 16:380–392. https://doi.org/10.1016/j.segan.2018.10.003
69. Savsani P, Savsani V (2016) Passing vehicle search (PVS): A novel metaheuristic algorithm. Applied Mathematical Modelling 40(5–6):3951–3978. https://doi.org/10.1016/j.apm.2015.10.040
70. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Applied Intelligence 48(10):3462–3481. https://doi.org/10.1007/s10489-018-1158-6
71. Simon D (2008) Biogeography-based optimization. IEEE Transactions on Evolutionary Computation 12(6):702–713. https://doi.org/10.1109/TEVC.2008.919004
72. Singh N, Son LH, Chiclana F, Magnot JP (2020) A new fusion of salp swarm with sine cosine for optimization of non-linear functions. Engineering with Computers 36(1):185–212. https://doi.org/10.1007/s00366-018-00696-8
73. Storn R, Price K (1997) Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11(4):341–359. https://doi.org/10.1023/A:1008202821328
74. Sun G, Ma P, Ren J, Zhang A, Jia X (2018) A stability constrained adaptive alpha for gravitational search algorithm. Knowledge-Based Systems 139:200–213. https://doi.org/10.1016/j.knosys.2017.10.018
75. Sun ZX, Hu R, Qian B, Liu B, Che GL (2018) Salp swarm algorithm based on blocks on critical path for reentrant job shop scheduling problems. In: International conference on intelligent computing (pp. 638–648). Springer, Cham. https://doi.org/10.1007/978-3-319-95930-6_64
76. Tillman FA, Hwang CL, Kuo W (1977) Optimization Techniques for System Reliability with Redundancy—A Review. IEEE Transactions on Reliability R-26(3):148–155. https://doi.org/10.1109/TR.1977.5220100
77. Tubishat M, Idris N, Shuib L, Abushariah MA, Mirjalili S (2020) Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection. Expert Systems with Applications 145:113122. https://doi.org/10.1016/j.eswa.2019.113122
78. Valian E, Tavakoli S, Mohanna S, Haghi A (2013) Improved cuckoo search for reliability optimization problems. Computers and Industrial Engineering 64(1):459–468. https://doi.org/10.1016/j.cie.2012.07.011
79. Valian E, Valian E (2013) A cuckoo search algorithm by Lévy flights for solving reliability redundancy allocation problems. Engineering Optimization 45(11):1273–1286. https://doi.org/10.1080/0305215X.2012.729055
80. Wang D, Zhou Y, Jiang S, Liu X (2018) A simplex method-based salp swarm algorithm for numerical and engineering optimization. IFIP Advances in Information and Communication Technology 538:150–159. https://doi.org/10.1007/978-3-030-00828-4_16
81. Wu P, Gao L, Zou D, Li S (2011) An improved particle swarm optimization algorithm for reliability problems. ISA Transactions 50(1):71–81. https://doi.org/10.1016/j.isatra.2010.08.005
82. Wu J, Nan R, Chen L (2019) Improved salp swarm algorithm based on weight factor and adaptive mutation. Journal of Experimental Theoretical Artificial Intelligence 31(3):493–515. https://doi.org/10.1080/0952813X.2019.1572659
83. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1):67–82. https://doi.org/10.1109/4235.585893

