An Improved Reptile Search Algorithm Based on Lévy Flight and Interactive Crossover Strategy to Engineering Application

Liqiong Huan¹,², Yuanyuan Wang³, Yuxuan Guo² and Gang Hu²,*

¹ College of Mathematics and Computer Application, Shangluo University, Shangluo 726000, China; 229024@slxy.edu.cn
² Department of Applied Mathematics, Xi’an University of Technology, Xi’an 710054, China; guoyx@163.com
³ Electronic Information and Electrical Engineering College, Shangluo University, Shangluo 726000, China; 230030@slxy.edu.cn
* Correspondence: hugang@xaut.edu.cn

Abstract: In this paper, we propose a reptile search algorithm based on Lévy flight and interactive crossover strategy (LICRSA), and the improved algorithm is employed to improve the problems of poor convergence accuracy and slow iteration speed of the reptile search algorithm. First, the proposed algorithm increases the variety and flexibility of the people by introducing the Lévy flight strategy to prevent premature convergence and improve the robustness of the population. Secondly, an iteration-based interactive crossover strategy is proposed, inspired by the crossover operator and the difference operator. This strategy is applied to the reptile search algorithm (RSA), and the convergence accuracy of the algorithm is significantly improved. Finally, the improved algorithm is extensively tested using 2 test sets: 23 benchmark test functions and 10 CEC2020 functions, and 5 complex mechanical engineering optimization problems. The numerical results show that LICRSA outperforms RSA in 15 (65%) and 10 (100%) of the 2 test sets, respectively. In addition, LICRSA performs best in 10 (43%) and 4 (40%) among all algorithms. Meanwhile, the enhanced algorithm shows superiority and stability in handling engineering optimization.

Keywords: reptile search algorithm; Lévy flight; interactive crossover strategy; benchmark; engineering optimization

MSC: 49K35; 68T20

1. Introduction

Almost all scientific and engineering fields can readily translate into optimization problems [1–3], given the different real-world requirements of various application domains [4]. These optimization problems often include many needs and requirements, such as uncertainty, multi-objective, and high dimensions, whereas finding a solution strategy for optimization problems is usually a specification [5]. In contrast, various complex issues emerge as the different domains are deepened. Therefore, a more robust and faster solution is needed to solve them. This trend suggests that scientists need to find an algorithm adaptable to various applications to meet the complex situations of difficult problems in cutting-edge engineering fields [6,7]. A wide variety of optimization algorithms are found in delving into the literature on the proposed optimization algorithms [8,9]. These evolved from traditional optimization techniques using planning techniques to nature-inspired metaheuristic algorithms, each with its highlights [10,11]. As a general-purpose optimization method, traditional algorithms often fail to provide solutions that can solve the problem when faced with optimization problems of high complexity and high gradient variability in high dimensions [12,13]. In contrast, the MH benefits from multi-
group cooperation through shared experience to find the optimal solution accurately, bypassing the optimal local solution. It has the advantages of being easy to understand, converging quickly, and avoiding getting trapped in a local optimum [14–16].

In the latest decade, many MH algorithms have been proposed. In general, there is no standard/unique classification of metaheuristic algorithms. In this research, MH algorithms can be classified into four categories based on the source of inspiration: population intelligence (SI)-based, human-based algorithms, physics/chemistry-based algorithms, and evolutionary algorithms.

SI methods are usually inspired by the real-life and cooperative behaviors of different plants and animals in nature, and these methods search for the optimal solution adaptively by exchanging and sharing the information flow of multiple candidate solutions. Also, this approach is the most extended. Methods of this type include but are not limited to krill herd (KH) [17], GOA [18], moth flame optimization algorithm (MFO) [19], gray wolf optimizer (GWO) [20], red fox optimization algorithm (RFO) [21], slime mould algorithm (SMA) [22], aquila optimizer (AO) [23], Harris hawks optimization (HHO) [8], whale optimization algorithm (WOA) [24], sparrow search algorithm (SSA) [25], and butterfly optimization algorithm (BOA) [26].

Genetic laws often influence evolutionary algorithms, mimicking the crossover and variation behavior therein. Methods of this type include but are not limited to, evolutionary strategy (ES) [27], genetic algorithm (GA) [28].

Physics-based and chemistry-based methods mimic physical theorems or chemical phenomena prevalent in the universe and life, usually based on universal rules to distinguish the interactions between candidate solutions. There are various methods in this class, such as atomic search optimization (ASO) [29], thermal exchange optimization (TEO) [30], multi-verse optimization algorithm (MVO) [31], chemical reaction optimization (CRO) [32], and gravity search algorithms (GSA) [33].

The final method classified in this research is a human-based algorithm, mainly motivated by social and natural behavior and guided by autonomous human thought. This method contains human mental search (HMS) [34], student psychology based-optimization algorithm (SPBO) [35], and TLBO [36].

SI-based method, as a subset of the MH algorithm, has some more significant advantages: (1) More frequent information exchange. (2) The structure of the algorithm is more straightforward. (3) It is less prone to fall into local optimal solutions. Therefore, SI-based methods have been applied to deal with complex sequential optimization issues such as feature selection [37] and surface shape parameter optimization [12,38]. However, many methods in SI are not powerful enough and often cannot handle optimization situations with a wide range of scope areas, which may be due to the low level of development and exploration and poor diversity of some methods [39]. However, the balance between exploration and development for an SI method is an area that requires too much attention. With the increasing demands of optimization problems, many studies have proposed search strategies to ensure that balance [40–42]. Recently, Abualigah et al. established a proposed search algorithm—reptile search algorithm—which is capable of selecting both local and global searches to solve complex optimization problems [7]. The method mainly simulates the encirclement mechanism in the exploration stage and the hunting mechanism in the exploitation stage of the crocodile.

However, some experimental results also demonstrate that RSA also has the problem of poor convergence and slow convergence when facing high-dimensional complex non-linear optimization problems. Therefore, this paper uses some promising modification strategies to enhance the RSA algorithm capability. Firstly, Lévy flight has been considered an excellent strategy to help optimization algorithms improve their performance [43]. Zhang et al. applied the Lévy flight strategy to the backtracking search algorithm to solve the parameter estimation of the PV model [44]. OB-LF-ALO is an efficient algorithm optimized by Lévy flight and is employed to deal with some specific mechanical optimization in life [45]. Feng et al. proposed a Lévy flight-based gravitational search algorithm for
solving ecological scheduling of step hydro power plants [46]. Further, Gao et al. suggested an enhanced chicken flock algorithm based on Lévy flight for the multi-objective optimization problem of integrated energy for smart communities, considering the utility of decision-makers [47]. Therefore, the Lévy flight strategy is introduced into the reptile search algorithm in this paper to improve the global search capability of RSA and guarantee the search to escape the local optimum. Moreover, influenced by the crossover operator [48] and the difference operator [49], an interactive crossover strategy is proposed in this paper. In this strategy, the candidate solutions are more influenced by the iterative process, making the difference between the preliminary search and the later development more obvious. Therefore, an improved reptile search algorithm (LICRSA) based on Lévy flight and interactive crossover is proposed in this study based on these two strategies.

The remainder of this paper is structured as follows: Section 2 presents a comprehensive description of the reptile search algorithm. Section 3 depicts the contents of both Lévy flight and crossover strategies. In Section 4, the algorithm is evaluated using 23 benchmark functions and CEC2020 and compared with different MH methods. Section 5 selects five commonly used mechanical engineering optimization problems the proposed algorithm solves. Section 6 summarizes the work.

2. Description of the Reptile Search Algorithm (RSA)

The reptile search algorithm mainly imitates the predation strategy and social behavior of crocodiles in nature [7]. The first is the encirclement mechanism in the exploration phase, and the second is the hunting mechanism in the exploitation phase. A standard MH algorithm tends to be first applied to a set of candidate solutions before the iteration starts. It is a random strategy generated.

\[ Z = \begin{bmatrix}
    z_{1,1} & \cdots & z_{1,j} & \cdots & z_{1,d}
    \\
    \vdots & \ddots & \vdots & \ddots & \vdots
    \\
    z_{N,1} & \cdots & z_{N,j} & \cdots & z_{N,d}
\end{bmatrix} \]  

where \( z_{i,j} \) represents the \( j \)th dimension of the \( i \)th crocodile. \( N \) is all the crocodiles, and \( d \) denotes dimension. \( Z \) is \( N \) candidate solutions randomly generated by Equation (2).

\[ Z_{i,j} = rand \times (up - low) + low, \]  

where \( rand \) is a stochastic number; \( up \) and \( low \) are the maximum and minimum limits of the optimization problem.

Encircling behavior is the hallmark of the globalized search of RSA. The process is divided into two behaviors, aerial and abdominal walk. These two actions often do not allow crocodiles to get close to food. However, the crocodile will find the general area of the target food by chance after several search attempts because it is a global search of the whole solved spatial range. Meanwhile, ensure that the stage can be continuously adapted to the next developmental stage. The process is often limited to the first half of the total iteration. Equation (3) simulates the encircling behavior of the crocodile and is shown below.

\[ z_{i,j}^{(t+1)} = \begin{cases} 
    best_j(t) \times (-\eta_{i,j}(t)) \times \beta - R_{i,j}^{(t)} \times r, & t \leq \frac{T_{\text{Max}}}{4} \\
    best_j(t) \times z_{i,j} \times ES(t) \times r, & \frac{T_{\text{Max}}}{4} \leq t < \frac{2 \times T_{\text{Max}}}{4} 
\end{cases} \]
where $best_{i,j}(t)$ is the best-positioned crocodile at $t$ iterations, and $r$ is a stochastic number of 0 to 1. $T_{\text{Max}}$ represents the maximum iteration. $\eta_{i,j}$ is the operator of the $i$th crocodile at the $j$th dimension and is given by Equation (4). $\beta$ is a sensitive parameter controlling the search accuracy and is given as 0.1 in the original text [7]. $R_{i,j}$ is the reduce function, and the equation is used to decrease the explored area and is computed by Equation (5). $r_1$ is a stochastic integer between 1 and $N$, and $z_{r_1,j}$ is the $j$th dimension of the crocodile at position $r_1$. $ES(t)$ is an arbitrary decreasing probability ratio between 2 and $-2$ and is given in Equation (6).

$$\eta_{i,j} = best_{i,j}(t) \times P_{i,j},$$  

(4)

$$R_{i,j} = \frac{best_{j}(t) - x_{r_2,j}}{best_{j}(t) + \epsilon},$$  

(5)

$$ES(t) = 2 \times r_3 \times \left(1 - \frac{1}{T}\right),$$  

(6)

where $r_2$ is a random integer between 1 and $N$, and $r_3$ is an integer number of $-1$ to 1. $\epsilon$ is a small value. $P_{i,j}$ represents the percentage difference between the crocodile in the best position and crocodiles in the current position, updated as in Equation (7).

$$P_{i,j} = \alpha + \frac{z_{i,j} - M(z_i)}{best_{j}(t) \times (up_{j} - low_{j}) + \epsilon},$$  

(7)

where $M(z)$ is the average position of the crocodile of $z$ and is given in Equation (8), $up_j$ and $low_j$ are the maximum and minimum limits of the $j$th dimension. $\alpha$ is the same as $\beta$ as a sensitive parameter to control the search accuracy. It was set to 0.1 in the original paper [7].

$$M(z_i) = \frac{1}{n} \sum_{j=1}^{n} z_{i,j},$$  

(8)

Connected to the search process of RSA is the hunting process of local symbolic exploitation, which has two strategies in this part: coordination and cooperation. After the influence of the encirclement mechanism, the crocodiles almost lock the location of the target prey, and the hunting strategy of the crocodiles will make it easier for them to approach the target. The development phase will often find the near-optimal candidate solution after several iterations. The mathematical model of its simulated crocodile hunting behavior is presented in Equation (9). This development hunting strategy occurs in the second half of the iterations.

$$z_{i,j}(t+1) = \begin{cases} 
best_{j}(t) \times P_{i,j}(t) \times r, & \frac{2 \times T_{\text{Max}}}{4} \leq t < \frac{3 \times T_{\text{Max}}}{4} \\
best_{j}(t) - \eta_{i,j}(t) \times \epsilon - R_{i,j}(t) \times r, & \frac{3 \times T_{\text{Max}}}{4} \leq t < \frac{4 \times T_{\text{Max}}}{4}
\end{cases}$$  

(9)

where $best_i$ is the optimal position of the crocodile, $\eta_{i,j}$ is the operator of the $i$th crocodile at the $j$th dimension and is given by Equation (4). $R_{i,j}$ is the reduce function, and the equation is used to decrease the explored area and is computed by Equation (5). Algorithm 1 is the pseudo-code for RSA.
Algorithm 1: The framework of the RSA

1: **Input**: The parameters of RSA including the sensitive parameter $\alpha$, $\beta$, crocodile size $(N)$, and the maximum generation $T_{Max}$.
2: Initializing $n$ crocodile, $z_i$ and calculate $f_i$.
3: Determine the best crocodile $best_j$.
4: while ($t \leq T_{Max}$) do
5:     Update the ES by Equation (6).
6:     for $i = 1$ to $N$ do
7:         Calculate the $\eta$, $R$, $P$ by Equations (4), (5), (7).
8:         if $t \leq T_{Max}/4$ then
9:             $z_i(t + 1) = best_j(t) \times -\eta_i \times \beta - R_i(t) \times r$.
10:        else if $T_{Max}/4 \leq t < 2 \times T_{Max}/4$ then
11:            $z_i(t + 1) = best_j(t) \times z_i(t) \times ES(t) \times r$.
12:        else if $2 \times T_{Max}/4 \leq t < 3 \times T_{Max}/4$ then
13:            $z_i(t + 1) = best_j(t) \times P_i(t) \times r$.
14:        else
15:            $z_i(t + 1) = best_j(t) - \eta_i \times \epsilon - R_i(t) \times r$.
16:     end if
17:     end for
18: end for
19: Find the best crocodile.
20: $t = t + 1$.
21: end while
22: **Output**: The best crocodile.

