A Hybrid SSA and SMA with Mutation Opposition-Based Learning for Constrained Engineering Problems

Shuang Wang, Qingxin Liu, Yuxiang Liu, Heming Jia, Laith Abualigah, Rong Zheng, and Di Wu

1School of Information Engineering, Sanming University, Sanming 365004, China
2School of Computer Science and Technology, Hainan University, Haikou 570228, China
3College of Physics and Information Engineering, Fuzhou University, Fuzhou 350108, China
4Faculty of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan
5School of Computer Science, Universiti Sains Malaysia, Gelugor, Penang, Malaysia
6School of Education and Music, Sanming University, Sanming 365004, China

Correspondence should be addressed to Heming Jia; jiaheminglucky99@126.com

Received 25 July 2021; Accepted 19 August 2021; Published 7 September 2021

Abstract

Based on Salp Swarm Algorithm (SSA) and Slime Mould Algorithm (SMA), a novel hybrid optimization algorithm, named Hybrid Slime Mould Salp Swarm Algorithm (HSMSSA), is proposed to solve constrained engineering problems. SSA can obtain good results in solving some optimization problems. However, it is easy to suffer from local minima and lower density of population. SMA specializes in global exploration and good robustness, but its convergence rate is too slow to find satisfactory solutions efficiently. BX_hus, in this paper, considering the characteristics and advantages of both the above optimization algorithms, SMA is integrated into the leader position updating equations of SSA, which can share helpful information so that the proposed algorithm can utilize these two algorithms’ advantages to enhance global optimization performance. Furthermore, Levy flight is utilized to enhance the exploration ability. It is worth noting that a novel strategy called mutation opposition-based learning is proposed to enhance the performance of the hybrid optimization algorithm on premature convergence avoidance, balance between exploration and exploitation phases, and finding satisfactory global optimum. To evaluate the efficiency of the proposed algorithm, HSMSSA is applied to 23 different benchmark functions of the unimodal and multimodal types. Additionally, five classical constrained engineering problems are utilized to evaluate the proposed technique’s practicable abilities. The simulation results show that the HSMSSA method is more competitive and presents more engineering effectiveness for real-world constrained problems than SMA, SSA, and other comparative algorithms. In the end, we also provide some potential areas for future studies such as feature selection and multilevel threshold image segmentation.

1. Introduction

In recent years, metaheuristic algorithms have been widely concerned by a large number of scholars. Compared with other traditional optimization algorithms, the concept of metaheuristic algorithms is simple. Besides, they are flexible and can bypass local optima. Thus, metaheuristics have been successfully applied in different fields to solve various complex optimization problems in the real world [1–3].
algorithm [9], Multiverse Optimizer (MVO) [10], Artificial Chemical Reaction Optimization Algorithm (ACROA) [11], Ray Optimization (RO) [12], Curved Space Optimization (CSO) [13], Sine Cosine Algorithm (SCA) [14], Arithmetic Optimization Algorithm (AOA) [15], and Heat Transfer Relation-based Optimization Algorithm (HTOA) [16]. The third category algorithm is swarm-based techniques, which simulate the social behavior of creatures in nature. Some optimization techniques of this class include Particle Swarm Optimization (PSO) [17] Ant Colony Optimization Algorithm (ACO) [18], Firefly Algorithm (FA) [19], Grey Wolf Optimizer (GWO) [20], Cuckoo Search (CS) Algorithm [21], Whale Optimization Algorithm (WOA) [22], Bald Eagle Search (BES) algorithm [23], and Aquila Optimizer (AO) [24].

It is worth noting that the most widely used swarm-based optimization algorithm is PSO [25]. PSO simulates the behavior of birds flying together in flocks. During the search, they all follow the best solutions in their paths. Cacciola et al. [26] discussed the problem of corrosion profile reconstruction starting from electrical data, in which PSO was utilized to obtain the image of the reconstructed corrosion profile. The result shows that PSO can obtain the optimal solution compared with LSM, and it takes the least time. This allows us to recognize the huge potential of the optimization algorithm.

Salp Swarm Algorithm (SSA) [27] is a swarm-based algorithm proposed in 2017. SSA is inspired by swarm behavior, navigation, and foraging of salps in the ocean. Since SSA has fewer parameters and is easier to be realized in a program than other algorithms, SSA has been applied to many optimization problems, such as feature selection, image segmentation, and constrained engineering problems. However, like other metaheuristic algorithms, SSA may be easy to trap into local minima and lower population density. Therefore, many improved researches have been proposed to enhance the performance of SSA in many fields. Tubishat et al. [28] presented a Dynamic SSA (DSSA), which shows better accuracy than SSA in feature selection. Salgova et al. [29] proposed a self-adaptive SSA to enhance exploitation ability and convergence speed. Negaz et al. [30] proposed an improved leader in SSA using Sine Cosine Algorithm and disrupt operator for feature selection. Jia and Lang [31] presented an enhanced SSA with a crossover scheme and Levy flight to improve the movement patterns of salp leaders and followers. There are also other attempts on the hybrid algorithm of SSA. Saafan and El-gendy [32] proposed a hybrid improved Whale Optimization Salp Swarm Algorithm (IWOSSA). The IWOSSA achieves a better balance between exploration and exploitation phases and avoids premature convergence effectively. Singh et al. [33] developed a hybrid SSA with PSO, which integrated the advantages of SSA and PSO to eliminate trapping in local optima and imbalanced exploitation. Abadi et al. [34] proposed a hybrid approach by combining SSA with GA, which could obtain good results in solving some optimization problems.

Slime Mould Algorithm (SMA) [35] is the latest swarm intelligence algorithm proposed in 2020. This algorithm simulates the oscillation mode and the foraging of Slime Mould in nature. SMA has a unique search mode, which keeps the algorithm from falling into local optima, and has superior global exploration capability. The approach has been applied in real-world optimization problems like feature selection [36], parameters optimization of the fuzzy system [37], multilevel threshold image segmentation [38], control scheme [39], and parallel connected multistacks fuel cells [40].