84. Xing Z, Jia H (2019) Multilevel color image segmentation based on GLCM and improved salp swarm algorithm. IEEE Access 7:37672–37690. https://doi.org/10.1109/ACCESS.2019.2904511
85. Yang XS (2010) A new metaheuristic Bat-inspired Algorithm. Studies in Computational Intelligence 284:65–74. https://doi.org/10.1007/978-3-642-12538-6_6
86. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: 2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings, pp 210–214. https://doi.org/10.1109/NABIC.2009.5393690
87. 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. Knowledge-Based Systems 159:51–62. https://doi.org/10.1016/j.knosys.2018.06.004
88. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation 3(2):82–102. https://doi.org/10.1109/4235.771163
89. Yeh WC, Hsieh TJ (2011) Solving reliability redundancy allocation problems using an artificial bee colony algorithm. Computers and Operations Research 38(11):1465–1473. https://doi.org/10.1016/j.cor.2010.10.028
90. Yi J, Gao L, Li X, Shoemaker CA, Lu C (2019) An online variable-fidelity surrogate-assisted harmony search algorithm with multi-level screening strategy for expensive engineering design optimization. Knowledge-Based Systems 170:1–19. https://doi.org/10.1016/j.knosys.2019.01.004
91. Yokota T, Gen M, Li YX (1996) Genetic algorithm for non-linear mixed integer programming problems and its applications. Computers and Industrial Engineering 30(4):905–917. https://doi.org/10.1016/0360-8352(96)00041-1
92. Zervoudakis K, Tsafarakis S (2020) A mayfly optimization algorithm. Computers Industrial Engineering 145:106559. https://doi.org/10.1016/j.cie.2020.106559
93. Zhang H, Hu X, Shao X, Li Z, Wang Y (2013) IPSO-based hybrid approaches for reliability-redundancy allocation problems. Science China Technological Sciences 56(11):2854–2864. https://doi.org/10.1007/s11431-013-5372-5
94. Zhang Y, Jin Z, Chen Y (2020) Hybridizing grey wolf optimization with neural network algorithm for global numerical optimization problems. Neural Computing and Applications 32(14):10451–10470. https://doi.org/10.1007/s00521-019-04580-4
95. Zhang J, Wang Z, Luo X (2018) Parameter estimation for soil water retention curve using the salp swarm algorithm. Water 10(6):815. https://doi.org/10.3390/w10060815
96. Zou D, Gao L, Li S, Wu J (2011) An effective global harmony search algorithm for reliability problems. Expert Systems with Applications 38(4):4642–4648. https://doi.org/10.1016/j.eswa.2010.09.120
97. Zou D, Gao L, Wu J, Li S, Li Y (2010) A novel global harmony search algorithm for reliability problems. Computers and Industrial Engineering 58(2):307–316. https://doi.org/10.1016/j.cie.2009.11.003
98. Zou D, Liu H, Gao L, Li S (2011) A novel modified differential evolution algorithm for constrained optimization problems. Computers and Mathematics with Applications 61(6):1608–1623. https://doi.org/10.1016/j.camwa.2011.01.029

Dr. Tanmay Kundu received his B.Sc, M.Sc. degrees in Mathematics from West Bengal State University and University of Kalyani in 2011 and 2013 respectively in India. After that, he received his Ph. D degree in Operation Research in 2019 from University of Kalyani. From March, 2019, he worked as a Research Associate at PDPM Indian Institute of Information Technology Design and Manufacturing, Jabalpur, M.P., India. His research interests includes applied computational intelligence, applied soft computing, reliability optimization, fuzzy and neutrosophic optimization.

Dr. Deepmala is working as an Assistant Professor in Mathematics Discipline of Indian Institute of Information Technology, Design & Manufacturing, Jabalpur, India. Before IIIT Jabalpur, she was Visiting Scientist at SQC & OR Unit of Indian Statistical Institute, Kolkata, India. Her research interests are in the areas of pure and applied mathematics including optimization techniques, applied soft computing, integral equations, game theory, airlines disruption management etc. She has published more than 40 research articles in esteemed journals and member of several scientific committees of reputed journals.

Dr. Pramod K. Jain is working as a Professor in the Department of Mechanical and Industrial Engineering at Indian Institute of Technology Roorkee, India. Currently, he is also the director of Indian Institute of Technology Varanasi (BHU), India. Trained as an Industrial Engineer, Prof. Jain received his B.E., M.E., and Ph.D. degree from the University of Roorkee (Now IIT Roorkee). He has a broad background in manufacturing systems and engineering including product design and development. He is also the recipient of several prestigious awards from various national and international bodies for recognition of his research and administrative work. He previously served as the Director of Indian Institute of Information Technology Design & Manufacturing, Jabalpur and Indian Institute of Technology, Patna. Prof. Jain is on the panel of several policy making committees of national importance, constituted by the Department of Science and Technology (DST), All India Council for Technical Education (AICTE), and Ministry of Education (MoE) in India. He has published several research papers and articles in various reputed national and international journals and conferences. He has also delivered several talks, keynote addresses at various national and international conferences. He has guided several M. Tech and PhD thesis and is also member of Editorial board and Reviewer of many international and national reputed journals. His research interests are in the areas of CAD/CAM, CAPP, Concurrent Engineering: Design for Manufacture, Design for Assembly, Tolerance Design, Manufacturing Systems, Modelling and Simulation, FMS, CMS, RMS, Agent Based Systems, Scheduling, Operations Management, Capacity Planning, Loading and Scheduling, Resource Planning, Machining Science, Conventional and Unconventional both (EDM, WEDM, ECM, ECH etc.).

Affiliations

Tanmay Kundu[1] · Deepmala[1] · Pramod K. Jain[2]

Tanmay Kundu
tanmay@iiitdmj.ac.in

Pramod K. Jain
pkjain123@gmail.com

1 PDPM IIITDM: PDPM Indian Institute of Information Technology Design and Manufacturing Jabalpur, Jabalpur, MP, India

2 Department of Mechanical and Industrial Engineering, Indian Institute of Technology Roorkee, Uttarakhand-247667, India