3. The Proposed Hybrid RSA (LICRSA)

In this paper, the Lévy flight strategy and the interaction crossover strategy are chosen, respectively. The Lévy flight strategy helps candidate solutions leap out of local solutions and enhances the algorithm’s precision. An interaction crossover strategy mainly enhances the ability of algorithm development. The two methods are introduced primarily in the following.

3.1. Lévy Flight Strategy

The Lévy flight strategy is used in this paper to generate a random number with a replacement rand with the following characteristics: (1) The random numbers generated are occasionally large but mostly interspersed with small numbers in between. (2) The probability density function of the steps is heavy-tailed. This random number will help update the position to produce oscillations, perform a small search in the neighborhood by the fluctuations of the random number at one iteration, and help the candidate solution leap out of the local optimum.

The definition of Lévy distribution is given by Equation (10) below.

$$Levy(\gamma) \sim u = t^{1-\gamma}, \quad 0 < \gamma \leq 2,$$

The step length of the Lévy flight can be calculated as in Equation (11).

$$S = \frac{U}{|V|^{1/\gamma}},$$

where, both $U$ and $V$ follow Gaussian distribution as shown in Equation (12).

$$U \sim N(0, \sigma_U^2), \quad V \sim N(0, \sigma_V^2)$$

where $\sigma_U$ and $\sigma_V$ meet the following Equations (13) and (14).
\[
\sigma_U = \left( \frac{\Gamma(1+\beta) \cdot \sin(\pi \cdot \beta/2)}{\Gamma((1+\beta/2) \cdot \beta \cdot 2^{(\beta-1)/2})} \right)^{1/\beta},
\]
\[
\sigma_V = 1,
\]
(13)
(14)

where \( \Gamma \) is a standard Gamma function and \( \beta \) is a parameter with a fixed 1.5.

Lévy flight strategy is deployed for the crocodile’s high and belly walking in the encircling mechanism to expand the surrounding search area effectively. Meanwhile, the Lévy flight strategy is also deployed for hunting coordination and cooperation in the hunting behavior phase to increase the flexibility of the optimal exploitation phase. The encircling mechanism based on Lévy flight is displayed in Equation (15). \( \lambda \) is a parameter with a fixed 0.1.

\[
z_{i,j}(t+1) = \begin{cases} 
    \text{best}_j(t) \times (-\eta_{i,j}(t)) \times \beta - R_{i,j}(t) \times \lambda \times \text{Levy}(\gamma), & t \leq \frac{T_{\text{Max}}}{4} \\
    \text{best}_j(t) \times z_{i,j} \times \text{ES}(t) \times \lambda \times \text{Levy}(\gamma), & \frac{T_{\text{Max}}}{4} \leq t < \frac{2 \times T_{\text{Max}}}{4} 
\end{cases}
\]
\[
\text{The hunting behavior (hunting coordination and cooperation) based on Lévy flight is displayed in Equation (16).}
\]
\[
z_{i,j}(t+1) = \begin{cases} 
    \text{best}_j(t) \times P_{i,j}(t) \times \lambda \times \text{Levy}(\gamma), & \frac{2 \times T_{\text{Max}}}{4} \leq t < \frac{3 \times T_{\text{Max}}}{4} \\
    \text{best}_j(t) - \eta_{i,j}(t) \times \varepsilon - R_{i,j}(t) \times \lambda \times \text{Levy}(\gamma), & \frac{3 \times T_{\text{Max}}}{4} \leq t < \frac{4 \times T_{\text{Max}}}{4} 
\end{cases}
\]
\[
(15)
(16)

3.2. Interaction Crossover Strategy

The interaction crossover strategy helps the candidate in the current position readjust by exchanging the information of the two candidate solutions and the candidate at the optimal position. The new position draws the information on the optimal solution and other candidate solutions to improve the searchability of the candidate solutions. First, a parameter controlling the activity of the crocodile population is defined, which has a linearly decreasing trend with the number of iterations. \( CF \) is defined as follows:

\[
CF = \left(1 - \frac{t}{T_{\text{Max}}} \right)^{2 \times \frac{t}{T_{\text{Max}}}},
\]
\[
(17)
\]

where \( t \) represents the current iteration, \( T_{\text{Max}} \) stands for all iterations. The crocodile population is randomly divided into two parts with the same number of crocodiles. Two positions of crocodiles \( z_{21} \) and \( z_{22} \) are selected from these two parts, and then the positions are interacted with to update the two crocodiles. The updated strategy is the same as that provided by Equations (18) and (19).

\[
z_{k1,j}(t+1) = z_{k1,j}(t) + CF \times (\text{best}_j(t) - z_{k1,j}(t)) + c_1 \times (z_{k1,j}(t) - z_{k2,j}(t)),
\]
\[
z_{k2,j}(t+1) = z_{k2,j}(t) + CF \times (\text{best}_j(t) - z_{k2,j}(t)) + c_2 \times (z_{k2,j}(t) - z_{k1,j}(t)),
\]
\[
(18)
(19)

where \( \text{best}_j \) is the best positioned crocodile, \( c_1 \) and \( c_2 \) are stochastic numbers in the interval 0 to 1, and \( z_{21} \) is the crocodile in position \( k_1 \). Figure 1 illustrates the introduction graph of the interaction crossover.
For positions of the updated crocodiles, this paper set up an elimination mechanism: retaining the crocodiles with better positions after the interactive crossover and eliminating the crocodiles with poorer abilities. The updates are as follows:

\[
z_{i,j}(t+1) = \begin{cases} 
z_{i,j}(t), & \text{if } f(z_{i,j}(t)) < f(z_{i,j}(t+1)) \\
z_{i,j}(t+1), & \text{elseif } f(z_{i,j}(t+1)) < f(z_{i,j}(t)) \
\end{cases}
\]  

(20)

Figure 1. The introduction diagram of the interaction crossover.

3.3. Specific Implementation Steps of LICRSA

The Lévy flight strategy and the interactive crossover strategy are incorporated in RSA. These strategies effectively improve the accuracy and exploitation of the RSA algorithm while obtaining a better traversal of the cluster. The specific operation steps of LICRSA are displayed below.

**Step1**: Provide all the relevant parameters of LICRSA, such as the number of alligators N, the dimension of variables Dim, the upper limit up of all variables, the lower limit low of all variables, all iterations TMax, and the sensitive parameter β.

**Step2**: Initialize N random populations and calculate the corresponding adaptation values. The best crocodile individuals are selected.

**Step3**: Update ES values by Equation (6) and calculate η, R, and P values, respectively.

**Step4**: While \( t < T_{\text{Max}}/4 \), the search phase of RSA: If \( t < T_{\text{Max}}/4 \), update the new situation of the crocodile by the first part of Equation (15), and similarly, if \( t \) is between \( T_{\text{Max}}/4 \) and \( 2T_{\text{Max}}/4 \), then update the crocodile position according to the second part of Equation (15).

**Step5**: The development phase of RSA: if \( t \) is between \( 2T_{\text{Max}}/4 \) and \( 3T_{\text{Max}}/4 \), the position of the crocodile is updated by the first part of Equation (16), and in the final stage of the iteration, when \( t > 3T_{\text{Max}}/4 \), the position of the crocodile is updated by the second part of Equation (16).

**Step6**: The fitness of the crocodile positions is re-evaluated, and the crocodiles with function values worse than the last iteration are eliminated, while the position of the optimal crocodile is updated.

**Step7**: Interactive crossover strategy: firstly, the crocodile population is divided into two parts, and then crocodiles are randomly selected from the two groups in turn for crossover, and their information is interacted by the positions of the two crocodiles updated by Equation (18) and Equation (19), respectively.
Step 8: According to the principle of Equation (20), crocodile individuals with poor predation ability are eliminated, and the optimal ones are renewed.

Step 9: Determine if the iteration limit is exceeded and output the optimal value.

To show the hybrid RSA algorithm more clearly, in this paper, the pseudo-code of LICRSA is provided in Algorithm 2. Where line 10, line 12, line 14, and line 16 are the encirclement and predation phases of the Lévy flight strategy improvement. Lines 20–26 are the strategies for the interaction crossover. Figure 2 shows the flow chart of the LICRSA algorithm.

---

**Algorithm 2**: The framework of the LICRSA

1: **Input**: The parameters of LICRSA including the sensitive parameter \( \alpha, \beta \), crocodile size \( N \), the maximum generation \( T_{\text{Max}} \), and variable range \( u_b, l_b \).

2: Initializing \( n \) crocodile, \( z_i (i = 1, 2, \ldots, N) \) and calculate the fitness \( f_i \).

3: Determine the best crocodile \( \text{best}_j \).

4: **while** \( t \leq t_{\text{Max}} \) **do**

5: Update the ES by Equation (6).

6:  **for** \( i = 1 \) to \( N \) **do**

7:    **for** \( i = 1 \) to \( N \) **do**

8:      Calculate the \( \eta, R, P \) by Equations (4), (5), (7).

9:      **if** \( t \leq T_{\text{Max}}/4 \) **then**

10:     \( z_i (t + 1) = \text{best}_i (t) - \eta_i \times \beta - R_i (t) \times \lambda \times \text{levy}(\gamma) \).

11:     **else if** \( T_{\text{Max}}/4 \leq t < 2 \times T_{\text{Max}}/4 \) **then**

12:     \( z_i (t + 1) = \text{best}_i (t) \times \frac{z_i (t)}{\text{ES}(t)} \times \lambda \times \text{levy}(\gamma) \).

13:     **else if** \( 2 \times T_{\text{Max}}/4 \leq t < 3 \times T_{\text{Max}}/4 \) **then**

14:     \( z_i (t + 1) = \text{best}_i (t) \times P_i (t) \times \lambda \times \text{levy}(\gamma) \).

15:     **else**

16:     \( z_i (t + 1) = \text{best}_i (t) - \eta_i \times \varepsilon - R_i (t) \times \lambda \times \text{levy}(\gamma) \).

17:     **end if**

18:  **end for**

19:  **end for**

20: Update the fitness and location of the crocodile.
21: Replace the optimal location and optimal fitness.
22: $\text{Group} = \text{permute}(n)$.
23: for $i = 1$ to $N/2$ do
24: let $k_1 = \text{Group}(2 \times i - 1)$ and $k_2 = \text{Group}(2 \times i)$.
25: Update the $\text{CF}$ by Equation (17).
26: $z_{k_1}(t + 1) = z_{k_1}(t) + \text{CF} \times (\text{best}_j(t) - z_{k_1}(t)) + c_1 \times (z_{k_1}(t) - z_{k_2}(t))$
27: $z_{k_2}(t + 1) = z_{k_2}(t) + \text{CF} \times (\text{best}_j(t) - z_{k_2}(t)) + c_1 \times (z_{k_2}(t) - z_{k_1}(t))$
28: end for
29: Find the best crocodile.
30: $t = t + 1$.
31: end while
32: Output: The best crocodile.

3.4. Considering the Time Complexity of LICRSA

The estimation of LICRSA time complexity is mainly based on RSA with the addition of the interaction crossover part, and the Lévy strategy only improves the update strategy of RSA. The time complexity of RSA is:

$$O(T_{\text{Max}} \times d \times N) + O(T_{\text{Max}} \times N)$$

(21)

The adaptation value evaluation mainly depends on the complexity of the problem, so it is not considered in this paper. In the crossover phase, the method is a two-part population operation process, so the time complexity is:

$$O(T_{\text{Max}} \times d \times N) + O(T_{\text{Max}} \times N) + O(T_{\text{Max}} \times d \times N/2)$$

(22)

where $d$ is the variables, $N$ is all the crocodiles, and $T_{\text{Max}}$ is the maximum generation.

4. Simulation Experiments

To demonstrate the effectiveness of this improvement plan, 23 standard benchmark functions and CEC2020 will be used to test the local exploitation capability of LICRSA, global search capability, and ability to step out of local optimization. The experimental parameters of this section are set: the crocodile size $N = 30$ and the maximum iterations $T_{\text{Max}} = 1000$. In addition, for each test function, all methods will be run 30 times independently in the same dimension, and the numerical results will be counted. The comparative MH algorithms in this section include reptile search algorithm (RSA) [7], aquila optimizer (AO) [23], arithmetic optimization algorithm (AOA) [4], differential evolution (DE) [49], crisscross optimization algorithm (CSO) [48], dwarf mongoose optimization (DMO) [6], white shark optimizer (WSO) [50], whale optimization algorithm (WOA) [24], wild horse optimizer (WHOA) [15], seagull optimization algorithm (SOA) [51], and gravitational search algorithm (GSA) [33] in 23 test functions. Additionally, reptile search algorithm (RSA) [7], aquila optimizer (AO) [23], arithmetic optimization algorithm (AOA) [4], differential evolution (DE) [49], dwarf mongoose optimization (DMO) [6], grasshopper optimization algorithm (GOA) [18], gravitational search algorithm (GSA) [33], honey badger algorithm (HBA) [52], HHO [8], SSA [11], and WOA [24] in CEC2020. It is worth noting that to guarantee the equity of the experiments, the parameter settings of all MH methods are the same as those provided in Table 1 and are consistent with the source paper.
Table 1. Parameters setting.

| Algorithm | Parameters | Values |
|-----------|------------|--------|
| AOA [4]   | \( \mu \)  | 0.499  |
|           | \( a \)    | \( a \) |
| WSO [50]  | \( f_{\text{min}}, f_{\text{max}} \) | 0.07, 0.75 |
|           | \( \tau, a_0, a_1, a_2 \) | 4.125, 6.25, 100, 0.0005 |
| GSA [33]  | ElitistCheck, Rpower, Rnorm, alpha, \( G_0 \) | 1, 1, 2, 20, 100 |
| WOA [24]  | A          | Decreased from 2 to 0 |
|           | b          | 2      |
| SSA [11]  | Leader position update probability | \( c_3 = 0.5 \) |
| RSA [7]   | \( a \)    | 0.1    |
|           | \( \beta \) | 0.1    |
| DE [49]   | Scaling factor | 0.5 |
|           | Crossover probability | 0.5 |
| WHO [15]  | Crossover percentage | \( \text{PC} = 0.13 \) |
|           | Stallions percentage (number of groups) | \( \text{PS} = 0.2 \) |
|           | Crossover Mean | |
| HBA [52]  | \( \beta \) (the ability of a honey badger to get food) | 6 |
|           | \( C \)     | 2      |
| SOA [51]  | Control Parameter (A) | \( [2, 0] \) |
|           | \( fc \)    | 2      |