Therefore, based on the capabilities of both above algorithms, we try to do a hybrid operation to improve the performance of SMA or SSA and then propose a new hybrid optimization algorithm (HSMSSA) to speed up the convergence rate and enhance the overall optimization performance. The specific method is that we integrate SMA as the leader role into SSA and retain the exploitation phase of SSA. At the same time, inspired by the significant performance of opposition-based learning and quasiopposition-based learning, we propose a new strategy named mutation opposition-based learning (MOBL), which switches the algorithm between opposition-based learning and quasiopposition-based learning through a mutation rate to increase the diversity of the population and speed up the convergence rate. In addition, Levy flight is utilized to improve SMA’s exploration capability and balance the exploration and exploitation phases of the algorithm. The proposed HSMSSA algorithm can improve both the exploration and exploitation abilities. The proposed HSMSSA is tested on 23 different benchmark functions and compared with other optimization algorithms. Furthermore, five constrained engineering problems are also utilized to evaluate HSMSSA’s capability on real-world optimization problems. The experimental results illustrate that the HSMSSA possesses the superior capability to search the global minimum and achieve less cost engineering design results than other state-of-the-art metaheuristic algorithms.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of SSA, SMA, Levy flight, and mutation opposition-based learning strategy. Section 3 describes the proposed hybrid algorithm in detail. In Section 4, the details of benchmark functions, parameter settings of the selected algorithms, simulation experiments, and results analysis are introduced. Conclusions and prospects are given in Section 5.

2. Preliminaries

2.1. Salp Swarm Algorithm. In the deep sea, salps live in groups and form a salp chain to move and forage. In the salp chain, there are leaders and followers. The leader moves towards the food and guides the followers. In the process of moving, leaders explore globally, while followers thoroughly search locally [27]. The shapes of a salp and salp chain are shown in Figure 1.

2.1.1. Leader Salps. The front salp of the chain is called the leader, so the following equation is used to perform this action to the salp leader:
where $X^1_j$ and $F_i$ represent the new position of the leader and food source in the $j$th dimension and $r_1$ and $r_2$ are randomly generated numbers in the interval [0, 1]. It is worth noting that $c_1$ is essential for SSA because it balances exploration and exploitation during the search process. $t$ is the current iteration and $T$ is the max iteration.

2.1.2. Follower Salps. To update the position of the followers, the new concept is introduced, which is based on Newton’s law of motion as in the following equation:

$$X^i_j = \frac{1}{2} g t^2 + \omega_0 t, \quad i \geq 2,$$

where $X^i_j$ represents the position of the $i$th follower salp in the $j$th dimension and $g$ and $\omega_0$ indicate the acceleration and the velocity, respectively. The updating process of followers can be expressed as in the following equation:

$$X^i_j = \frac{1}{2} (X^i_j + X^{i-1}_j).$$

The pseudocode of SSA is presented in Algorithm 1.

2.2. Slime Mould Algorithm. The main idea of SMA is inspired by the behavior and morphological changes of Slime Mould in foraging. Slime Mould can dynamically change search mode based on the quality of food sources. If the food source has a high quality, the Slime Mould will use the region-limited search method. If the food concentration is low, the Slime Mould will explore other food sources in the search space. Furthermore, even if Slime Mould has found a high-quality food source, it still divides some individuals to explore another area in the region [35]. The behavior of Slime Mould can be mathematically described as follows:

$$X(t + 1) = \begin{cases} r_4 \times (UB - LB) + LB, & r_5 < z, \\ X_b(t) + \vec{v} \times (W \cdot X_A(t) - X_B(t)), & r_5 < p, \\ \vec{v} \times X(t), & r_5 \geq p, \end{cases}$$

where $X_A(t)$ and $X_B(t)$ represent two individuals selected randomly from the population, and $X(t)$ represents the location of Slime Mould. $\vec{v}$ decreases linearly from one to zero, and $\vec{v}$ is an oscillation parameter in the range $[-a, a]$, in which $a$ is calculated as follows:

$$a = \arctan h \left( -\frac{t}{T} + 1 \right).$$

The coefficient $W$ is a very important parameter, which simulates the oscillation frequency of different food concentrations so that Slime Mould can approach food more quickly when they find high-quality food. The formula of $W$ is listed as follows:
(1) Initialize the population size $N$ and max iteration $T$;
(2) Initialize the positions of salp $X_i$ $(i=1, 2, \ldots, N)$
(3) While $(t \leq T)$
(4) Calculate fitness of each salp;
(5) Denote the best solution as $F$
(6) update $c_1$ by equation (2);  
(7) For $i=1$ to $N$ Do
(8) if $(i=1)$ then
(9) update position of leader salp by equation (1)
(10) Else
(11) update position of follower salp by equation (4)
(12) End if
(13) End for
(14) $t = t + 1$
(15) End While
(16) Return the best solution $F$;

ALGORITHM 1: Pseudocode of Salp Swarm Algorithm.

$$
W(\text{SmellIndex}(i)) = \begin{cases} 
1 + r_6 \times \log\left(\frac{bF - S(i)}{bF - wF} + 1\right), & \text{condition}, \\
1 - r_6 \times \log\left(\frac{bF - S(i)}{bF - wF} + 1\right), & \text{others},
\end{cases}
$$

SmellIndex = sort $(S)$

where $i=1, 2, \ldots, N$ and $S(i)$ represents the fitness of $X$.

condition indicates that $S(i)$ ranks first half of the Slime
Mould, and $r_6$ are random numbers uniformly generated in
the interval of $[0, 1]$. $bF$ represents the optimal fitness
obtained in the current iterative process, $wF$ represents the
worst fitness value obtained in the iterative process cur-
currently, and $\text{SmellIndex}$ denotes the sequence of fitness values
sorted (ascends in the minimum value problem).

The $p$ parameter can be described as follows:

$$p = \tanh[S(i) - DF],$$

where $DF$ represents the best fitness over all iterations.

Figure 2 visualizes the general logic of SMA.

The pseudocode of SMA is presented in Algorithm 2.

2.3. Levy Flight. Levy flight is an effective strategy for meta-
heuristic algorithms, successfully designed in many algorithms
[41–44]. Levy flight is a class of non-Gaussian random processes
that follows Levy distribution. It alternates between short-dis-
tance and occasionally long-distance walking, which can be
inferred from Figure 3. The formula of Levy flight is as follows:

$$
\text{Levy} = 0.01 \times \frac{r_7 \times \sigma}{r_8^{(1/\beta)}},
$$

$$
\sigma = \left(\frac{\Gamma(1 + \beta) \times \sin(\pi\beta/2)}{\Gamma((1 + \beta)/2) \times \beta \times 2^{(\beta-1)/2}}\right)^{(1/\beta)}.
$$

where $r_7$ and $r_8$ are random values in the range of $[0, 1]$ and $\beta$
is a constant equal to 1.5.

