Article

A Hybrid Shuffled Frog Leaping Algorithm and Its Performance Assessment in Multi-Dimensional Symmetric Function

Fei Li ^{1,2,3} , Wentai Guo ¹, Xiaotong Deng ¹, Jiamei Wang ¹, Liangquan Ge ^{1,3,*} and Xiaotong Guan ³

¹ College of Nuclear Technology and Automation Engineering, Chengdu University of Technology, Chengdu 610059, China; lifeimvp@sina.com (F.L.); gwt@cdut.edu.cn (W.G.); dengxiaot@cdut.edu.cn (X.D.); wangjiamei@cdut.edu.cn (J.W.)

² School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China

³ Applied Nuclear Technology in Geosciences Key Laboratory of Sichuan Province, Chengdu 610059, China; guanxt@uestc.edu.cn

* Correspondence: glq@cdut.edu.cn

Abstract: Ensemble learning of swarm intelligence evolutionary algorithm of artificial neural network (ANN) is one of the core research directions in the field of artificial intelligence (AI). As a representative member of swarm intelligence evolutionary algorithm, shuffled frog leaping algorithm (SFLA) has the advantages of simple structure, easy implementation, short operation time, and strong global optimization ability. However, SFLA is susceptible to fall into local optima in the face of complex and multi-dimensional symmetric function optimization, which leads to the decline of convergence accuracy. This paper proposes an improved shuffled frog leaping algorithm of threshold oscillation based on simulated annealing (SA-TO-SFLA). In this algorithm, the threshold oscillation strategy and simulated annealing strategy are introduced into the SFLA, which makes the local search behavior more diversified and the ability to escape from the local optima stronger. By using multi-dimensional symmetric function such as drop-wave function, Schaffer function N.2, Rastrigin function, and Griewank function, two groups (i: SFLA, SA-SFLA, TO-SFLA, and SA-TO-SFLA; ii: SFLA, ISFLA, MSFLA, DSFLA, and SA-TO-SFLA) of comparative experiments are designed to analyze the convergence accuracy and convergence time. The results show that the threshold oscillation strategy has strong robustness. Moreover, compared with SFLA, the convergence accuracy of SA-TO-SFLA algorithm is significantly improved, and the median of convergence time is greatly reduced as a whole. The convergence accuracy of SFLA algorithm on these four test functions are 90%, 100%, 78%, and 92.5%, respectively, and the median of convergence time is 63.67 s, 59.71 s, 12.93 s, and 8.74 s, respectively; The convergence accuracy of SA-TO-SFLA algorithm on these four test functions is 99%, 100%, 100%, and 97.5%, respectively, and the median of convergence time is 48.64 s, 32.07 s, 24.06 s, and 3.04 s, respectively.

Keywords: artificial neural network (ANN); swarm intelligence evolutionary algorithm; shuffled frog leaping algorithm (SFLA); multi-dimensional symmetric function

Citation: Li, F.; Guo, W.; Deng, X.; Wang, J.; Ge, L.; Guan, X. A Hybrid Shuffled Frog Leaping Algorithm and Its Performance Assessment in Multi-Dimensional Symmetric Function. *Symmetry* **2022**, *14*, 131. <https://doi.org/10.3390/sym14010131>

Academic Editor: Aviv Gibali

Received: 8 December 2021

Accepted: 5 January 2022

Published: 11 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Artificial neural network (ANN) originated in the 1940s. It is a kind of mathematical model that imitates the structure and function of biological neural network [1–3]. It proves the logical function of a single neuron through mathematical description and initiates the era of ANN research [4–6]. It has high commercial value and brings a lot of job opportunities [7–9]. It is essentially an exploration of the relationship between large amounts of data to capture the subtle characteristics of them [10–12]. In 1958, Rosenblatt proposed the perceptron, which can be regarded as a feedforward ANN in a simple form [13]. ANN has been developing in various fields (face recognition, voice recognition, vehicle tracking, image super-resolution construction, etc.) with the rapid development of hardware, powerful computational efficiency, and strong learning ability [14,15]. The neuro fuzzy