4.1. Exploration–Exploitation Analysis

The two main parts of different MH algorithms are the exploration and exploitation phases. The role of exploration is that distant regions of the search domain can be explored, ensuring better search candidate solutions. Alternatively, exploitation is the convergence to the region where the optimal solution is promising and expected to be found through a local convergence strategy. Maintaining a balance between these two conflicting phases is critical for the optimization algorithm to locate the optimal candidate solution to the optimization problem [53]. Dimensional diversity in the search procedure can be expressed in Equations (23) and (24).

\[
\text{Div}_j = \frac{1}{N} \sum_{i=1}^{N} \text{median}(z_{i,j}) - z_{i,j} \quad (23)
\]

\[
\text{Div}(t) = \frac{1}{d} \sum_{j=1}^{d} \text{Div}_j, t = 1, 2, \ldots T_{\text{Max}} \quad (24)
\]

where \( z_{i,j} \) is the position of the \( i \)th crocodile in the \( j \)th dim, \( \text{Div}_j \) is the average diversity of dimension \( j \), and \( \text{median}(z) \) is the median value of the \( j \)th dim of the candidate solution. \( N \) is all the crocodiles, \( d \) is the dim, and \( T_{\text{Max}} \) is the maximum iteration. The following equation calculates the percentage of exploration and exploitation.

\[
\text{Exploration}\% = \frac{\text{Div}(t)}{\text{max(Div)}} \times 100\%, \quad (25)
\]

\[
\text{Exploitation}\% = \frac{[\text{Div}(t) - \text{max(Div)}]}{\text{max(Div)}} \times 100\%, \quad (26)
\]

where \( \text{max(Div)} \) is the maximum diversity of the iterative process.
The test set of CEC2020 is selected in this section, and Figure 3 demonstrates the exploration and exploitation behavior of LICRSA in the CEC2020 test set [5] during the search process. Analyzing Figure 3, LICRSA mainly starts from high exploration and low exploitation. Still, as the exploration proceeds, LICRSA quickly switches to high exploitation and converges to the region where the optimal solution is expected to be found. There is a partial addition of the exploration process during the iteration process, which is energetic in both exploration and utilization. Therefore, LICRSA can find better optimal solutions than other MH algorithms. In addition, Figure 4 shows the percentage of different functions, the rate of exploration is generally between 10% and 30%, which indicates that LICRSA provides enough exploration time. The rest of the effort is consumed in development. This result means that LICRSA maintains a sufficiently high exploration percentage. On the other hand, LICRSA development utilization is high.

Figure 3. The exploration and exploitation behavior: (a) cec02; (b) cec04; (c) cec06; (d) cec07; (e) cec08; (f) cec09.

Figure 4. The percentage of different functions.
4.2. Performance Evaluation for the 23 Classical Functions

The 23 benchmark functions include 7 unimodal, 6 multimodal, and 10 fixed-dimensional functions [5]. Among them, unimodal functions are used to explore MH algorithms, and multimodal functions are often used to understand the development of algorithms. The dimension function aims to know how well different exploration algorithms leap out of the local optimum and how accurately algorithms converge. It should be noted that the objective of all 23 functions is to minimize the problem. In addition, four criteria are used in this part of the experiment, including maximum, minimum, mean, and standard deviation (Std). In addition, the Wilcoxon signed-rank test (significance level = 0.05) is used to test for the numerical cases. Where “+” means that an algorithm is more accurate than LICRSA, “=” indicates that there is no significant difference between the two, and “−” indicates that an algorithm is less accurate than LICRSA. We also provide the p-values as reference data.

Table 2 provides the numerical case of 12 MH algorithms in solving 23 benchmark problems, where the bold indicates the most available mean and standard deviation of all algorithms. From the table, LICRSA ranked first with an average of 3.8696, and LICRSA ranked first in 10 of these test problems and second or higher in 13. The LICRSA algorithm performs the best in single-peak, ranking second or higher in 6 out of 7 unimodal functions. Meanwhile, LICRSA also performed well in multimodal functions, obtaining optimal solutions in 4 of the 6 functions (F9, F10, F11, F13). In addition, AO ranked first in 5, outperforming other MH algorithms on F5, F11, and F21, second only to LICRSA. In addition, GSA showed dominance in the fixed dimension, achieving the best runs on F16, F17, F19, F20, F22, and F23. The results show that LICRSA has more robust solution accuracy than other existing algorithms in unimodal and multimodal functions when solving the 23 benchmark problems, indicating that LICRSA can obtain a greater chance of convergence success to the optimal value resulting in good development performance. It performs better than most search algorithms in the fixed dimension. Figure 5 provides a bar plot of the mean rank of different algorithms.

Table 2. Numerical results for best, worst, mean, std of the search algorithm in 12 of the 23 benchmark functions.
|        | Worst | Mean | Std | Rank | 1 | 3 | 5 | 8 | 10 | 12 | 7 | 11 | 4 | 9 | 6 | 1 |
|--------|-------|------|-----|------|---|---|---|---|----|----|---|---|---|---|---|---|
| Best   | 6.29  | 8.32 | 1.52|     | 6  | 7 |   |   |    |    |   |   |   |   |   |   |
| Worst  | 28.99 | 27.71| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Mean   | 6.72  | 6.83 | 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Std    | 13.09 | 0.002| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Rank   | 1     | 3    | 5   | 8   | 10 | 12 | 7 | 11 | 4   | 9  | 6  | 1  | 0 | 0 | 0 |

|        | Worst | Mean | Std | Rank | 1 | 3 | 5 | 8 | 10 | 12 | 7 | 11 | 4 | 9 | 6 | 1 |
|--------|-------|------|-----|------|---|---|---|---|----|----|---|---|---|---|---|---|
| Best   | 6.72  | 6.83 | 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Worst  | 28.99 | 27.71| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Mean   | 6.72  | 6.83 | 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Std    | 13.09 | 0.002| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Rank   | 1     | 3    | 5   | 8   | 10 | 12 | 7 | 11 | 4   | 9  | 6  | 1  | 0 | 0 | 0 |

|        | Worst | Mean | Std | Rank | 1 | 3 | 5 | 8 | 10 | 12 | 7 | 11 | 4 | 9 | 6 | 1 |
|--------|-------|------|-----|------|---|---|---|---|----|----|---|---|---|---|---|---|
| Best   | 6.72  | 6.83 | 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Worst  | 28.99 | 27.71| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Mean   | 6.72  | 6.83 | 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Std    | 13.09 | 0.002| 1.52|     | 2  | 3 |   |   |    |    |   |   |   |   |   |   |
| Rank   | 1     | 3    | 5   | 8   | 10 | 12 | 7 | 11 | 4   | 9  | 6  | 1  | 0 | 0 | 0 |
|        | Mean       | Std        | Rank |
|--------|------------|------------|------|
|        | 8.88 × 10^{-16} | 0.0035      | 1    |
|        | 8.88 × 10^{-16} | 0.0109      | 9    |
|        | 1.79 × 10^{-4}  | 0.6814      | 5    |
|        | 4.82 × 10^{-11} | 2.27 × 10^{-15} | 8    |
|        | 19.9648      | 0.1480      | 1    |
|        | 0.0160       | 3.6012      | 1    |
|        | 3.6012       | 0.0109      | 8    |
|        | 1.11 × 10^{-4} | 3.0000      | 1    |
|        | 1.84 × 10^{-10} | 0.3979      | 8    |
|        | 1.11 × 10^{-4} | 5.0000      | 9    |
|        | 4.20 × 10^{-15} | 5.28 × 10^{-14} | 8    |
|        | 2.90 × 10^{-15} | 3.07 × 10^{-15} | 8    |
|        | 8.88 × 10^{-16} | 0.1480      | 5    |
|        | 8.88 × 10^{-16} | 5.30 × 10^{-16} | 1    |
|        | 2.90 × 10^{-15} | 6.30 × 10^{-16} | 1    |
|        | 8.88 × 10^{-16} | 8.29 × 10^{-16} | 1    |
|        | 8.88 × 10^{-16} | 1.15 × 10^{-16} | 1    |
|        | 8.88 × 10^{-16} | 2.76 × 10^{-16} | 1    |
|        | 8.88 × 10^{-16} | 12.6705     | 1    |
|        | 8.88 × 10^{-16} | 0.0017      | 1    |
|        | 8.88 × 10^{-16} | 1.0480      | 9    |
| Best   | 0.9980      | 0.9980      | 1    |
| Worst  | 0.9980      | 0.9980      | 9    |
| Mean   | 0.9980      | 0.9980      | 5    |
| Std    | 0.9980      | 0.9980      | 5    |
| Rank   | 0.9980      | 0.9980      | 5    |
|        | 1.33 × 10^{-8} | 0.0003      | 12   |
|        | 0.6944      | 0.0003      | 7    |
|        | 10.7632     | 0.0003      | 12   |
|        | 5.0000      | 0.0003      | 7    |
| F1     | 1.15 × 10^{-6} | 0.0003      | 12   |
|        | 0.6944      | 0.0003      | 7    |
|        | 10.7632     | 0.0003      | 12   |
|        | 5.0000      | 0.0003      | 7    |
| Rank   | 0.9980      | 0.9980      | 5    |
| Best   | 0.9980      | 0.9980      | 1    |
| Worst  | 0.9980      | 0.9980      | 9    |
| Mean   | 0.9980      | 0.9980      | 5    |
| Std    | 0.9980      | 0.9980      | 5    |
| Rank   | 0.9980      | 0.9980      | 5    |
|        | 6.28 × 10^{-32} | 0.28235     | 10   |
|        | 3.9039      | 0.28235     | 10   |
| F14    | 3.4254      | 0.28235     | 10   |
|        | 10.5733     | 0.28235     | 10   |
|        | 5.3937      | 0.28235     | 10   |
|        | 4.0939      | 0.28235     | 10   |
|        | 2.8235      | 0.28235     | 10   |
|        | 3.11 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
|        | 3.09 × 10^{-5} | 0.28235     | 10   |
| Rank   | 0.9980      | 0.9980      | 5    |
| Best   | 0.0002      | 0.0002      | 1    |
| Rank   | 0.9980      | 0.9980      | 5    |
| Best   | 0.9980      | 0.9980      | 1    |
| Rank   | 0.9980      | 0.9980      | 5    |
|     | Mean          | Std          | Rank | 10 |  6 | 12 |  9 |  8 |  3 |  1 |  5 |  7 | 11 |  2 |  4 |
|-----|---------------|--------------|------|----|----|----|----|----|----|----|----|----|----|----|----|
| F19 | Best          | −3.8625      | 10   |  6 | 12 |  9 |  8 |  3 |  1 |  5 |  7 | 11 |  2 |  4 |     |
|     | Worst         | −3.7354      |      |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Mean          | −3.8285      |      |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Std           | 0.0325       |      |    |    |    |    |    |    |    |    |    |    |    |     |
| F20 | Best          | −3.0811      | 12   |  9 |  6 |  10 |  2 |  3 |  7 |  5 |  11 |  10 |     |
|     | Worst         | −2.1505      |      |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Mean          | −2.7328      |      |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Std           | 0.2495       |      |    |    |    |    |    |    |    |    |    |    |    |     |
| F21 | Best          | −5.0552      | 11   |  1 |  3 |  4 |  9 |  5 |  2 |  6 |  7 | 12 |  10 |  8 |     |
|     | Worst         | −5.0552      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Mean          | −5.0552      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Std           | 3.56 × 10^{-7} | 0.0124 | 2.0586 | 2.5136 | 3.1874 | 2.1675 | 1.3639 | 3.2052 | 3.2236 | 1.9158 | 3.5292 | 3.3801 |     |
| F22 | Best          | −6.0680      | 11   |  3 |  8 |  6 |  10 |  4 |  1 |  5 |  7 | 12 |  10 |  8 |     |
|     | Worst         | −5.0877      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Mean          | −5.1203      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Std           | 0.1790       |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
| F23 | Best          | −5.1244      | 11   |  2 |  8 |  4 |  9 |  5 |  3 |  7 |  6 | 12 |  10 |  10 |     |
|     | Worst         | −5.1285      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Mean          | −5.1667      |      |    |    |    |    |    |    |    |    |    |    |    |    |     |
|     | Std           | 0.2092       |      |    |    |    |    |    |    |    |    |    |    |    |    |     |

**Figure 5.** Mean rank of different algorithms.
In this section, the convergence analysis of the search algorithm is performed on 23 benchmark test suites. Figure 6 provides the number of functions better than LICRSA, significantly the same as LICRSA, and worse than LICRSA. Figure 7 shows the results of the problems for the 23 test problems. Some of these functions have images for which we have taken logarithmic data processing. From the results in the figure, it can be noticed that LICRSA starts with a fast convergence and mutation behavior on most of the problems, which is more pronounced than the other tested algorithms. This result is due to the excellent exploration capability of the added interaction crossover strategy. Moreover, the results display that LICRSA has the fastest convergence on most functions than AO, AOA, RSA, CSO, DE, GSA, WOA, WHOA, DMO, SOA, and WSO. This result is due to the reduced search space and enhanced exploitation capability at iteration time by Lévy flight and crossover strategies. Also, from the convergence curves, LICRSA converges quickly to the global optimum for most unimodal and multimodal functions (F1, F3, F5, F9, F10, F11), indicating that LICRSA has better convergence capability. The results further verify the effectiveness of the Lévy flight and the interaction crossover strategy, which enhances the local search capability. In addition, Figure 8 provides the box plots of the 23 test functions, from which it is clear that the box of LICRSA is narrower, which also indicates the stability ability of LICRSA for multiple runs, and then LICRSA has some instability in the face of certain fixed dimensional test functions.

Table 3 provides the final results of p-values and statistics between each algorithm and LICRSA. 1/11/11, 6/6/11, 0/6/17, 5/1/17, 1/6/16, 3/6/14 for RSA, AO, AOA, CSO, DE, and DMO, respectively, and 7/4/12, 7/4/12, 7/7/13, 7/7/13, and 7/7/13 for WSO, WOA, WHOA, SOA, and GSA. 3/7/13, 7/8/8, 1/6/16, 5/6/12.