2.4. Mutation Opposition-Based Learning.

Opposition-based learning (OBL) was proposed by Tizhoosh
in 2005 [45]. The essence of OBL is selecting the best solution
to the next iteration by comparing the current solution and its
opposition-based learning solution. The OBL strategy has
been successfully used in varieties of metaheuristic algorithms
[46–51] to improve the ability of local optima stagnation
avoidance, and the mathematical expression is as follows:

$$
X_{\text{OBL}}(t + 1) = LB + UB - X(t).
$$

Quasiopposition-based learning (QOBL) [52] is an
improved version from OBL, which applies quasiopposite
points instead of opposite points. These points produced
through QOBL have more likelihood of being unknown
solutions than the points created by OBL. The mathematical
formula of QOBL is as follows:

$$
X_{\text{QOBL}}(t + 1) = \begin{cases} 
CS + r_9 \times (MP - CS), & \text{if } MP > CS, \\
MP + r_9 \times (CS - MP), & \text{otherwise},
\end{cases}
$$

where $CS = LB + UB / 2$ and $MP = LB + UB - X(t)$. 
Considering the superior performance of the two kinds of opposition-based learning, we propose mutation opposition-based learning (MOBL) by combining the mutation rate with these two opposition-based learning. By selecting the mutation rate, we can give full play to the characteristics of the OBL and QOBL and effectively enhance the ability of the algorithm to jump out of the local optima. Figure 4 is an MOBL example, in which Figure 4(a) shows an objective function and Figure 4(b) displays three candidate solutions and their OBL solutions or QOBL solutions. The mathematical formula is as follows:

\[
X(t+1) = \begin{cases} 
X_{OBL}(t), & \text{if } r_{10} < \text{rate}, \\
X_{QOBL}(t), & \text{else,}
\end{cases}
\]  

(13)

where rate is mutation rate, and we set it to 0.1.

3. The Proposed Algorithm

3.1. Details of HSMSSA. In SSA, the population is divided into leader salps and follower salps: leader salps are the first half of salps in the chain, and follower salps follow the leader. However, the leader salp has poor randomness and is easy to fall into local optima. For the SMA algorithm, Slime Mould selects different search modes according to the positive and negative feedback of the current food concentration and has a certain probability of isolating some individuals to explore other regions in search space. These mechanisms increase the randomness of Slime Mould and enhance the ability to explore. The \(vb\) parameter is utilized to realize the oscillation mode of Slime Mould, which is in the range of \([-a, a]\). However, \(vb\) has the drawback of low randomness, which cannot effectively simulate the process of Slime Mould.
looking for food sources. Therefore, we introduce Levy flight into the exploration phase to further enhance the exploration ability. Next, we integrate SMA into SSA, change the position update method of leader salps, and further improve the randomness of the algorithm through Levy flight. For followers, we propose a mutation opposition-based learning to enhance its population diversity and increase the ability of the algorithm to jump out of the local optima. The mathematical formula of leader salps is as follows:

\[
X(t + 1) = \begin{cases} 
    r_4 \times (UB - LB) + LB, & r_5 < z, \\
    X_b + vb \times (W \times X_A - X_B) \times \text{Levy}, & r_5 < p, \\
    wc \times X_i, & r_5 \geq p.
\end{cases}
\]  

(14)

The pseudocode of HSMSSA is given in Algorithm 3, and the summarized flowchart is displayed in Figure 5. As shown in Algorithm 3, the position of the population is initially generated randomly. Then, each individual’s fitness will be calculated. For the entire population in each iteration, parameter \( W \) is calculated using equation (7). The search agents of population size \( N \) are assigned to the two algorithms, which can utilize the advantages of SSA and SMA, and realize the sharing of helpful information to achieve global optimization. If the search agent belongs to the first half of the population, the position will be updated using equation (14) in SMA with Levy flight. Otherwise, the position is determined using equation (4) and MOBL. Finally, if the termination criteria are satisfied, the algorithm returns the best solution found so far; else the previous steps are repeated.

3.2. Computational Complexity Analysis. HSMSSA mainly consists of three components: initialization, fitness evaluation, and position updating. In the initialization phase, the computational complexity of positions generated is \( O(N \times D) \), where \( D \) is the dimension size of the problem. Then, the computational complexity of fitness evaluation for the solution is \( O(N) \) during the iteration process. Finally, we utilize mutation opposition-based learning to keep the algorithm from falling into local optima; thus, the computational complexities of position updating of HSMSSA are \( O(2 \times N \times D) \). Therefore, the total computational complexity of the proposed HSMSSA algorithm is \( O(N \times D + N + 2 \times N \times D) \).

4. Experimental Results and Discussion

This section compared the HSMSSA with some state-of-the-art metaheuristics algorithms on 23 benchmark functions to validate its performance. Moreover, five engineering design problems are employed as examples for real-world applications. The experimentations ran on Windows 10 with 24 GB RAM and Intel (R) i5-9500. All simulations were carried out using MATLAB R2020b.

4.1. Definition of 23 Benchmark Functions. To assess HSMSSA’s ability of exploration, exploitation, and escaping from local optima, 23 benchmark functions, including unimodal and multimodal functions, are tested [27]. The unimodal benchmark functions (F1–F7) are utilized to examine the exploitation ability of HSMSSA. The description of the unimodal benchmark function is shown in Table 1. The multimodal and fixed-dimension multimodal benchmark functions (F8–F23) shown in Tables 2 and 3 are used to test the exploration ability of HSMSSA.

In order to show the experimental results more representative, the HSMSSA is compared with the basic SMA [35]
Figure 4: MOBL example: (a) objective function and (b) candidate solutions (red point), its OBL solution (blue point), and its QOBL solution (green point).

Figure 5: Flowchart of HSMSSA.
and SSA [27], AO [24], AOA [15], WOA [22], SCA [14], and MVO [10]. For all tests, we set the population size $N \geq 30$, dimension size $D = 30$, and maximum iteration $T = 500$, respectively, for all algorithms with 30 independent runs. The parameter settings of each algorithm are shown in Table 4. After all, average results and standard deviations are employed to evaluate the results. Note that the best results will be bolded.