system [16], which is composed of type-2 fuzzy system [17,18] and ANN, is also an important research field. It can not only well express the reasoning function of human brain, but also has an effective learning mechanism. In recent years ANN's achievement were clearly demonstrated, from AlphaGo, a deep reinforcement learning model in human knowledge guidance [19] defeating the game of go world champion Lee Sedol, and its upgraded version AlphaGo Zero, an unsupervised reinforcement learning model without human knowledge [20], defeating AlphaGo with a record of 100:0 to GPT3 [21], a powerful unsupervised natural language processing model, which writes novels, scripts, and codes and caused a great sensation all over the world. Obviously, due to the continuous breakthroughs in unsupervised learning, the field of ANN is developing more rapidly and has a broader application prospect [22,23].

In the process of ANN model training, the optimization algorithm is a vital issue. Excellent optimization algorithm is helpful for the effective adjustment of network weight, shortening the training time and avoiding falling into the local optima [24–26]. As an effective optimization algorithm, swarm intelligence evolutionary algorithm (SIEA), has an outstanding performance in many fields [27–29]. The optimization algorithm of ANN adopts SIEA or takes ANN as an individual for integrated learning, which is also a major research hotspot in the field of artificial intelligence. Shuffled frog leaping algorithm (SFLA), as one of the representative SIEA, was proposed by Eusuff in 2003 [30,31]. SFLA integrates memetic algorithm (MA) with particle swarm optimization (PSO) to achieve the advantages of simple structure, easy implementation, short operation time, and strong global optimization ability. Since SFLA was proposed, it has made remarkable achievements in water resource network optimization, UAV route planning, feeder allocation optimization, limited buffer flow dispatching, power grid distribution, etc.

However, in the later stage of evolution, SFLA may cause problems due to high convergence time, susceptibility to fall into the local optima, and low solution accuracy. Furthermore, the more complex the solution function, the faster the performance degrades. Aiming at the problem of high convergence time, Xuncui Zhang et al., proposed an improved SFLA (ISFLA) in 2008, which introduced the “individual cognitive ability” of PSO algorithm into the SFLA to upgrade frog jumping rules and individual cognitive behavior. The frog is endowed with memory and individual cognitive ability to decrease the convergence time and accuracy of the algorithm. Six representative test functions are adopted to test; the result shows that the accuracy of the improved SFLA is over 93% except for one of the test functions, $f(x_i) = \sum_{i=1}^{N-1} [100 * (x_{i+1} - x_i^2)]$, but the algorithm has many iterative rounds and the iteration time is not given in the experiment, nor is the convergence time [32]. In order to improve the diversity and randomness of the algorithm, Wang QS et al., proposed modified SFLA (MSFLA) in 2011. The addition of the algorithm is to improve the range of the acceleration factor, which is set as a random number between 0 and 2 instead of between 0 and 1 of the SFLA. The randomness of the scanning process is increased. The precision of 10–24 can be achieved by 100 iterations on Rosenbrock function [33]. In 2014, the combined algorithm of Simulated Annealing (SA) [34] and SFLA was adopted by Du. J., which enables the frog population to accept more inferior quality states in the high-energy state, and guides the frog population to jump out of the local optima, thus optimizing the convergence time and convergence accuracy of the algorithm to a certain extent [35]. In the same year, Jaballah et al., proposed a two-factor accelerated optimization strategy which updates the worst and best frogs in the subpopulation at the same time during the local search, which again decreases the convergence time of the algorithm [36]. Additionally, Ahandani MA et al., proposed diversified SFLA (DSFLA), which introduced a new differential operator in the evolution process of SFLA, increasing the diversity of population. Additionally, in Enzyme Effusion Problem (EEP), this algorithm can find the global optima within 186.7 s [37]. Wang H B et al., developed the evolution rules of SFLA in the process of local optimization, which further decreases the convergence time and improves convergence accuracy of the algorithm [38]. Although the convergence accuracy

and convergence time of the algorithm are improved by different methods above, the convergence accuracy of the algorithm is not acceptable on more complex functions.