Figure 6. The number of better than, worse than, same as LICRSA.
Figure 7. Convergence curves of LICRSA and different MH algorithms: (a) F01; (b) F02; (c) F03; (d) F04; (e) F05; (f) F06; (g) F07; (h) F08; (i) F09; (j) F10; (k) F11; (l) F12; (m) F13; (n) F14; (o) F15; (p) F16; (q) F17; (r) F18; (s) F19; (t) F20; (u) F21; (v) F22; (w) F23.
Figure 8. Box plot of LICRSA and RSA, AO, AOA, CSO, DMO, DE, WSO, SOA, GSA, WHOA, WOA:
(a) F01; (b) F02; (c) F03; (d) F04; (e) F05; (f) F06; (g) F07; (h) F08; (i) F09; (j) F10; (k) F11; (m) F13; (n) F14; (o) F15; (p) F16; (q) F17; (r) F18; (s) F19; (t) F20; (u) F21; (v) F22; (w) F23.

Table 3. p-value and statistical results.

| p-Value | RSA | AO | AOA | CSO | DE | DMO | WSO | WHOA | SOA | GSA |
|---------|-----|----|-----|-----|----|-----|-----|------|-----|-----|
| F1      | NaN/=| 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | NaN/= | 1.21 × 10⁻¹³/− |
| F2      | NaN/=| 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.93 × 10⁻¹³/− | 1.21 × 10⁻¹³/− |
| F3      | NaN/=| 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− |
| F4      | NaN/=| 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− | 1.21 × 10⁻¹³/− |
| F5      | 1.95 × 10⁻³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− |
| F6      | 3.69 × 10⁻¹³/− | 1.09 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 3.02 × 10⁻¹³/− | 4.98 × 10⁻¹³/− | 3.02 × 10⁻¹³/− |
| F7      | 0.8073 /= | 0.0646 /= | 0.4733 /= | 0.032 × 10⁻¹³/− | 0.032 × 10⁻¹³/− | 0.032 × 10⁻¹³/− | 0.032 × 10⁻¹³/− | 8.09 × 10⁻¹³/− | 1.41 × 10⁻⁹/− | 0.0625 /= | 3.02 × 10⁻¹³/− |
4.3. Performance Evaluation for the CEC-2020 Test Functions

These 10 CEC2020 include a multi-peak function, 3 BASIC functions, 3 hybrid functions, and 3 composite functions [5]. The multimodal functions are usually employed to understand the development of the search algorithm, and hybrid functions can better combine the properties of sub-functions and maintain the continuity of global/local optima. The composite function combines various hybrid functions and can be employed to evaluate the convergence accuracy of the algorithm. Note that the objective of all 10 CEC2020 functions is also the minimization problem, and again, this section still uses the maximum, minimum, average, and standard deviation (Std) in the results of the numerical experiments. In addition, time metrics and p-values are also important factors in the evaluation system. In addition, the Wilcoxon signed-rank test (significance level = 0.05) is used to test the numerical cases; where “+” indicates that an algorithm is more accurate than LICRSA, “=” means that LICRSA is not different from MH, and “-” shows that a method is less accurate than LICRSA.

Table 4 provides a comparison between LICRSA and different MH algorithms. The table indicates that LICRSA has the best running results in four test functions, cec02, cec04, cec05, cec10, and the second ranking in cec06, cec07, and cec09, achieving the second-ranking. The overall average order of LICRSA of 2.9 is better than the other MH algorithms, indicating that LICRSA can handle this test function well. Figures 9 and 10 provide the iterative and box-line plots for the 10 test functions of this test set, respectively, from which it is found that LICRSA can find the neighborhood of the optimal case in the earlier
part of the iteration for most of the test functions, and has a faster convergence rate compared to the other comparative search algorithms. Additionally, in Figure 10 it can be found that LICRSA tends to have a narrower box. It has very stable results when run many times. Figure 11 provides the radar plots of the 12 algorithms, and the more concentrated shaded parts represent a smaller rank ranking and better performance in CEC2020. It can be seen from the figure that LICRSA has a more minor shaded part.

Tables 5 and 6 provide the p-values of all the search algorithms run in the CEC2020 test set and the running time statistics, respectively. From the tables, it can be found that RSA, AO, AOA, DMO, DE, and GOA are 0/1/9, 1/3/6, 0/2/8, 3/2/5, 0/3/7, 1/1/8, while GSA, HBA, HHO, SSA, and WOA are 3/1/6, 3/4/3, 1/1/8, 3/1/6, 1/0/9. In Table 6, it can be found that LICRSA takes a longer running time, but the increase in running time is a typical situation due to the added improvement policy, while the running time of RSA itself is longer, and the increase in running time of LICRSA compared to RSA is not too much.
Figure 9. The convergence curves of the LICRSA and search algorithms on CEC2020: (a) cec01; (b) cec02; (c) cec03; (d) cec04; (e) cec05; (f) cec06; (g) cec07; (h) cec08; (i) cec09; (j) cec10.
Figure 10. Box plot of the LICRSA and other MH algorithms on CEC2020: (a) cec01; (b) cec02; (c) cec03; (d) cec04; (e) cec05; (f) cec06; (g) cec07; (h) cec08; (i) cec09; (j) cec10.
Table 4. Results of LICRSA and MH algorithms on CEC2020.

| Function | Index | Algorithms |
|----------|-------|------------|
| RSA      | AO    | AOA        | DMO | DE | GOA | GSA | HBA | HHO | SSA | WOA | LIC-RSA |
| Cec01    |       |            |     |    |     |     |     |     |     |     |         |
|          |      |            |     |    |     |     |     |     |     |     |         |
| Best     | 5.25  | 2.31       | 1.17 | 1.08 | 1.14 | 405.052 | 100.172 | 100.795 | 1.53 | 100.056 | 9.64 | 2.33 |
|          | 10^6  | 10^5       | 10^10 | 10^5 | 10^5 | 5 | 5 | 5 | 10^5 | 6 | 10^3 | 10^5 |
| Worst    | 1.95  | 5.25       | 2.28 | 2.32 | 1.94 | 1749.40 | 1.27 | 1.97 | 1.03 | 5.97 | 4.91 |
|          | 10^6  | 10^10      | 10^4 | 10^6 | 92 | 10^4 | 10^4 | 10^4 | 10^7 |
| Mean     | 1.21  | 1.07       | 1.80 | 3.69 | 1.77 | 612.40 | 2439.67 | 5.64 | 2.54 | 7.75 | 1.24 |
|          | 10^6  | 10^10      | 10^3 | 10^6 | 7 | 84 | 10^3 | 10^3 | 10^7 |
| Std      | 3.67  | 9.43       | 2.92 | 5.14 | 5.02 | 316 | 569.184 | 3604.00 | 4.41 | 2.54 | 1.15 | 1.08 |
|          | 10^6  | 10^6       | 10^8 | 10^6 | 7 | 83 | 10^6 | 10^3 | 10^7 |
| Rank     | 11    | 6          | 12   | 3   | 10 | 9 | 4 | 5 | 2 | 7 | 8 |
| Cec02    |       |            |     |    |     |     |     |     |     |     |         |
|          |      |            |     |    |     |     |     |     |     |     |         |
| Best     | 2233.82 | 1455.43  | 1933.08 | 2260.59 | 1496.37 | 1361.89 | 1778.86 | 1133.52 | 1580.22 | 1454.56 | 1489.74 | 1417.85 |
|          | 85 | 70 | 66 | 62 | 98 | 31 | 92 | 53 | 49 | 24 | 13 |
| Worst    | 3010.59 | 2375.25  | 2592.47 | 2853.31 | 3513.45 | 3086.16 | 3144.76 | 2862.73 | 2601.75 | 2553.64 | 2789.98 | 2055.77 |
| Mean     | 2670.15 | 2009.30  | 2277.72 | 2601.10 | 2583.79 | 2064.78 | 2562.14 | 1823.76 | 2124.17 | 1966.28 | 2178.02 | 1808.09 |
|          | 86  | 83 | 83 | 83 | 52 | 77 | 61 | 13 | 48 | 27 | 86 |
| Std      | 196.350 | 227.093  | 180.522 | 152.303 | 484.985 | 390.764 | 337.913 | 361.730 | 258.754 | 312.441 | 345.704 | 174.328 |
|          | 5   | 8   | 3   | 5   | 3   | 5   | 0   | 4   | 3   | 7   | 3 |
| Rank     | 12  | 4   | 8   | 11  | 10  | 5   | 9   | 2   | 6   | 3   | 7 |
| Cec03    |       |            |     |    |     |     |     |     |     |     |         |
|          |      |            |     |    |     |     |     |     |     |     |         |
| Best     | 785.783 | 723.202  | 779.046 | 729.120 | 721.370 | 720.382 | 713.106 | 717.630 | 750.680 | 720.598 | 736.813 | 724.473 |
|          | 6   | 8   | 9   | 1   | 2   | 1   | 2   | 1   | 5   | 0   | 6   | 0 |
| Worst    | 829.316 | 795.291  | 811.578 | 746.932 | 836.863 | 747.042 | 720.390 | 744.970 | 832.099 | 782.134 | 821.409 | 754.712 |
| Mean     | 809.443 | 751.488  | 798.330 | 736.777 | 760.192 | 730.718 | 716.647 | 731.283 | 786.196 | 738.331 | 780.585 | 745.038 |
|          | 9   | 3   | 4   | 4   | 7   | 1   | 2   | 3   | 3   | 5   | 3 |
| Std      | 10.0994 | 14.9263  | 9.3792 | 5.1123 | 30.1127 | 7.4560 | 2.0621 | 7.6149 | 21.8597 | 13.4556 | 22.5280 | 7.0427 |
|          | 12  | 7   | 11  | 4   | 8   | 2   | 1   | 3   | 10  | 5   | 9   | 6 |
| Cец04    |       |            |     |    |     |     |     |     |     |     |         |
|          |      |            |     |    |     |     |     |     |     |     |         |
| Best     | 1900  | 1900 | 1900 | 1901.27 | 1900.93 | 1900.43 | 1900.48 | 1900 | 1900 | 1900.76 | 1900 | 1900 |
|          | 52  | 28 | 18 | 26 | 18 | 18 | 18 | 18 | 18 | 1900 | 69 |
| Worst    | 1902.68 | 2004.62  | 83 | 75 | 79 | 92 | 92 | 92 | 92 | 1903.71 | 1900.68 | 22 |
| Mean     | 1902.01 | 1912.22  | 51 | 34 | 87 | 36 | 36 | 36 | 36 | 1901.89 | 1900.65 | 1900 |
|          | 0 | 2.11 × 10^-1 | 0 | 0.3909 | 22.9276 | 0.6453 | 0.2516 | 0 | 0 | 0.7755 | 0.1920 | 0 |
| Rank     | 1   | 6   | 1   | 12  | 11  | 9   | 8   | 1   | 1   | 10   | 7   | 1 |

Figure 11. Radar plot of 12 algorithms in CEC2020: (a) RSA; (b) AO; (c) AOA; (d) DMO; (e) DE; (f) GOA; (g) GSA; (h) HBA; (i) HHO; (j) SSA; (k) WOA; (l) LICRSA.
| Name | Std | Mean | Worst | Best |
|------|-----|------|-------|------|
| Cec05 | 8.90 | 23.33 | 24.41 | 25.00 |
| Cec06 | 9.00 | 23.33 | 24.41 | 25.00 |
| Cec07 | 9.00 | 23.33 | 24.41 | 25.00 |
| Cec08 | 9.00 | 23.33 | 24.41 | 25.00 |
| Cec09 | 9.00 | 23.33 | 24.41 | 25.00 |
| Cec10 | 9.00 | 23.33 | 24.41 | 25.00 |

Mean Rank = 6

Final Ranking

- Mean Rank
- Final Ranking

**Mathematics 2022, 10, 2329**
Table 5. p-value and statistical results on CEC2020.

| p-Value | RSA      | AO      | AOA     | DMO     | DE      | GOA     | GSA     | HBA     | HHO     | SSA     | WOA     |
|---------|----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| CEC01   | 3.02 × 10^{-11} | 9.06 × 10^{-11} | 3.02 × 10^{-11} | 8.66 × 10^{-11} | 8.48 × 10^{-11} | 3.02 × 10^{-11} | 3.02 × 10^{-11} | 1.55 × 10^{-11} | 3.02 × 10^{-11} | 0.0468/+ |
| CEC02   | 3.02 × 10^{-11} | 0.0001/- | 2.15 × 10^{-11} | 3.02 × 10^{-11} | 1.70 × 10^{-11} | 9.21 × 10^{-11} | 6.72 × 10^{-11} | 0.8073/- | 5.86 × 10^{-11} | 0.0378/- | 3.16 × 10^{-11} |
| CEC03   | 3.02 × 10^{-11} | 0.0933/= | 2.97 × 10^{-11} | 5.46 × 10^{-11} | 0.0594/= | 1.85 × 10^{-11} | 3.02 × 10^{-11} | 4.31 × 10^{-11} | 2.15 × 10^{-11} | 0.0017/+ | 1.07 × 10^{-11} |
| CEC04   | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= | NaN/= |
| CEC05   | 3.02 × 10^{-11} | 1.31 × 10^{-11} | 3.02 × 10^{-11} | 3.50 × 10^{-11} | 4.31 × 10^{-11} | 0.0016/- | 3.02 × 10^{-11} | 0.0163/- | 4.18 × 10^{-11} | 0.0108/- | 8.15 × 10^{-11} |
| CEC06   | 3.02 × 10^{-11} | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- | 10.0/- |
| CEC07   | 3.02 × 10^{-11} | 1.07 × 10^{-11} | 0.9000/= | 3.82 × 10^{-11} | 0.8303/= | 2.83 × 10^{-11} | 3.02 × 10^{-11} | 0.1715/- | 5.46 × 10^{-11} | 7.60 × 10^{-11} | 3.02 × 10^{-11} |
| CEC08   | 3.02 × 10^{-11} | 0.0002/- | 3.02 × 10^{-11} | 1.39 × 10^{-11} | 0.2009/- | 0.5793/- | 2.64 × 10^{-11} | 5.53 × 10^{-11} | 3.81 × 10^{-11} | 6.74 × 10^{-11} | 7.09 × 10^{-11} |
| CEC09   | 1.78 × 10^{-11} | 0.2838/- | 5.60 × 10^{-11} | 0.4733/- | 0.1018/- | 2.88 × 10^{-11} | 0.1567/- | 0.6520/- | 2.60 × 10^{-11} | 0.5201/- | 10.0/- |
| CEC10   | 3.02 × 10^{-11} | 4.74 × 10^{-11} | 3.02 × 10^{-11} | 0.0261/- | 1.03 × 10^{-11} | 0.0040/- | 1.33 × 10^{-11} | 0.0015/- | 0.0224/- | 0.0008/- | 5.57 × 10^{-11} |