4.1.1. Evaluation of Exploitation Capability (F1–F7). As we can see, unimodal benchmark functions have only one global optimum. These functions are allowed to evaluate the exploitation ability of the metaheuristic algorithms. It can be seen from Table 5 that HSMSSA is very competitive with SMA, SSA, and other metaheuristic algorithms. In particular, HSMSSA can achieve much better results than other metaheuristic algorithms except F6. For F1–F4, HSMSSA can find the theoretical optimum. For all unimodal functions except F5, HSMSSA gets the smallest average values and standard deviations compared to other algorithms, which indicate the best accuracy and stability. Hence, the exploitation capability of the proposed HSMSSA algorithm is excellent.

4.1.2. Evaluation of Exploration Capability (F8–F23). Unlike unimodal functions, multimodal functions have many local optima. Thus, this kind of test problem turns very useful to evaluate the exploration capability of an optimization algorithm. The results shown in Table 5 for functions F8–F23 indicate that HSMSSA also has an excellent exploration capability. In fact, we can see that HSMSSA can find the theoretical optimum in F9, F11, F16–F17, and F19–F23. These results reveal that HSMSSA can also provide superior exploration capability.

4.1.3. Analysis of Convergence Behavior. The convergence curves of some functions are selected and shown in Figure 6, which show the convergence rate of algorithms. It can be seen that HSMSSA shows competitive performance compared to other state-of-the-art algorithms. The HSMSSA presents a faster convergence speed than all other algorithms in F7–F13, F15, and F19–F23. For other benchmark functions, HSMSSA shows a better capability of local optima avoidance than other comparison algorithms in F5 and F6.

4.1.4. Qualitative Results and Analysis. Furthermore, Figure 7 shows the results of several representative test functions on search history, trajectory, average fitness, and

\begin{algorithm}
\begin{algorithmic}
\State (1) Set the initial values of the population size $N$ and the maximum number of iterations $T$
\State (2) Initialize positions of the population $X$
\While{$t \leq T$}
\State (4) Check if the position goes out of the search space boundary and bring it back.
\State (5) Calculate the fitness of each search agent
\State (6) Update $W$, $bF$, and $wF$
\For{$i = 1$ to $N$}
\If{$r_1 < z$}
\State (7) Update position by equation (14)
\Else
\State (10) Update position by equation (14)
\Else
\State (13) Update position by equation (14)
\Else
\State (16) Update position by equation (14)
\Endif
\Endfor
\State (20) Update the position of MOBL using equation (13)
\State (23) Update the position of MOBL using equation (13)
\Endwhile
\State (30) Return $X_{\text{best}}$
\end{algorithmic}
\end{algorithm}

Algorithm 3: Pseudocode of HSMSSA.
### Table 1: Unimodal benchmark functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Dim</th>
<th>Range</th>
<th>$F_{min}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_1(x) = \sum_{i=1}^{n} x_i^2$</td>
<td>30</td>
<td>$[-100, 100]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_2(x) = \sum_{i=1}^{n}</td>
<td>x_i</td>
<td>+ \prod_{i=1}^{n}</td>
<td>x_i</td>
</tr>
<tr>
<td>$F_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$</td>
<td>30</td>
<td>$[-100, 100]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_4(x) = \max(</td>
<td>x_i</td>
<td>, 1 \leq i \leq n)$</td>
<td>30</td>
</tr>
<tr>
<td>$F_5(x) = \sum_{i=1}^{n+1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$</td>
<td>30</td>
<td>$[-30, 30]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_6(x) = \sum_{i=1}^{n} (x_i + 5)^2$</td>
<td>30</td>
<td>$[-100, 100]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_7(x) = \sum_{i=1}^{n} i x_i^2 + \text{random}(0, 1)$</td>
<td>30</td>
<td>$[-1.28, 1.28]$</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 2: Multimodal benchmark functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Dim</th>
<th>Range</th>
<th>$F_{min}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_6(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{</td>
<td>x_i</td>
<td>})$</td>
<td>30</td>
</tr>
<tr>
<td>$F_7(x) = \sum_{i=1}^{n} x_i - 10 \cos(2\pi x_i) + 10$</td>
<td>30</td>
<td>$[-512, 512]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_8(x) = -20 \exp(-0.2 \sqrt{(1/n) \sum_{i=1}^{n} x_i^2}) - \exp((-1/n) \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + \epsilon$</td>
<td>30</td>
<td>$[-32, 32]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_9(x) = \frac{1}{(4000)} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$</td>
<td>30</td>
<td>$[-200, 200]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_{10}(x) = (\pi/n) \left{ 10 \sin(\pi y_i) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right}$</td>
<td>30</td>
<td>$[-50, 50]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_{11}(x) = \frac{1}{n} \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, where $y_i = 1 + (x_i + 1)/4$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, &amp; x_i &gt; a, \ 0, &amp; -a &lt; x_i &lt; a, \ k(-x_i - a)^m, &amp; x_i &lt; -a \end{cases}$</td>
<td>30</td>
<td>$[-50, 50]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_{12}(x) = (3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)]$</td>
<td>30</td>
<td>$[-50, 50]$</td>
<td>0</td>
</tr>
<tr>
<td>$F_{13}(x) = \frac{1}{2} [1 + \sin^2(2\pi x_1)] + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$</td>
<td>30</td>
<td>$[-50, 50]$</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 3: Fixed-dimension multimodal benchmark functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Dim</th>
<th>Range</th>
<th>$F_{min}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_{14}(x) = \left{ \left(1 + \sum_{i=1}^{n} x_i \right) / \left(1 + \sum_{i=1}^{n} x_i \right) \right}^{1/2}$</td>
<td>2</td>
<td>$[-65, 65]$</td>
<td>0.998</td>
</tr>
<tr>
<td>$F_{15}(x) = \sum_{i=1}^{n} \left[ \left( b_i^2 + b_i x_i + b_i x_i + x_i \right) / \left( b_i^2 + b_i x_i + x_i \right) \right]^2$</td>
<td>4</td>
<td>$[-5, 5]$</td>
<td>0.00030</td>
</tr>
<tr>
<td>$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + 1/3x_1^6 + x_1x_2 - 4x_2^2 + x_2^4$</td>
<td>2</td>
<td>$[-5, 5]$</td>
<td>1.0316</td>
</tr>
<tr>
<td>$F_{17}(x) = \left( x_1 - \left( x_1 - 5.14/\pi^2 \right) x_1^2 + \left( 5 \pi x_1 - 6 \right)^2 + 10 \right) \cos \left( x_1 + 10 \right)$</td>
<td>2</td>
<td>$[-5, 5]$</td>
<td>0.398</td>
</tr>
<tr>
<td>$F_{18}(x) = \left[ 1 + \left( x_1 + x_2 + 1 \right)^2 \left( 19 - 14 x_1 + 3 x_2^2 - 14 x_1 + 6 x_1 x_2 + 3 x_2^2 \right) \right] \times \left[ 30 + \left( 2 x_1 - 3 x_2 \right)^2 + \left( 18 - 32 x_2 + 12 x_2^2 + 48 x_1 - 36 x_1 x_2 + 27 x_2^2 \right) \right]$</td>
<td>2</td>
<td>$[-2, 2]$</td>
<td>3</td>
</tr>
<tr>
<td>$F_{19}(x) = \left[ \left( \sum_{i=1}^{n} x_i \right) / \left( \sum_{i=1}^{n} x_i \right) \right] \exp\left( \sum_{i=1}^{n} x_i \right)$</td>
<td>3</td>
<td>$[-1, 2]$</td>
<td>3.86</td>
</tr>
<tr>
<td>$F_{20}(x) = \left[ \left( \sum_{i=1}^{n} x_i \right) / \left( \sum_{i=1}^{n} x_i \right) \right] \exp\left( \sum_{i=1}^{n} x_i \right)$</td>
<td>6</td>
<td>$[0, 1]$</td>
<td>3.32</td>
</tr>
<tr>
<td>$F_{21}(x) = \left[ \left( \sum_{i=1}^{n} x_i \right) / \left( \sum_{i=1}^{n} x_i \right) \right] \exp\left( \sum_{i=1}^{n} x_i \right)$</td>
<td>4</td>
<td>$[0, 10]$</td>
<td>10.1532</td>
</tr>
<tr>
<td>$F_{22}(x) = \left[ \left( \sum_{i=1}^{n} x_i \right) / \left( \sum_{i=1}^{n} x_i \right) \right] \exp\left( \sum_{i=1}^{n} x_i \right)$</td>
<td>4</td>
<td>$[0, 10]$</td>
<td>10.4028</td>
</tr>
<tr>
<td>$F_{23}(x) = \left[ \left( \sum_{i=1}^{n} x_i \right) / \left( \sum_{i=1}^{n} x_i \right) \right] \exp\left( \sum_{i=1}^{n} x_i \right)$</td>
<td>4</td>
<td>$[0, 10]$</td>
<td>10.5363</td>
</tr>
</tbody>
</table>
convergence curve. From search history maps, we can see the search agent’s distribution of the HSMSSA while exploring and exploiting the search space. Because of the fast convergence, the vast majority of search agents are concentrated near the global optimum. Inspecting trajectory figure in Figure 5, the first search agent constantly oscillates in the first dimension of the search space, which suggests that the search agent investigates the most promising areas and better solutions widely. This powerful search capability is likely to come from the Levy flight and MOBL strategies. The average fitness presents if exploration and exploitation are conducive to improve the first random population, and an accurate approximation of the global optimum can be found in the end.

Similarly, it can be noticed that the average fitness oscillates in the early iterations and then decreases abruptly and begins to level off. The average fitness maps also show the significant improvement of the first random population and the final global optimal, accurate approximation acquisition. At last, the convergence curves reveal the best fitness value found by search agents after each iteration. By observing this, the HSMSSA shows breakneck convergence speed.

4.1.5. Wilcoxon Signed-Rank Test. Because the algorithm results are random, we need to carry out statistical tests to prove that the results have statistical significance. We use Wilcoxon signed-ranks (WSR) test results to evaluate the statistical significance of the two algorithms at 5% significance level [53]. The WSR is a statistical test that is applied to two different results for searching the significantly different. As is well-known, a p-value less than 0.05 indicates that it is significantly superior to other algorithms. Otherwise, the obtained results are not statistically significant. The calculated results of the Wilcoxon signed-rank test between HSMSSA and other algorithms for each benchmark function are listed in Table 6. HSMSSA outperforms all other algorithms in varying degrees. This superiority is statistically significant on unimodal functions F2 and F4–F7, which indicates that HSMSSA possesses high exploitation. HSMSSA also shows better results on multimodal function F8–F23, suggesting that HSMSSA has a high capability of exploration. To sum up, HSMSSA can provide better results for almost all benchmark functions than other comparative algorithms.

### Table 4: Parameter settings for the comparative algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMA [35]</td>
<td>z = 0.03</td>
</tr>
<tr>
<td>SSA [27]</td>
<td>c₁ = [1, 0]; c₂ ∈ [0, 1]; c₃ ∈ [0, 1]</td>
</tr>
<tr>
<td>AO [24]</td>
<td>U = 0.00565; r₁ = 10; ω = 0.005; α = 0.1; δ = 0.1; G₁ ∈ [-1, 1]; G₂ = [2, 0]</td>
</tr>
<tr>
<td>AOA [15]</td>
<td>a = 5; μ = 0.5;</td>
</tr>
<tr>
<td>WOA [22]</td>
<td>a₁ = [2, 0]; a₂ = [-1, -2]; b = 1</td>
</tr>
<tr>
<td>SCA [14]</td>
<td>a = [2, 0]</td>
</tr>
<tr>
<td>MVO [10]</td>
<td>WEP ∈ [0.2, 1]; TDR ∈ [0, 1]; T₁, T₂, T₃ ∈ [0, 1]</td>
</tr>
</tbody>
</table>

4.2. Experiments on Engineering Design Problems. In this section, HSMSSA is evaluated to solve five classical engineering design problems: pressure vessel design problem, tension spring design problem, three-bar truss design problem, speed reducer problem, and cantilever beam design. To address these problems, we set the population size N = 30 and maximum iteration T = 500. The results of HSMSSA are compared to various state-of-the-art algorithms in the literature. The parameter settings are the same as previous numerical experiments.