Table 6. Run time of LICRSA and other search algorithms on CEC2020.

| Time    | RSA     | LICRSA  | AO      | AOA     | DMO     | DE      | GOA     | GSA     | HBA     | HHO     | SSA     | WOA     |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| cec01   | 26.1783 | 36.7426 | 9.8169  | 3.9698  | 37.9297 | 6.2376 | 50.7474 | 10.6257 | 5.9571  | 10.0409 | 3.2723  | 3.5769  |
| cec02   | 26.5651 | 37.6956 | 11.1770 | 4.5525  | 38.8434 | 6.7670 | 51.5997 | 11.0695 | 6.3650  | 11.7236 | 3.7308  | 4.1962  |
| cec03   | 27.0933 | 37.3185 | 10.8828 | 4.5389  | 39.1830 | 6.2591 | 51.1633 | 11.0190 | 6.4026  | 11.3222 | 3.6350  | 4.2131  |
| cec04   | 26.3118 | 36.8044 | 10.4214 | 4.2247  | 37.9658 | 5.8460 | 50.2218 | 10.7967 | 6.3192  | 11.1092 | 3.4310  | 3.7895  |
| cec05   | 26.6952 | 37.3432 | 10.7070 | 4.3864  | 38.8799 | 5.9315 | 51.8478 | 10.8394 | 6.2262  | 10.7683 | 3.5137  | 4.0030  |
| cec06   | 26.5340 | 37.1979 | 10.6527 | 4.3489  | 38.7714 | 5.9239 | 52.6064 | 10.8391 | 6.5493  | 10.8512 | 3.5037  | 3.9887  |
| cec07   | 26.5542 | 37.1876 | 10.6808 | 4.2912  | 38.9339 | 5.8890 | 53.1767 | 10.7757 | 5.9273  | 11.1000 | 3.4723  | 3.9535  |
| cec08   | 28.1748 | 40.4331 | 14.0198 | 6.0621  | 42.2293 | 7.5555 | 55.7892 | 12.5570 | 7.5857  | 14.9357 | 5.0989  | 5.5953  |
| cec09   | 28.6854 | 41.2043 | 14.7554 | 6.4715  | 49.3863 | 8.0228 | 52.8070 | 12.9327 | 8.1251  | 15.6190 | 5.4950  | 5.9161  |
| cec10   | 28.1078 | 40.1146 | 13.5807 | 5.8813  | 43.0863 | 7.3519 | 52.2190 | 12.3228 | 7.5167  | 14.3918 | 4.9666  | 5.3896  |

5. Case Studies of Real-World Applications

In this section, the performance of LICRSA is further evaluated through five engineering optimization examples. Also, in this experiment, the constraints are treated by linearly weighting the constraints into the objective function. This approach is commonly used [54]. Typically, the optimization equation with a minimization objective is defined as:

Minimize:

\[ f(Z), \quad Z = [z_1, z_2, \ldots, z_n] \]  (27)

Subject to:

\[
\begin{align*}
g_i(Z) & \leq 0, \quad i = 1, \ldots, m. \\
h_j(Z) & = 0, \quad j = 1, \ldots, l.
\end{align*}
\]  (28)
where \( m \) is the number of constraints and \( l \) is the number of equilibrium constraints. \( Z \) is a candidate solution that is in the feasible domain. Thus, the equation after linear weighting of the constraints to the objective function is described as:

\[
f(Z) = f(Z) + \alpha \sum_{i=1}^{m} \max\{g_i(Z), 0\} + \beta \sum_{j=1}^{l} \max\{h_j(Z), 0\}.
\]  

(29)

where \( \alpha \) is the weight of \( g(Z) \), and \( \beta \) is the weight of \( h(Z) \). In the practical case, to ensure that the solution is in the feasible domain, the candidate solution is severely penalized by larger weights on the objective function when the \( g(Z) \) and \( h(Z) \) constraints are exceeded.

5.1. Welded Beam Design Problem

The ultimate goal of the welded beam design optimization is to minimize the fabrication cost of the welded beam. Four decision variables exist for this design problem, namely: the thickness of the weld \( h \), length of the connected part of the reinforcement \( l \), the height of the reinforcement \( t \), and thickness of the reinforcement \( b \), with seven constraints [4]. The schematic diagram of the welded beam is provided in Figure 12. The mathematical expression of the welded beam optimization is as follows.

Minimize:

\[
f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2),
\]

(30)

Subject to:

\[
g_1(x) = \tau(x) - \tau_{\text{max}} \leq 0,
\]

(31)

\[
g_2(x) = \sigma(x) - \sigma_{\text{max}} \leq 0,
\]

(32)

\[
g_3(x) = \delta(x) - \delta_{\text{max}} \leq 0,
\]

(33)

\[
g_4(x) = x_1 - x_4 \leq 0,
\]

(34)

\[
g_5(x) = P - P_c(x) \leq 0,
\]

(35)

\[
g_6(x) = 0.125 - x_1 \leq 0,
\]

(36)

\[
g_7(x) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0,
\]

(37)

where,

\[
\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{R} + (\tau'')^2},
\]

(38)

\[
\tau' = \frac{P}{\sqrt{2x_1x_2}}, \quad \tau'' = \frac{MR}{J},
\]

(39)

\[
M = P(L + \frac{x_2}{2}),
\]

(40)

\[
R = \sqrt{\frac{x_1^2}{4} + (\frac{x_1 + x_2}{2})^2},
\]

(41)
\[ J = 2 \left\{ \sqrt{2x_1x_2} \left[ \frac{x_2^2}{4} + \left( \frac{x_1 + x_3}{2} \right)^2 \right] \right\}, \quad (42) \]

\[ \sigma(\bar{x}) = \frac{6PL}{x_4x_3^2}, \quad \delta(\bar{x}) = \frac{6PL^3}{Ex_3^2x_4}, \quad (43) \]

\[ P_c(\bar{x}) = \frac{40.35^{\frac{2}{3}}x_3^6}{36} \left( 1 - \frac{x_4}{2L} \sqrt{\frac{E}{4G}} \right), \quad (44) \]

\[ P = 6000 \text{ lb}, \quad L = 14 \text{ in}, \quad \delta_{\text{max}} = 0.25 \text{ in}, \quad E = 30 \times 10^6 \text{ psi}, \quad (45) \]

\[ G = 12 \times 10^6 \text{ psi}, \quad \tau_{\text{max}} = 13600 \text{ psi}, \quad \sigma_{\text{max}} = 30000 \text{ psi} \quad (46) \]

Variable range:

\[ 0.1 \leq x_1 \leq 2, \quad 0.1 \leq x_2 \leq 10, \quad (47) \]

\[ 0.1 \leq x_1 \leq 2, \quad 0.1 \leq x_2 \leq 10, \quad (48) \]

Eleven authoritative and most popular algorithms (AOA [4], WCA [55], EPO [56], LSA [40], GOA [18], ASO [29], GSA [33], HHO [8], SCA [42], WHOA [15], SSA [11]) are selected to ensure reliability during the experimental demonstration and compared with the results of LICRSA for solving and optimizing the welded beam design. The numerical results are presented in Table 7, which displays the design variables and minimum expenditures for each algorithm optimization, and Table 8 provides the statistics for 20 runs. In addition, Table 9 provides the algorithms from the literature (AOA [4], WCA [55], EPO [56], AHA [57], DMO [6], SO [58], HHO [8], SSA [11], COOT [14], HBA [52], GJO [59]) obtained. The table displays that LICRSA finds the optimal results and design variables for the welded beam design. The results show that LICRSA is one of the excellent methods to solve the problem.

Figure 12. Welded beam design.
Table 7. Numerical results for the welded beam design problem.

| Algorithms | Variables | Fabrication Cost |
|------------|-----------|------------------|
|            | $h$       | $l$             | $t$     | $b$     |               |
| AOA        | 0.719368488 | 4.031933267 | 7.182900706 | 0.768872634 | 6.360027463 |
| EPO        | 0.579158195 | 4.831560952 | 7.378944515 | 0.679429586 | 6.260178047 |
| WCA        | 0.503927824 | 5.063922345 | 6.591757927 | 0.594335236 | 5.021144989 |
| LSA        | 0.626915785 | 3.504310919 | 7.046424443 | 0.745973525 | 4.904396997 |
| GOA        | 0.623784118 | 1.731105368 | 5.361811162 | 0.669148118 | 3.073084436 |
| ASO        | 0.112461952 | 7.869657099 | 9.927606782 | 0.237246274 | 2.615188815 |
| GSA        | 0.231152943 | 4.152861485 | 7.833486332 | 0.306215518 | 2.27824162 |
| HHO        | 0.176535238 | 4.215064979 | 8.987259763 | 0.213592280 | 1.810408730 |
| SCA        | 0.190963193 | 3.390762720 | 9.396083298 | 0.210010956 | 1.784850890 |
| WHOA       | 0.240993638 | 2.747836964 | 8.472571962 | 0.241219438 | 1.783366573 |
| SSA        | 0.164264918 | 4.298891853 | 9.063217616 | 0.205643980 | 1.759914368 |
| LICRSA     | 0.201941354 | 3.086318875 | 9.022514058 | 0.206392365 | 1.668814555 |

Table 8. Statistics of 20 independent operations of welded beam.

| Algorithms | Best | Worst | Mean | Std  |
|------------|------|-------|------|------|
| AOA        | 2.690744810 | 11.478409822 | 6.360027463 | 2.220504135 |
| EPO        | 2.141279184 | 18.277839444 | 6.260178047 | 3.535797964 |
| WCA        | 1.984263283 | 18.512920304 | 5.021144989 | 4.069964644 |
| LSA        | 3.017467298 | 7.603956337 | 4.904396997 | 1.306668793 |
| GOA        | 1.900777191 | 4.661038249 | 3.073084436 | 0.726192820 |
| ASO        | 1.814047463 | 5.031409356 | 2.615188815 | 0.818493036 |
| GSA        | 1.949095983 | 2.814336686 | 2.278234162 | 0.244583730 |
| HHO        | 1.668728085 | 2.092938322 | 1.810408730 | 0.133532471 |
| SCA        | 1.711950739 | 1.837690043 | 1.784850890 | 0.032561529 |
| WHOA       | 1.660343003 | 2.377960751 | 1.783366573 | 0.192051576 |
| SSA        | 1.660791846 | 2.008564223 | 1.759914368 | 0.110515407 |
| LICRSA     | 1.658808128 | 1.737698284 | 1.668814555 | 0.019963180 |

Table 9. Numerical comparison of the results obtained for the welded beam design problem in the literature.

| Algorithms | Variables | Fabrication Cost |
|------------|-----------|------------------|
|            | $h$       | $l$             | $t$     | $b$     |               |
| AOA [4]    | 0.194475000 | 2.570920000 | 10.00000000 | 0.201827000 | 1.716400000 |
| EPO [56]   | 0.205411000 | 3.472341000 | 9.03521500 | 0.201153000 | 1.723589000 |
| WCA [55]   | 0.205728000 | 3.470522000 | 9.03662000 | 0.205729000 | 1.726427000 |
| AHA [57]   | 0.205730000 | 3.470492000 | 9.03662400 | 0.205730000 | 1.724853000 |
| DMO [6]    | 0.205570500 | 3.256772400 | 9.03617700 | 0.205769600 | 1.696400000 |
| SO [58]    | 0.205700000 | 3.470500000 | 9.03660000 | 0.205700000 | 1.724851931 |
| HHO [8]    | 0.204039000 | 3.531061000 | 9.02746300 | 0.206147000 | 1.731990570 |
| SSA [11]   | 0.205700000 | 3.471400000 | 9.03660000 | 0.205700000 | 1.724910000 |
| COOT [14]  | 0.198827100 | 3.337971000 | 9.19198600 | 0.198834500 | 1.670301154 |
| HBA [51]   | 0.205700000 | 3.470400000 | 9.03660000 | 0.205700000 | 1.724510000 |
| GJO [59]   | 0.205620000 | 3.471900000 | 9.03920000 | 0.205720000 | 1.725220000 |
| LICRSA     | 0.201941354 | 3.086318875 | 9.02251405 | 0.206392365 | 1.668814555 |
5.2. Pressure Vessel Design Problem

The ultimate requirement for pressure vessel design is to minimize the cost of fabrication, welding, and materials for the pressure vessel [4]. Figure 13 prompts us that a hemispherical head capped the cylindrical vessel at both ends. Four relevant design variables need to be considered for optimization, including shell thickness \(T_s(x_1)\), head thickness \(T_h(x_2)\), internal diameter \(R(x_3)\), and vessel cylindrical cross-section length \(L(x_4)\). The mathematical optimization model for pressure vessel design is as follows:

Minimize:

\[
f(\bar{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_3^2x_4,
\]

Subject to:

\[
g_1(\bar{x}) = -x_1 + 0.0193x_3 \leq 0, \tag{50}
\]

\[
g_2(\bar{x}) = -x_2 + 0.00954x_3 \leq 0, \tag{51}
\]

\[
g_3(\bar{x}) = -\pi x_2^2x_4 - \frac{4}{3} \pi x_3^3 + 1.296,000 \leq 0, \tag{52}
\]

\[
g_4(\bar{x}) = x_4 - 240 \leq 0, \tag{53}
\]

Variable range:

\[
0 \leq x_1, x_2 \leq 99, \tag{54}
\]

\[
10 \leq x_3, x_4 \leq 200, \tag{55}
\]