#### 4.2.1. Pressure Vessel Design Problem. The pressure vessel design problem [53] is to minimize the total cost of cylindrical pressure vessel to match pressure requirements and form the pressure vessel shown in Figure 8. Four parameters in this problem need to be minimized, including the thickness of the shell (Ts), the thickness of head (Th), inner radius (R), and the length of the cylindrical section without the head (L), as shown in Figure 8. The constraints and equation are as follows.

Consider

\[
\overline{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L].
\]

Minimize

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

subject to

\[
\begin{align*}
g_1(\overline{x}) &= -x_1 + 0.0193x_3 \leq 0, \\
g_2(\overline{x}) &= -x_3 + 0.00954x_3 \leq 0, \\
g_3(\overline{x}) &= \pi x_1^2x_4 - \frac{4}{3} \pi x_3^3 + 129600 \leq 0, \\
g_4(\overline{x}) &= x_4 - 240 \leq 0.
\end{align*}
\]

Variable range is

\[
\begin{align*}
0 \leq x_1 &\leq 99, \\
0 \leq x_2 &\leq 99, \\
10 \leq x_3 &\leq 200, \\
10 \leq x_4 &\leq 200.
\end{align*}
\]
<table>
<thead>
<tr>
<th>Function</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
<th>F7</th>
<th>F8</th>
<th>F9</th>
<th>F10</th>
<th>F11</th>
<th>F12</th>
<th>F13</th>
<th>F14</th>
<th>F15</th>
<th>F16</th>
<th>F17</th>
<th>F18</th>
<th>F19</th>
<th>F20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.0000E+00</td>
<td>1.4800E-07</td>
<td>1.2386E-03</td>
<td>1.2570E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td></td>
</tr>
<tr>
<td>Std</td>
<td>0.0000E+00</td>
<td>1.4600E-07</td>
<td>1.2386E-03</td>
<td>1.2570E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td>0.0000E+00</td>
<td>8.5380E-15</td>
<td>3.5900E-03</td>
<td>3.8630E-03</td>
<td>3.8970E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td>3.9020E-03</td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>0.0000E+00</td>
<td>1.4800E-07</td>
<td>1.2386E-03</td>
<td>1.2570E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td>1.2750E-03</td>
<td></td>
</tr>
</tbody>
</table>

Note: The table represents the results of algorithms on 23 benchmark functions. The values reported are the mean and standard deviation (Std) for each function.
Table 5: Continued.

<table>
<thead>
<tr>
<th>Function</th>
<th>HSMSSA</th>
<th>SMA</th>
<th>SSA</th>
<th>AO</th>
<th>AOA</th>
<th>WOA</th>
<th>SCA</th>
<th>MVO</th>
</tr>
</thead>
<tbody>
<tr>
<td>F21 Mean</td>
<td><strong>-1.0153E+01</strong></td>
<td>-1.0153E+01</td>
<td>-6.3902E+00</td>
<td>-1.0142E+01</td>
<td>-7.4694E+00</td>
<td>-8.1869E+00</td>
<td>-2.4287E+00</td>
<td>-7.0425E+00</td>
</tr>
<tr>
<td>Std</td>
<td><strong>1.0300E-07</strong></td>
<td>1.9553E-04</td>
<td>3.5233E+00</td>
<td>2.6961E-02</td>
<td>3.0153E+00</td>
<td>2.6299E+00</td>
<td>2.0334E+00</td>
<td>3.0750E+00</td>
</tr>
<tr>
<td>F22 Mean</td>
<td><strong>-1.0403E+01</strong></td>
<td>-1.0403E+01</td>
<td>-7.8362E+00</td>
<td>-1.0392E+01</td>
<td>-7.3418E+00</td>
<td>-7.6682E+00</td>
<td>-3.4206E+00</td>
<td>-8.4065E+00</td>
</tr>
<tr>
<td>Std</td>
<td><strong>1.6600E-07</strong></td>
<td>3.1185E-04</td>
<td>3.4923E+00</td>
<td>1.8669E-02</td>
<td>3.2433E+00</td>
<td>3.4333E+00</td>
<td>1.7953E+00</td>
<td>2.9406E+00</td>
</tr>
<tr>
<td>F23 Mean</td>
<td><strong>-1.0536E+01</strong></td>
<td>-1.0536E+01</td>
<td>-9.4298E+00</td>
<td>-1.0525E+01</td>
<td>-7.9685E+00</td>
<td>-7.2480E+00</td>
<td>-3.6694E+00</td>
<td>-9.6618E+00</td>
</tr>
<tr>
<td>Std</td>
<td><strong>9.5800E-08</strong></td>
<td>2.5920E-04</td>
<td>2.5607E+00</td>
<td>1.6743E-02</td>
<td>3.5174E+00</td>
<td>3.2297E+00</td>
<td>1.8159E+00</td>
<td>2.3084E+00</td>
</tr>
</tbody>
</table>
Figure 6: Convergence curves of 23 benchmark functions.
Figure 7: Parameter space, search history, trajectory, average fitness, and convergence curves of HSMSSA.
obtain superior optimal values compared with SMA, SSA, BX he mathematical of this problem can be written as follows.

Consider

\[ \overrightarrow{x} = [x_1 \ x_2 \ x_3 \ x_4] = [d \ D \ N]. \]  

Minimize

\[ f(\overrightarrow{x}) = (x_3 + 2)x_1x_1^2, \quad (20) \]

subject to

\[ g_1(\overrightarrow{x}) = 1 - \frac{x_3^3}{71785x_1^2} \leq 0, \]

\[ g_2(\overrightarrow{x}) = \frac{4x_3^3 - x_1x_2}{12566(x_1x_1^4 - x_1^4)} + \frac{1}{5108x_1^2} \leq 0, \]

\[ g_3(\overrightarrow{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0, \]

\[ g_4(\overrightarrow{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0. \]

From the results in Table 7, we can see that HSMSSA can obtain superior optimal values compared with SMA, SSA, AO, AOA, WOA, SCA, and MVO.

### 4.2.2. Tension Spring Design Problem

This problem [27] tries to minimize the weight of the tension spring, and there are three parameters that need to be minimized, including the wire diameter (d), mean coil diameter (D), and the number of active coils (N). Figure 9 shows the structure of the tension spring. The mathematical of this problem can be written as follows.

Consider

\[ \overrightarrow{x} = [x_1 \ x_2 \ x_3 \ x_4] = [d \ D \ N]. \]  

Minimize

\[ f(\overrightarrow{x}) = (x_3 + 2)x_1x_1^2, \quad (20) \]

subject to

\[ g_1(\overrightarrow{x}) = 1 - \frac{x_3^3}{71785x_1^2} \leq 0, \]

\[ g_2(\overrightarrow{x}) = \frac{4x_3^3 - x_1x_2}{12566(x_1x_1^4 - x_1^4)} + \frac{1}{5108x_1^2} \leq 0, \]

\[ g_3(\overrightarrow{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0, \]

\[ g_4(\overrightarrow{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0. \]
Variable range is
\[0.05 \leq x_1 \leq 2.00,\]
\[0.25 \leq x_2 \leq 1.30,\]
\[2.00 \leq x_3 \leq 15.00.\]

Results of HSMSSA for solving the three-bar design problem are listed in Table 9, which are compared with SMA, SSA, AO, AOA, WOA, SCA, and MVO. It can be observed that HSMSSA has an excellent ability to solve the problem in confined space.

### 4.2.3. Three-Bar Truss Design Problem

Three-bar truss design is a complex problem in the field of civil engineering [49]. The goal of this problem is to achieve the minimum weight in truss design. Figure 10 shows the design of this problem. The formula of this problem can be described as follows.

Consider
\[
\vec{x} = [x_1, x_2] = [A_1, A_2].
\]

Minimize
\[
f(\vec{x}) = (2\sqrt{2}x_1 + x_2) \times l,
\]

subject to
\[
g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0,
\]
\[
g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0,
\]
\[
g_3(\vec{x}) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0.
\]

Variable range is
\[0 \leq x_1,\]
\[x_2 \leq 1,\]

where \(l = 100 \text{ cm}, P = 2 \text{ KN/cm}^2\), and \(\sigma = 2 \text{ KN/cm}^2\).

Results of HSMSSA for solving the three-bar design problem are listed in Table 9, which are compared with SMA, SSA, AO, AOA, WOA, SCA, and MVO. It can be observed that HSMSSA has an excellent ability to solve the problem in confined space.

### 4.2.4. Speed Reducer Problem

In this problem [15], the total weight of the reducer is minimized by optimizing seven variables. Figure 11 shows the design of this problem, and the mathematical formula is as follows.

Consider
\[
\vec{y} = [y_1, y_2, y_3, y_4, y_5, y_6, y_7] = [D, d, A_1, A_2, a, b, c].
\]

Minimize
\[
f(\vec{y}) = D + d + A_1 + A_2 + a + b + c,
\]

subject to
\[
g_1(\vec{y}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0,
\]
\[
g_2(\vec{y}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0,
\]
\[
g_3(\vec{y}) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0.
\]

Variable range is
\[0 \leq y_1,\]
\[y_2 \leq 1,\]

where \(l = 100 \text{ cm}, P = 2 \text{ KN/cm}^2\), and \(\sigma = 2 \text{ KN/cm}^2\).

Results of HSMSSA for solving the speed reducer problem are listed in Table 10, which are compared with SMA, SSA, AO, AOA, WOA, SCA, and MVO. It can be observed that HSMSSA has an excellent ability to solve the problem in confined space.
### Table 8: Comparison of HSMSSA results with other competitors for the tension spring design problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Optimum variables</th>
<th>Optimum weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSMSSA</td>
<td>0.05, 0.348099, 10.6486</td>
<td>0.011007</td>
</tr>
<tr>
<td>SMA</td>
<td>0.057372, 0.57232, 4.1494</td>
<td>0.011584</td>
</tr>
<tr>
<td>SSA</td>
<td>0.05936, 0.63542, 3.4741</td>
<td>0.012256</td>
</tr>
<tr>
<td>AO</td>
<td>0.056912, 0.55833, 4.3271</td>
<td>0.011442</td>
</tr>
<tr>
<td>AOA</td>
<td>0.052579, 0.41008, 8.9843</td>
<td>0.012453</td>
</tr>
<tr>
<td>WOA</td>
<td>0.055292, 0.47277, 6.929</td>
<td>0.012906</td>
</tr>
<tr>
<td>SCA</td>
<td>0.0588, 0.6172, 3.65</td>
<td>0.012057</td>
</tr>
<tr>
<td>MVO</td>
<td>0.05251, 0.37602, 10.3351</td>
<td>0.012790</td>
</tr>
</tbody>
</table>

**Figure 10:** Three-bar truss design problem.

### Table 9: Comparison of HSMSSA results with other competitors for the three-bar truss design problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Optimum variables</th>
<th>Optimum cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSMSSA</td>
<td>0.78842, 0.40811</td>
<td>263.8523</td>
</tr>
<tr>
<td>SMA</td>
<td>0.79109, 0.40022</td>
<td>263.8668</td>
</tr>
<tr>
<td>SSA</td>
<td>0.77823, 0.43736</td>
<td>263.9363</td>
</tr>
<tr>
<td>AO</td>
<td>0.77899, 0.43541</td>
<td>263.9197</td>
</tr>
<tr>
<td>AOA</td>
<td>0.76342, 0.48382</td>
<td>264.3549</td>
</tr>
<tr>
<td>WOA</td>
<td>0.78357, 0.42252</td>
<td>263.886</td>
</tr>
<tr>
<td>SCA</td>
<td>0.78058, 0.43766</td>
<td>264.5463</td>
</tr>
<tr>
<td>MVO</td>
<td>0.75492, 0.51216</td>
<td>264.7851</td>
</tr>
</tbody>
</table>