In this section, eleven different MH algorithms (AOA [4], CSO [48], HHO [8], BOA [26], GSA [33], GBO [13], SMA [22], SSA [11], COOT [14], HGS [39], AO [23]) are used to compare with the experimental results of LICRSA and provide the experimental results. Among them, Table 10 provides the optimization results for the four design variables and the minimum overall cost. Table 11 provides the statistics for 20 independent runs. In addition, Table 12 provides the results obtained by the algorithms (AOA [4], HHO [8], SMA [22], AO [23], SO [58], GJO [59], AVOA [60], AHA [57], COOT [14], SMO [5], CSA [61]) in the literature. According to the table, LICRSA has a minimum cost of 6014.62661, and the results also prove that LICRSA can handle the pressure vessel design problem very well.

| Algorithms | Variables | Minimize Total Cost |
|------------|-----------|---------------------|
| AOA        | x1 = 27.530388, x2 = 34.138394, x3 = 63.069849, x4 = 100.717010 | 26,631.517725 |
| CSO        | x2 = 16.809529, x3 = 8.928204, x4 = 54.953414, x5 = 75.344837 | 6697.878507 |
| HHO        | x2 = 16.299333, x3 = 8.018776, x4 = 52.961474, x5 = 82.816687 | 6454.926298 |
| BOA        | x2 = 16.848615, x3 = 8.220844, x4 = 56.859497, x5 = 60.135991 | 6407.727518 |
| GSA        | x2 = 15.810543, x3 = 7.963037, x4 = 51.272371, x5 = 90.229725 | 6384.404789 |
| GBO        | x2 = 15.295085, x3 = 7.579440, x4 = 49.530667, x5 = 110.866599 | 6294.769796 |
| SMA        | x2 = 14.624979, x3 = 7.242314, x4 = 48.097655, x5 = 132.372275 | 6196.180613 |
| SSA        | x2 = 14.528154, x3 = 7.419590, x4 = 45.315957, x5 = 127.549461 | 6129.945260 |
| COOT       | x2 = 14.326820, x3 = 7.240143, x4 = 46.274177, x5 = 136.306179 | 6063.502432 |
| HGS        | x2 = 14.075195, x3 = 7.034522, x4 = 46.056238, x5 = 147.375773 | 6037.109755 |
| AO         | x2 = 13.586214, x3 = 6.874278, x4 = 45.168063, x5 = 155.115594 | 6014.626610 |

Table 10. Numerical results for the pressure vessel design problem.
Figure 13. Pressure vessel design.

Table 11. Statistics of 20 independent operations of pressure vessel.

| Algorithms | Best         | Worst         | Mean           | Std            |
|------------|--------------|---------------|----------------|----------------|
| AOA        | 5897.462372  | 46,501.56488  | 26,631.51772   | 13,276.53509   |
| CSO        | 5628.027711  | 7674.402414   | 6697.878507    | 601.2257852    |
| HHO        | 5627.760854  | 6960.328263   | 6454.926298    | 330.0697368    |
| BOA        | 4136.939768  | 8959.042006   | 6407.769796    | 1197.932184    |
| GSA        | 5978.230885  | 6917.020959   | 6384.404789    | 195.4921284    |
| GBO        | 5654.370337  | 7332.814508   | 6294.769796    | 601.2257852    |
| SMA        | 5654.370311  | 7332.851947   | 6196.180613    | 660.8464512    |
| SSA        | 5653.718693  | 6820.410235   | 6184.961       | 333.793435     |
| COOT       | 5654.370265  | 6820.410353   | 6129.94526     | 256.5232078    |
| HGS        | 5654.370337  | 7339.600755   | 6063.502432    | 537.420386     |
| AO         | 5618.334168  | 6815.937061   | 6037.109755    | 383.0356993    |
| LICRSA     | 5654.369001  | 5658.427987   | 5654.789764    | 1.222105494    |

Table 12. Numerical comparison of the results obtained for the pressure vessel in the literature.

| Algorithms | Variables | Minimize Total Cost |
|------------|-----------|---------------------|
| AOA [4]    | 0.830374  | 42.751270 62048.784400 |
| HHO [8]    | 0.817584  | 42.091746 62048.784400 |
| SMA [22]   | 0.793100  | 40.671100 5994.185700 |
| AO [23]    | 1.054000  | 40.575200 5887.529768 |
| SO [58]    | 0.781900  | 40.575200 5887.529768 |
| GJO [59]   | 0.778296  | 40.575200 5887.529768 |
| AVOA [60]  | 0.778954  | 40.360312 5886.676593 |
| AHA [57]   | 0.778171  | 40.319618 5885.332950 |
| COOT [14]  | 0.778170  | 40.319618 5885.332950 |
| SMO [5]    | 0.778169  | 40.319618 5885.332950 |
| CSA [61]   | 12.450698 | 45.168063 5654.789764 |
| LICRSA     | 13.586214 | 45.168063 5654.789764 |

5.3. Three-Bar Truss Design Problem

The ultimate goal of the three-bar truss design is to minimize the volume of a statically loaded three-bar truss constrained by the stresses (σ) in each bar. Two design variables exist for this problem as cross-sectional area $A_1 = x_1$ and cross-sectional area $A_2 = x_2$. A
schematic design of the triple truss is given in Figure 14 [5]. The mathematical optimization model of the three-bar truss design is shown below.

Minimize:

\[ f(\bar{x}) = (2\sqrt{x_1} + x_2) * l, \]  

(56)

Subject to:

\[ g_1(\bar{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \]  

(57)

\[ g_2(\bar{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \]  

(58)

\[ g_3(\bar{x}) = \frac{1}{\sqrt{2x_2 + x_1}} P - \sigma \leq 0, \]  

(59)

Variable range:

\[ 0 \leq x_1, x_2 \leq 1, \]  

(60)

where,

\[ l = 100 \text{ cm}, P = 2 \text{ KN/cm}^2, \sigma = 2 \text{ KN/cm}^2. \]  

(61)

The experiments of LICRSA and other well-known MH techniques (ASO [29], SCA [42], CSO [48], G-QPSO [62], AO [23], CSA [33], HHO [8], GWO [20], SSA [11], JS [16], GBO [13]) are provided in Table 13. It is clear from the experimental results that LICRSA outperforms selected optimization methods. The statistical results of various optimizers, including the LICRSA, are shown in Table 14. In addition, Table 15 provides the results obtained by the algorithms (CS [63], AOA [4], GBO [13], GOA [18], MVO [64], GJO [59], SSA [11], HHO [8], AVOA [60], SMO [5], CSA [61]) in the literature. The numerical results in the table prove that LICRSA can find the best minimum volume.

### Table 13. Results for the three-bar truss design.

| Algorithms | Variables | Minimize Volume |
|------------|-----------|----------------|
| ASO        | 0.90514715 | 0.23210052     | 279.22432697 |
| SCA        | 0.83127880 | 0.32661521     | 267.78267280 |
| CSO        | 0.77483698 | 0.45542842     | 264.69983622 |
| G-QPSO     | 0.79402041 | 0.39700591     | 264.28347770 |
| AO         | 0.78613749 | 0.41791460     | 264.14472144 |
| GSA        | 0.78190299 | 0.42840643     | 263.99620601 |
| HHO        | 0.79160624 | 0.40082108     | 263.98216359 |
| GWO        | 0.78910501 | 0.40707138     | 263.89982151 |
| SSA        | 0.78800142 | 0.41017025     | 263.89748524 |
| JS         | 0.78862473 | 0.40839151     | 263.89590834 |
| GBO        | 0.78967699 | 0.40824306     | 263.89584379 |
| LICRSA     | 0.78867514 | 0.40824829     | 263.89584338 |
Figure 14. Three-bar truss design.

Table 14. Statistics of 20 independent operations of three-bar truss design.

| Algorithms | Best     | Worst    | Mean      | Std       |
|------------|----------|----------|-----------|-----------|
| ASO        | 269.1574840 | 282.8427125 | 279.2243270 | 4.3189100 |
| SCA        | 263.9019374 | 282.8427125 | 267.7826728 | 7.7260120 |
| CSO        | 263.8959011 | 270.4633032 | 264.6998362 | 1.5273318 |
| G-QPSO     | 263.9219460 | 265.0597391 | 264.2834777 | 0.3423240 |
| AO         | 263.9245641 | 264.4579632 | 264.1447214 | 0.1553716 |
| GSA        | 263.8974897 | 264.2263056 | 263.9962060 | 0.0817948 |
| HHO        | 263.8968136 | 264.2487053 | 263.9821636 | 0.1092004 |
| GWO        | 263.8968248 | 263.9127430 | 263.8998215 | 0.0038984 |
| SSA        | 263.8958435 | 263.9092891 | 263.8958434 | 2.92 × 10⁻¹⁴ |
| GBO        | 263.8958434 | 263.8958434 | 263.8958434 | 2.92 × 10⁻¹⁴ |

Table 15. Numerical comparison of the results obtained for the three-bar truss design in the literature.

| Algorithms | Variables | Minimize Volume |
|------------|-----------|-----------------|
| CS [63]    | x₁=0.78867000, x₂=0.40902000 | 263.97160000    |
| AOA [4]    | x₁=0.79369000, x₂=0.39426000 | 263.91540000    |
| GBO [13]   | x₁=0.78869300, x₂=0.40819700 | 263.89590000    |
| GOA [18]   | x₁=0.78889756, x₂=0.40761957 | 263.89588150    |
| MVO [64]   | x₁=0.78860276, x₂=0.40845307 | 263.89584990    |
| GJO [59]   | x₁=0.78865716, x₂=0.40829913 | 263.89584390    |
| SSA [11]   | x₁=0.78866541, x₂=0.40827578 | 263.89584340    |
| HHO [8]    | x₁=0.78866282, x₂=0.40828313 | 263.89584340    |
| AVOA [60]  | x₁=0.78868039, x₂=0.40823341 | 263.89584340    |
| SMO [5]    | x₁=0.78867694, x₂=0.40824317 | 263.89584338    |
| CSA [61]   | x₁=0.78867513, x₂=0.40824831 | 263.89584338    |
| LICRSA     | x₁=0.78867514, x₂=0.40824829 | 263.89584338    |

5.4. Rolling Element Bearing Design

The ultimate goal of a rolling element bearing is to increase the dynamic load of the rolling bearing as much as possible. Ten design variables and nine nonlinear constraints exist for the design of rolling element bearings. These ten different design variables are
ball diameter $D_b$, pitch diameter $D_m$, inner and outer raceway curvature coefficients $f_i$ and $f_0$, the number of ball $Z$, $K_{D_{\min}}$, $K_{D_{\max}}$, $\varepsilon$, $e$, and $\zeta$. In addition, five of the design parameters ($K_{D_{\min}}$, $K_{D_{\max}}$, $\varepsilon$, $e$, and $\zeta$) exist only in the constraints and are not in the objective function, so they will indirectly radiate to the internal geometry. Let $[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] = [D_b, D_m, f_i, f_0, Z, K_{D_{\min}}, K_{D_{\max}}, \varepsilon, e, \zeta]$. A schematic diagram of the rolling element bearing structure is shown in Figure 15. The mathematical optimization of the rolling element bearing is formulated as follows.

Figure 15. Rolling element bearing design.

Minimize:

$$f(\bar{x}) = \begin{cases} f_c Z^{2/3} D_b^{1.8} & \text{if } D_b \leq 25.4 \text{mm} \\ 3.647 f_c Z^{2/3} D_b^{1.4} & \text{otherwise} \end{cases}$$

Subject to:

$$g_1(\bar{x}) = Z - \frac{\phi_0}{2 \sin^{-1}(D_b / D_m)} - 1 \leq 0,$$

$$g_2(\bar{x}) = K_{D_{\max}} (D - d) - 2D_b \leq 0,$$

$$g_3(\bar{x}) = 2D_b - K_{D_{\max}} (D - d) \leq 0,$$

$$g_4(\bar{x}) = D_b - \zeta B_n \leq 0,$$

$$g_5(\bar{x}) = 0.5(D + d) - D_n \leq 0,$$

$$g_6(\bar{x}) = D_m - (0.5 + e)(D + d) \leq 0,$$

$$g_7(\bar{x}) = \varepsilon D_b - 0.5(D - D_n - D_b) \leq 0,$$

$$g_8(\bar{x}) = 0.515 - f_i \leq 0,$$

$$g_9(\bar{x}) = 0.515 - f_0 \leq 0,$$

where,
\[
f_c = 37.91 \left\{ 1 + \frac{1 - \gamma}{1 + \gamma} \right\}^{1.72} \left( \frac{f_i(2f_i - 1)}{f_0(2f_0 - 1)} \right)^{0.41/3} 0^{1/3} - 0.3 ,
\]  
\[
\gamma = \frac{D_e \cos(\alpha)}{D_m}, \quad f_i = \frac{r_i}{D_b}, \quad f_0 = \frac{r_0}{D_b},
\]  
\[
\phi_b = 2\pi - 2\cos^{-1}\left\{ \frac{(D - d) - 3(T/4)^2 + (D/2 - T/4 - D_b)^2 - (d/2 + T/4)^2}{2(D - d)/2 - 3(T/4)[D/2 - (T/4) - D_b]} \right\},
\]  
\[
T - D - d - 2D_b, \quad D = 160, d = 90, B_n = 30.
\]

with bounds:
\[
0.5(D + d) \leq D_m \leq 0.6(D + d),
\]
\[
0.15(D - d) \leq D_b \leq 0.45(D - d),
\]
\[
4 \leq Z \leq 50, \quad 0.515 \leq f_0 \leq 0.6,
\]
\[
0.4 \leq K_{p_{\text{max}}} \leq 0.5, \quad 0.6 \leq K_{D_{\text{max}}} \leq 0.7,
\]
\[
0.3 \leq \varepsilon \leq 0.4, \quad 0.02 \leq \varepsilon \leq 0.1, \quad 0.6 \leq \zeta \leq 0.85.
\]

Table 16 provides the design variables and optimal dynamic loads for LICRSA and eleven other MH algorithms (G-QPSO [62], AOA [4], SOA [51], WOA [24], SSA [11], SCA [42], AO [23], CSO [48], HGS [39], HHO [8], HBA [52]). Table 17 provides the statistics for 20 independent runs. In addition, Table 18 provides the results obtained by the algorithms (AHA [57], AVOA [60], HBO [65], GBO [13], CSA [61], TSA [66], SOA [51], EPO [56], MVO [64], COOT [14], HHO [8]) in the literature. The LICRSA can find the optimal design variables and dynamic loading. Thus, LICRSA proves to be a great advantage in dealing with the problem.