**Figure 11:** Speed reducer problem.
Table 10: Comparison of HSMSSA results with other competitors for the speed reducer design problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Optimum variables</th>
<th>Optimum weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSMSSA</td>
<td>3.4976 0.7 17 7.3 7.8 3.35006 5.28553</td>
<td>2995.4374</td>
</tr>
<tr>
<td>SMA</td>
<td>3.51443 0.7 17 7.32444 7.80527 3.35235 5.28494</td>
<td>3002.5941</td>
</tr>
<tr>
<td>SSA</td>
<td>3.49767 0.7 17 7.87797 8.09401 3.3943 5.28574</td>
<td>3018.644</td>
</tr>
<tr>
<td>AO</td>
<td>3.5138 0.7 17 7.41461 7.81291 3.37709 5.28457</td>
<td>3009.9097</td>
</tr>
<tr>
<td>AOA</td>
<td>3.55989 0.7 17 7.49997 8.3 3.4623 5.28512</td>
<td>3061.8731</td>
</tr>
<tr>
<td>WOA</td>
<td>3.49739 0.7 17 7.87039 8.0769 3.37932 5.2799</td>
<td>3034.0596</td>
</tr>
<tr>
<td>SCA</td>
<td>3.6 0.7 17 7.3 8.3 3.37709 5.2799</td>
<td>3063.9102</td>
</tr>
<tr>
<td>MVO</td>
<td>3.6 0.7 17 8.3 8.3 3.38276 5.36041</td>
<td>3111.6609</td>
</tr>
</tbody>
</table>

\[
f(\mathbf{x}) = 0.7854x_1 x_2^2 (3.3333x_3^3 + 14.9334x_3 - 43.0934) - 1.508x_1 (x_2^2 + x_2^2) + 7.4777(x_3^2 + x_3^2),
\]

subject to

\[
\begin{align*}
g_1(\mathbf{x}) &= \frac{27}{x_1 x_2^2 x_3} - 1 \leq 0, \\
g_2(\mathbf{x}) &= \frac{397.5}{x_1 x_2^2 x_3^3} - 1 \leq 0, \\
g_3(\mathbf{x}) &= \frac{1.93x_1}{x_2 x_3^4} - 1 \leq 0, \\
g_4(\mathbf{x}) &= \frac{1.93x_1^3}{x_2 x_3^4} - 1 \leq 0, \\
g_5(\mathbf{x}) &= \frac{\sqrt{(745x_1/x_2 x_3)^2 + 16.9 \times 10^6}}{110.0x_6^4} - 1 \leq 0, \\
g_6(\mathbf{x}) &= \frac{\sqrt{(745x_1/x_2 x_3)^2 + 157.5 \times 10^6}}{85.0x_6^4} - 1 \leq 0, \\
g_7(\mathbf{x}) &= \frac{x_2 x_3}{40} - 1 \leq 0, \\
g_8(\mathbf{x}) &= \frac{5x_2}{x_1} - 1 \leq 0, \\
g_9(\mathbf{x}) &= \frac{x_1}{12x_2} - 1 \leq 0, \\
g_{10}(\mathbf{x}) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0, \\
g_{11}(\mathbf{x}) &= \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0.
\end{align*}
\]

The comparison results are listed in Table 10, which shows the advantage of HSMSSA in realizing the minimum total weight of the problem.

4.2.5. Cantilever Beam Design. Cantilever beam design is a type of concrete engineering problem. This problem aims to determine the minimal total weight of the cantilever beam by optimizing the hollow square cross-section parameters [24]. Figure 12 illustrates the design of this problem, and the mathematical described is as follows.

Consider

\[
x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5].
\]

Minimize

\[
f(\mathbf{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5),
\]

subject to

\[
g(\mathbf{x}) = \frac{60}{x_1} + \frac{27}{x_2} + \frac{19}{x_3} + \frac{7}{x_4} + \frac{1}{x_5} - 1 \leq 0.
\]

Variable range is as follows: 0.01 ≤ x_1, x_2, x_3, x_4, x_5 ≤ 100.

The results are shown in Table 11. From this table, we can see that the performance of HSMSSA is better than all other algorithms and the obtained total weight is minimized.

As a summary, this section demonstrates the superiority of the proposed HSMSSA algorithm in different
characteristics and real case studies. HSMSSA is able to outperform the basic SMA and SSA and other well-known algorithms with very competitive results, which are derived from the robust exploration and exploitation capabilities of HSMSSA. Excellent performance in solving industrial engineering design problems indicates that HSMSSA can be widely used in real-world optimization problems.

5. Conclusion

In this paper, a Hybrid Slime Mould Salp Swarm Algorithm (HSMSSA) is proposed by combining the whole SMA as leaders and the exploitation phase of SSA as followers. At the same time, two strategies, including Levy flight and mutation opposition-based learning, are incorporated to enhance the capabilities of exploration and exploitation of HSMSSA. The 23 standard benchmark functions are utilized to evaluate this algorithm for analyzing its exploration, exploitation, and local optima avoidance capabilities. The experimental results show competitive advantages compared to other state-of-the-art metaheuristic algorithms, proving that HSMSSA has better performance than others. Five engineering design problems are solved as well to verify the superiority of the algorithm further, and the results are also very competitive with other metaheuristic algorithms.

The proposed HSMSSA can produce very effective results for complex benchmark functions and constrained engineering problems. In the future, HSMSSA can be applied to real-world optimization problems such as multi-objective problems, feature selection, multithresholding image segmentation, convolution neural network, or any problem that belongs to NP-complete or NP-hard problems.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

All authors declare that there are no conflicts of interest.

Acknowledgments

This research was funded by the Sanming University Introduces High-Level Talents to Start Scientific Research Funding Support Project (20YG01 and 20YG14), the Guiding Science and Technology Projects in Sanming City (2020-S-39, 2020-G-61, and 2021-S-8), the Educational Research Projects of Young and Middle-Aged Teachers in Fujian Province (JAT200638 and JAT200618), the Scientific Research and Development Fund of Sanming University (B202029 and B202009), the Open Research Fund of Key Laboratory of Agricultural Internet of Things in Fujian Province (ZD2101), the Ministry of Education Cooperative Education Project (20202064014), the School Level Education and Teaching Reform Project of Sanming University (J2010306 and J2010305), and the Higher Education Research Project of Sanming University (SHE2102 and SHE2013).

References


<table>
<thead>
<tr>
<th>Table 11: Comparison of HSMSSA results with other competitors for the cantilever beam design problem.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>HSMSSA</td>
</tr>
<tr>
<td>SMA</td>
</tr>
<tr>
<td>SSA</td>
</tr>
<tr>
<td>AO</td>
</tr>
<tr>
<td>AOA</td>
</tr>
<tr>
<td>WOA</td>
</tr>
<tr>
<td>SCA</td>
</tr>
<tr>
<td>MVO</td>
</tr>
</tbody>
</table>