Table 16. Numerical results for the rolling element bearing problem.
Table 17. Statistics of 20 independent operations of rolling element bearing.

| Algorithms | Best       | Worst       | Mean       | Std        |
|------------|------------|-------------|------------|------------|
| GQPSO      | 34,869.42553 | 39,975.38601 | 35,135.1343 | 1139.29012 |
| AOA        | 20,268.61547 | 54,937.54545 | 32,181.72474 | 10,755.46822 |
| SOA        | 16,176.43125 | 36,056.93710 | 23,408.13961 | 5031.47835 |
| WOA        | 16,982.83369 | 17,848.50849 | 17,412.79383 | 304.91575 |
| SSA        | 16,972.94537 | 17,915.57397 | 17,224.02375 | 254.19596 |
| SCA        | 16,925.01445 | 17,571.79443 | 17,180.07707 | 186.87150 |
| AO         | 17,023.73892 | 17,114.12475 | 17,070.33342 | 25.72117 |
| CSO        | 16,977.92561 | 17,062.90482 | 17,038.34401 | 28.02089 |
| HGS        | 16,958.20229 | 17,058.76692 | 17,033.62576 | 44.67701 |
| HHO        | 16,958.25317 | 17,058.77361 | 17,003.61421 | 36.71393 |
| HBA        | 16,958.20229 | 17,058.76692 | 16,997.30994 | 44.91810 |
| LICRSA     | 16,958.20229 | 17,058.76692 | 16,994.75722 | 37.14945 |

Table 18. Numerical comparison of the results obtained for the rolling element bearing in the literature.

| Algorithms | x1  | x2  | x3  | x4  | x5  | x6  | x7  | x8  | x9  | x10 |
|------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| AHA [57]   | 125.71841  | 21.42535 | 10.52798 | 0.51500 | 0.51516 | 0.47022 | 0.64082 | 0.30001 | 0.09512 | 0.68224 | 85,547.49822 |
| AVOA [60]  | 125.72272  | 21.42329 | 11.00116 | 0.51500 | 0.51500 | 0.40443 | 0.61868 | 0.30000 | 0.06913 | 0.60247 | 85,359.15785 |
| HBO [65]   | 125.71939  | 21.42322 | 11.00000 | 0.51500 | 0.51500 | 0.48806 | 0.67937 | 0.30001 | 0.06902 | 0.60832 | 85,532.57000 |
| GBO [13]   | 125.00000  | 21.87500 | 11.28817 | 0.51500 | 0.51500 | 0.41484 | 0.62866 | 0.30000 | 0.02033 | 0.67206 | 85,245.06110 |
| CSA [61]   | 125.00000  | 21.41800 | 11.35600 | 0.51500 | 0.51500 | 0.40000 | 0.70000 | 0.30000 | 0.02000 | 0.61200 | 85,201.64100 |
| TSA [66]   | 125.00000  | 21.41750 | 10.94110 | 0.51000 | 0.51500 | 0.40000 | 0.70000 | 0.30000 | 0.02000 | 0.60000 | 85,070.08000 |
| SOA [50]   | 125.00000  | 21.41892 | 10.94123 | 0.51500 | 0.51500 | 0.40000 | 0.70000 | 0.30000 | 0.02000 | 0.60000 | 85,068.05200 |
| EPO [56]   | 125.00000  | 21.41890 | 10.94113 | 0.51500 | 0.51500 | 0.40000 | 0.70000 | 0.30000 | 0.02000 | 0.60000 | 85,067.98300 |
| MVO [64]   | 125.60020  | 21.32250 | 10.97338 | 0.51500 | 0.51500 | 0.50000 | 0.68782 | 0.30135 | 0.03617 | 0.61061 | 84,491.26600 |
| COOT [14]  | 125.00000  | 21.87500 | 10.77700 | 0.51500 | 0.51500 | 0.43190 | 0.65290 | 0.30000 | 0.02000 | 0.60000 | 83,918.49200 |
| HHO [8]    | 125.00000  | 21.00000 | 11.09207 | 0.51500 | 0.51500 | 0.40000 | 0.60000 | 0.30000 | 0.05047 | 0.60000 | 83,011.88322 |
| LICRSA     | 129.55754  | 18.00458 | 4.90002  | 0.59998 | 0.59999 | 0.46100 | 0.67964 | 0.33308 | 0.08799 | 0.60000 | 16,994.75722 |

5.5. Speed Reducer Design

Reducers are one of the most critical components of a gearbox system. The ultimate requirement for the gearbox is weight minimization while satisfying 11 constraints. In addition, seven design variables exist for this design. $x_1$ is the face width, $x_2$ is the tooth mode, $x_3$ represents the number of teeth in the pinion, $x_4$ represents the length of the first shaft between the bearings, the length of the second shaft between the bearings is $x_5$, and $x_6$ and $x_7$ represent the diameters of the first shaft and the second shaft. Figure 16 provides a schematic diagram of the reducer [5]. Also, the mathematical design equation for the reducer design is as follows.

Minimize:

$$f(\mathbf{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_1 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^2 + x_7^2),$$  \hspace{1cm} (81)

Subject to:

$$g_1(\mathbf{x}) = \frac{27}{x_1x_2^2 x_3} - 1 \leq 0,$$  \hspace{1cm} (82)
\[ g_2(\bar{x}) = \frac{397.5}{x_1 x_2 x_3^2} - 1 \leq 0, \quad (83) \]

\[ g_3(\bar{x}) = \frac{1.93x_4^3}{x_2 x_3 x_6^4} - 1 \leq 0, \quad (84) \]

\[ g_4(\bar{x}) = \frac{1.93x_5^3}{x_2 x_3 x_4^4} - 1 \leq 0, \quad (85) \]

\[ g_5(\bar{x}) = \sqrt{\frac{745x_4^2}{x_2 x_3}} + 16.9 \times 10^6 \frac{1}{110.0x_6^3} - 1 \leq 0, \quad (86) \]

\[ g_6(\bar{x}) = \sqrt{\frac{745x_4^2}{x_2 x_3}} + 157.5 \times 10^6 \frac{1}{85.0x_6^3} - 1 \leq 0, \quad (87) \]

\[ g_7(\bar{x}) = \frac{x_2 x_3}{40} - 1 \leq 0, \quad (88) \]

\[ g_8(\bar{x}) = \frac{5x_2}{x_1} - 1 \leq 0, \quad (89) \]

\[ g_9(\bar{x}) = \frac{x_1}{12x_2} - 1 \leq 0, \quad (90) \]

\[ g_{10}(\bar{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0, \quad (91) \]

\[ g_{11}(\bar{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0, \quad (92) \]

Variable range:

\[ 2.6 \leq x_1 \leq 3.6, \quad 0.7 \leq x_2 \leq 0.8, \quad (93) \]

\[ 17 \leq x_3 \leq 28, \quad 7.3 \leq x_4 \leq 8.3, \quad (94) \]

\[ 7.8 \leq x_5 \leq 8.3, \quad 2.9 \leq x_6 \leq 3.9, \quad 5 \leq x_7 \leq 5.5. \quad (95) \]

Table 19 provides the optimal design variables and minimum weights for LICRSA and 11 other MH algorithms (LSO [67], BAT [9], BOA [26], DMO [6], SCA [41], WOA [24], HHO [8], SSA [11], CSO [48], GWO [20], SCSO [41]) for the reducer problem. Table 20 provides the statistics for 20 independent runs. In addition, Table 21 provides the results obtained by the algorithms (GOA [18], GSA [33], SCA [42], EA [68], FA [69], AO [23], MVO
[64], GWO [20], CS [63], MFO [19], AOA [4]) in the literature. The results show that LIC-RSA achieves the best result among the search algorithms for both minimum weights. Therefore, LICRSA can solve the reducer problem well.

**Figure 16.** Speed reducer design.

**Table 19.** Numerical results for the speed reducer design.

| Algorithms | Variables | Minimum Weight |
|------------|-----------|----------------|
| LSO        | 3.551135803 0.736923623 25.12309012 8.20716774 3.810971145 5.46787447 | 5477.462985 |
| BAT        | 3.368142883 0.70000847 22.64438282 8.035242714 3.672563578 5.342158123 | 4213.051971 |
| BOA        | 3.523249019 0.707558217 21.39373073 7.870916877 3.629370446 5.331842865 | 4095.416921 |
| DMO        | 3.598259019 0.708842963 17 7.639422189 3.407858186 5.305442136 | 3111.977751 |
| SCA        | 3.582632258 0.7 17.00413795 7.680864826 3.446360885 5.342929824 | 3101.694019 |
| WOA        | 3.522023781 0.7 17.03522238 7.695659172 3.489935053 5.327565943 | 3086.729202 |
| HHO        | 3.550532222 0.7 17.0298238 7.507127823 3.381293938 5.293488313 | 3037.864187 |
| SSA        | 3.506903016 0.7 17.00000006 7.693799023 3.444231088 5.286766517 | 3033.402389 |
| CSO        | 3.525023295 0.705664363 17 7.3 3.351257141 5.286754733 | 3032.813598 |
| GWO        | 3.502456111 0.700011724 17.00198205 7.573843774 3.358079603 5.28817256 | 3004.756902 |
| SCSO       | 3.500778985 0.700048877 17.00005987 7.723582413 3.353267544 5.286988024 | 3004.480179 |
| LICRSA     | 3.50009504 0.70001495 17.00039814 7.344740534 3.350802205 5.286723295 | 2997.550054 |

**Table 20.** Statistics of 20 independent operations of speed reducer design.

| Algorithms | Best | Worst | Mean  | Std   |
|------------|------|-------|-------|-------|
| LSO        | 3375.49065 6783.32679 | 5477.46298 | 1160.36894 |
| BAT        | 3134.25557 5489.08616 | 4213.05520 | 760.69313 |
| BOA        | 3049.42307 5671.13463 | 4095.41692 | 900.43458 |
| DMO        | 3057.84155 3260.93923 | 3111.97775 | 54.51718 |
| SCA        | 3046.18056 3198.09847 | 3101.69402 | 40.56807 |
| WOA        | 2996.84150 3192.65667 | 3086.72920 | 61.34899 |
| HHO        | 3005.69276 3079.26997 | 3037.86419 | 22.22538 |
| SSA        | 2999.04829 3106.79787 | 3033.40239 | 31.04791 |
| CSO        | 2996.30435 3195.02510 | 3032.81360 | 53.56356 |
| GWO        | 2999.76380 3009.91957 | 3004.75690 | 3.18261 |
| SCSO       | 2997.27696 3013.60422 | 3004.48018 | 4.96548 |
| LICRSA     | 2996.30156 3007.99931 | 2997.55005 | 3.49117 |
Table 21. Numerical comparison of the results obtained for the speed reducer design in the literature.

| Algorithms | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | Minimum Weight |
|------------|-------|-------|-------|-------|-------|-------|-------|----------------|
| GOA        | 3.5126| 0.7033| 17.2246| 7.9131| 7.9627| 3.6567| 5.2784| 3169.32        |
| GSA        | 3.6   | 0.7   | 17    | 8.3   | 7.8   | 3.369658| 5.289224| 3051.12        |
| SCA        | 3.508755| 0.7 | 17 | 7.3 | 7.8 | 3.46102 | 3.6567 | 3.2874 | 3030.563 |
| EA         | 3.50745 | 0.7 | 17 | 7.719674 | 8.080854 | 3.351512 | 5.287051 | 3010.137492 |
| FA         | 3.5021 | 0.7 | 17 | 7.3099 | 7.7476 | 3.3614 | 5.2994 | 3007.7328 |
| AO         | 3.508502 | 0.7 | 17 | 7.392843 | 7.816034 | 3.358073 | 5.286777 | 3002.928 |
| MVO        | 3.50669 | 0.7 | 17 | 7.380933 | 7.815726 | 3.357847 | 5.286786 | 3001.288 |
| GWO        | 3.50669 | 0.7 | 17 | 7.380933 | 7.815726 | 3.357847 | 5.286786 | 3001.288 |
| CS         | 3.5015 | 0.7 | 17 | 7.605   | 7.8181 | 3.352 | 5.2875 | 3000.981 |
| MFO        | 3.4976 | 0.7 | 17 | 7.3 | 7.8 | 3.3501 | 5.2857 | 2998.54 |
| AOA        | 3.50384 | 0.7 | 17 | 7.3 | 7.72933 | 3.35649 | 5.2867 | 2997.9157 |
| LICRSA     | 3.50009504 | 0.70001495 | 17.00039814 | 7.344740534 | 7.826768272 | 3.350802205 | 5.286723295 | 2997.550054 |

6. Discussion

According to the numerical situation of this study, the proposed LICRSA algorithm shows excellent performance compared with other optimization algorithms. The validation experiments are divided into three main parts. The first part compares the LICRSA algorithm with the original RSA and different optimization algorithms on 23 test functions. The LICRSA algorithm ranks first in 10 functions, accounting for 43% of all functions. In addition, the LICRSA algorithm outperforms RSA in 15 functions, implying that introducing the Lévy flight and interactive crossover strategy enhances the exploration and development capabilities. The second part compares the LICRSA algorithm with other optimization algorithms in the CEC2020 test set. After analyzing the comparative metrics such as best, mean, worst, and std of solving the benchmark function, the LICRSA also solves the test function with higher accuracy, better stability, and faster convergence. The final section will discuss the capability of LICRSA with different algorithms in solving five complex engineering examples. The simulation results verify the usefulness and dependability of the LICRSA algorithm for real optimization problems. Of course, there is room for improved algorithm stability when solving engineering optimization problems.

7. Conclusions

This paper proposes an improved reptile search algorithm based on Lévy flight and interactive crossover strategies. To improve the performance of the original RSA algorithm, two main approaches are employed to address its shortcomings. As a general modification technique to improve its optimization capability, the Lévy flight strategy enhances the global exploration of the algorithm and jumps out of local solutions. Meanwhile, the interactive crossover strategy, as a variant of the crossover operator, inherits the local exploitation ability and enhances the fit with iterations. Twenty-three benchmark test functions, the IEEE Conference Evolutionary Computation (CEC2020) benchmark, and five engineering design problems are used to test the proposed LICRSA algorithm and verify the effectiveness of LICRSA. The experimental results show that LICRSA obtains more suitable optimization results and significantly outperforms competing algorithms, including DMO, WOA, HHO, DE, AOA, and WSO. The LICRSA has a more significant global search capability and higher accuracy solutions. In addition, the results of the tested mechanical engineering problems prove that LICRSA outperforms other optimization methods in total cost evaluation criteria. LICRSA has excellent performance and can be utilized by other researchers to tackle complex optimization problems.
Similarly, the applicability of RSA will be investigated. For example, to further improve the spatial search capability, we try to integrate RSA with other general intelligent group algorithms to improve the global performance of RSA. We use RSA to provide optimal parameters for machine learning models in privacy-preserving and social networks in practical applications. Furthermore, the proposed LICRSA algorithm may also have excellent application potency in solving other complex optimization problems, such as shop scheduling problems [70], optimal degree reduction [71], image segmentation [72], shape optimization [73], and feature selection [74], etc.

**Author Contributions:** Conceptualization, L.H. and G.H.; Data curation, Y.W. and Y.G.; Formal analysis, L.H., Y.W. and Y.G.; Funding acquisition, L.H. and G.H.; Investigation, L.H., Y.W., Y.G. and G.H.; Methodology, L.H., Y.W., Y.G. and G.H.; Resources, L.H. and G.H.; Software, L.H., Y.W. and Y.G.; Supervision, G.H.; Validation, G.H.; Visualization, Y.W. and Y.G.; Writing—original draft, L.H., Y.W., Y.G. and G.H.; Writing—review & editing, L.H. and G.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is supported by the Research Fund of Department of Science and Department of Education of Shaanxi, China (Grant No. 21J0615).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** All data generated or analyzed during this study were included in this published article.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Houssiné, E.H.; Rezk, H.; Fathy, A.; Mahdy, M.A.; Nassef, A.M. A modified adaptive guided differential evolution algorithm applied to engineering applications. *Eng. Appl. Artif. Intell.* 2022, 113, 104920.

2. Minh, H.-L.; Sang-To, T.; Wahab, M.A.; Cuong-Le, T. A new metaheuristic optimization based on K-means clustering and its application for structural damage identification in a complex 3D concrete structure. *Knowl. Based Syst.* 2022, 251, 109189.

3. Hu, G.; Zhong, J.; Du, B.; Wei, G. An enhanced hybrid arithmetic optimization algorithm for engineering applications. *Comput. Methods Appl. Mech. Engrg.* 2022, 394, 114901.

4. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The Arithmetic Optimization Algorithm. *Comput. Methods Appl. Mech. Engrg.* 2021, 376, 113609.

5. Zamani, H.; Namidi-Shahraki, M.H.; Gandomi, A.H. Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization. *Comput. Methods Appl. Mech. Engrg.* 2022, 392, 114616.

6. Agushaka, J.O.; Ezugwu, A.E.; Abualigah, L. Dwarf Mongoose Optimization Algorithm. *Comput. Methods Appl. Mech. Engrg.* 2022, 391, 114570.

7. Abualigah, L.; Elaziz, M.A.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired metaheuristic optimizer. *Expert Syst. Appl.* 2022, 191, 116158.

8. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris Hawks Optimization: Algorithm and applications. *Future Gener. Comput. Syst.* 2019, 97, 849–872.

9. Gupta, D.; Arora, J.; Agrawal, U.; Khanna, A.; de Albuquerque, V.H.C. Optimized Binary Bat algorithm for classification of white blood cells. *Measurement* 2019, 143, 180–190.

10. Premkumar, M.; Jangir, P.; Sowmya, R. MOGBO: A new Multiobjective Gradient-Based Optimizer for real-world structural optimization problems. *Knowl.-Based Syst.* 2021, 218, 106856.

11. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* 2017, 114, 163–191.

12. Hu, G.; Zhu, X.; Wei, G.; Chang, C.-T. An improved marine predators algorithm for shape optimization of developable Ball surfaces. *Eng. Appl. Artif. Intell.* 2021, 105, 104417.

13. Ahmadianfar, I.; Bozorg-Haddad, O.; Chu, X. Gradient-based optimizer: A new metaheuristic optimization algorithm. *Inf. Sci.* 2020, 540, 131–159.

14. Naruei, I.; Keynia, F. A new optimization method based on COOT bird natural life model. *Expert Syst. Appl.* 2021, 183, 115352.

15. Naruei, I.; Keynia, F. Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems. *Eng. Comput.* 2021, https://doi.org/10.1007/s00366-021-01438-z.

16. Chou, J.-S.; Truong, D.-N. A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. *Appl. Math. Comput.* 2021, 389, 125535.
17. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* 2012, 17, 4831–4845.

18. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper Optimisation Algorithm: Theory and application. *Adv. Eng. Softw.* 2017, 105, 30–47.

19. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* 2015, 89, 228–249.

20. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* 2014, 69, 46–61.

21. Polap, D.; Wozniak, M. Red fox optimization algorithm. *Expert Syst. Appl.* 2021, 166, 114107.

22. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Gener. Comput. Syst.* 2020, 111, 300–323.

23. Abualigah, L.; Yousri, D.; Abd Elaziz, M.; Ewees, A.A.; Al-qaness, M.A.A.; Gandomi, A.H. Aquila Optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* 2021, 157, 107250.

24. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* 2016, 95, 51–67.

25. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control. Eng.* 2020, 8, 22–34.

26. Arora, S.; Singh, S. Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* 2019, 23, 715–734.

27. Beyer, H.-G.; Schwefel, H.-P. Evolution strategies—A comprehensive introduction. *Neural. Comput.* 2002, 1, 3–52. https://doi.org/10.1023/A:1015059928466.

28. Pourpasha, H.; Farshad, P.; Zeinali Heris, S. Modeling and optimization the effective parameters of nanofluid heat transfer performance using artificial neural network and genetic algorithm method. *Energy Rep.* 2021, 7, 8447–8464.

29. Zhao, W.; Wang, L.; Zhang, Z. Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowl.-Based Syst.* 2019, 163, 283–304.

30. Kaveh, A.; Dadras, A. A novel meta-heuristic optimization algorithm: Thermal exchange optimization. *Adv. Eng. Softw.* 2017, 110, 69–84.

31. Al-Madi, N.; Faris, H.; Mirjalili, S. Binary multi-verse optimization algorithm for global optimization and discrete problems. *Int. J. Mach. Learn. Cybern.* 2019, 10, 3445–3465.

32. Lam, A.Y.S.; Li, V.O.K. Chemical Reaction Optimization: A tutorial. *Memet. Comput.* 2012, 4, 3–17.

33. Rasbedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* 2009, 179, 2232–2248.

34. Mousavirad, S.J.; Ebrahimpour-Komeleh, H. Human mental search: A new population-based metaheuristic optimization algorithm. *Appl. Intell.* 2017, 47, 850–887.

35. Das, B.; Mukherjee, V.; Das, D. Student psychology based optimization algorithm: A new population-based optimization algorithm for solving optimization problems. *Adv. Eng. Softw.* 2020, 146, 102804.

36. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching—Learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. Aided Des.* 2011, 43, 303–315.

37. Hu, G.; Du, B.; Wang, X.; Wei, G. An enhanced black widow optimization algorithm for feature selection. *Knowl. Based Syst.* 2022, 235, 107638.

38. Hu, G.; Li, M.; Wang, X.; Wei, G.; Chang, C.-T. An enhanced manta ray foraging optimization algorithm for shape optimization of complex CCG-Ball curves. *Knowl.-Based Syst.* 2022, 240, 108071.

39. Yang, Y.; Chen, H.; Heidari, A.A.; Gandomi, A.H. Hunger games search: Visions, conception, implementation, data analysis, and towards performance shifts. *Expert Syst. Appl.* 2021, 177, 114864.

40. Shareef, H.; Ibrahim, A.A.; Mutlag, A.H. Lightning search algorithm. *Appl. Soft Comput.* 2015, 36, 315–333.

41. Seyyedabbasi, A.; Kiani, F. Sand Cat swarm optimization: A nature-inspired algorithm to solve global optimization problems. *Eng. Comput.* 2022. https://doi.org/10.1007/s10664-022-01604-x.

42. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl. Based Syst.* 2016, 96, 120–133.

43. Zhou, B.; Lei, Y. Bi-objective grey wolf optimization algorithm combined Levy flight mechanism for the FMC green scheduling problem. *Appl. Soft Comput.* 2021, 111, 107717.

44. Zhang, Y.; Jin, Z.; Zhao, X.; Yang, Q. Backtracking search algorithm with Lévy flight for estimating parameters of photovoltaic models. *Energy Convers. Manag.* 2020, 208, 112615.

45. Dinkar, S.K.; Deep, K. An efficient opposition based Lévy Flight Antlion optimizer for optimization problems. *J. Comput. Sci.* 2018, 29, 119–141.

46. Feng, Z.-K.; Liu, S.; Niu, W.-J.; Li, S.-S.; Wu, H.-J.; Wang, J.-Y. Ecological operation of cascade hydropower reservoirs by elitist-guided gravitational search algorithm with Lévy flight local search and mutation. *J. Hydrod.* 2020, 581, 124425.

47. Gao, J.; Gao, F.; Ma, Z.; Huang, N.; Yang, Y. Multi-objective optimization of smart community integrated energy considering the utility of decision makers based on the Lévy flight improved chicken swarm algorithm. *Sustain. Cities Soc.* 2021, 72, 103075.

48. Meng, A.-B.; Chen, Y.-C.; Yin, H.; Chen, S.-Z. Crisscross optimization algorithm and its application. *Knowl. Based Syst.* 2014, 67, 218–229.

49. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Glob. Optim.* 1997, 11, 341–359.

50. Braik, M.; Hammouri, A.; Atwan, J.; Al-Betar, M.A.; Awadallah, M.A. White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. *Knowl. Based Syst.* 2022, 243, 108457.
51. Dhiman, G.; Kumar, V. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. Knowl.-Based Syst. 2019, 165, 169–196.

52. Hashim, F.A.; Houssine, E.H.; Hussain, K.; Mabrouk, M.S.; Al-Atabany, W. Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. Math. Comput. Simul. 2022, 192, 84–110.

53. Hussain, K.; Salleh, M.N.M.; Cheng, S.; Shi, Y. On the exploration and exploitation in popular swarm-based metaheuristic algorithms. Neural Comput. Appl. 2019, 31, 7665–7683.

54. Coello Coello, C.A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Methods Appl. Mech. Engrg. 2002, 191, 1245–1287.

55. Eskandar, H.; Sadollah, A.; Bahreininejad, A.; Hamdi, M. Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput. Struct. 2012, 110–111, 151–166.

56. Dhiman, G.; Kumar, V. Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. Knowl.-Based Syst. 2018, 159, 20–50.

57. Zhao, W.; Wang, L.; Mirjalili, S. Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications. Comput. Methods Appl. Mech. Engrg. 2022, 388, 114194.

58. Hashim, F.A.; Hussien, A.G. Snake Optimizer: A novel meta-heuristic optimization algorithm. Knowl. Based Syst. 2022, 242, 108320.

59. Chopra, N.; Mohsin Ansari, M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications. Expert Syst. Appl. 2022, 198, 116924.

60. Abdollahzadeh, B.; Gharechopogh, F.S.; Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. Comput. Ind. Eng. 2021, 158, 107408.

61. Braik, M.S. Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems. Expert Syst. Appl. 2021, 174, 114685.

62. dos Santos Coelho, L. Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. Expert Syst. Appl. 2010, 37, 1676–1683.

63. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. Eng. Comput. 2013, 29, 17–35.

64. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-Verse Optimizer: A nature-inspired algorithm for global optimization. Neural Comput. Appl. 2016, 27, 495–513.

65. Askari, Q.; Saeed, M.; Younas, I. Heap-based optimizer inspired by corporate rank hierarchy for global optimization. Expert Syst. Appl. 2020, 161, 113702.

66. Kaur, S.; Awasthi, L.K.; Sangal, A.L.; Dhiman, G. Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. Eng. Appl. Artif. Intell. 2020, 90, 103541.

67. Liu, J.; Li, D.; Wu, Y.; Liu, D. Lion swarm optimization algorithm for comparative study with application to optimal dispatch of cascade hydropower stations. Appl. Soft Comput. 2020, 87, 105974.

68. Mezura-Montes, E.; Coello, C.A.C.; Landa-Becerra, R. Engineering optimization using simple evolutionary algorithm. In Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence, Sacramento, CA, USA, 5 November 2003; pp. 149–156.

69. Baykasoglu, A.; Ozsoydan, F.B. Adaptive firefly algorithm with chaos for mechanical design optimization problems. Appl. Soft Comput. 2015, 36, 152–164.

70. Ghasemi, A.; Ashoori, A.; Heavey, C. Evolutionary Learning Based Simulation Optimization for Stochastic Job Shop Scheduling Problems. Appl. Soft Comput. 2021, 106, 107309.

71. Hu, G.; Dou, W.; Wang, X.; Abbas, M. An enhanced chimp optimization algorithm for optimal degree reduction of Said–Ball curves. Math. Comput. Simul. 2022, 197, 207–252.

72. Larabi Marie-Sainte, S.; Alsikreem, R.; Alhalawani, S. Emerging Applications of Bio-Inspired Algorithms in Image Segmentation. Electronics 2021, 10, 3116.

73. Zheng, J.; Hu, G.; Ji, X.; Qin, X. Quintic generalized Hermite interpolation curves: Construction and shape optimization using an improved GWO algorithm. Comput. Appl. Math. 2022, 41, 115.

74. Wang, X.; Wang, Y.; Wong, K.-C.; Li, X. A self-adaptive weighted differential evolution approach for large-scale feature selection. Knowl. Based Syst. 2022, 235, 107633.