Therefore, this paper proposes a SFLA of threshold oscillation based on SA(SA-TO-SFLA). The core of this algorithm is to combine the threshold oscillation optimization strategy and the multi-neighborhood simulated annealing optimization strategy in order to optimize SFLA. On the basis of guaranteeing the characteristics of SFLA based on SA, during the later stage of frog population convergence, the solutions of some frog subpopulation oscillate within a certain range, so as to make individual variation in the subpopulation and expand the search range of the subpopulation. When better individuals are produced, information sharing among subpopulations is carried out to increase individual diversity of frog population, and the optimal subpopulation leads the others to escape from the local optimas. Finally, experiments show the robustness and effectiveness of the algorithm in terms of accuracy and convergence time.

2. Principles of SFLA

As a new heuristic SIEA, SFLA simulates the subpopulation coevolution process of a group of frogs when they are searching for the most food locations. It combines deterministic method and randomness method and has more efficient computing power and global search performance.

2.1. Origin and Concept

Imagine a frog population in a wetland. Many stones are scattered in the wetland for frogs to inhabit, and there is a certain amount of food around each stone. The frog hops to find the stones with more food around by looking for different stones. Each frog has its own unique culture. Each frog's culture is defined as a potential solution to the problem, and each frog can exchange information through cultural communication. The frog population of the whole wetland is divided into different subpopulations, each subpopulation has its own culture, and local search strategy is carried out for local optimization within each subpopulation. Each individual frog in a subpopulation has its own unique culture and influences the culture of other individuals, as well as being influenced by the culture of other individuals and evolving as the subpopulation evolves.

When each subpopulation evolves to a certain threshold, each subpopulation remixes together for cultural communication (global information exchange), so as to realize the mixed operation among subpopulations until the conditions set by the problem are satisfied. Similar to other optimization algorithms, SFLA has some necessary parameters (see Table 1).

Table 1. Related parameters and their corresponding meanings.

Parameter	Meaning
N	The total number of frogs in population
m	The number of subpopulations
n	The number of frogs in each subpopulation
Ds	The adjustment vector of the individual frog
$Dmax$	The maximum allowable step size of each individual frog in each jump
Xg	Global optima
Xb	Local optima
Xw	Local inferior quality
$Lmax$	The maximum times of subpopulation evolution
SF	Global information exchange times

2.2. Algorithm Update Strategy

Firstly, local search is carried out for each subpopulation, that is, update operation is carried out for individual frogs with the worst adaptive value in the subpopulation, and the update strategy is as follows.

The frog's regeneration distance is,

$$Ds = rand() \times (X_b - X_w) \quad (1)$$

The updated solution is,

$$X_w' = X_w + Ds(|Ds| \leq Dmax) \quad (2)$$

where $rand()$ represents a random number uniformly distributed between 0 and 1, Ds represents the adjustment vector of the individual frog, and $Dmax$ represents the maximum allowable step size of each individual frog in each jump.

2.2.1. Global Search Process

Step 1. Initialization. Determine the total number of frogs N in population, subpopulation number m , and number of frogs n in each subpopulation.

Step 2. The initial frog population N is randomly generated (multiple initial solutions are generated randomly), and the fitness value of each frog is calculated $P = \{X_1, X_2, \dots, X_F\}$. In the S -dimensional solution space, the i th frog is represented as $X_i = [X_{i1}, X_{i2}, \dots, X_{is}]$.

Step 3. Sort the fitness values in descending order, then record the corresponding optima of the current optima fitness value Xg , and the frog population is divided into several subpopulations. That is, N frogs are assigned to m subpopulations $M^1, M^2, M^3, \dots, M^m$; each one contains n frogs, satisfying $N = m \times n$. Distribution according to the rules:

$$M^k = \{X_{k+m(l-1)} \in P \mid 1 \leq k \leq m\} \quad (3)$$

where m is the number of divided subpopulations in population, M^k is the k th subpopulation, the first frog is distributed into the first subpopulation, the second frog is distributed into the second subpopulation, the m th frog is distributed into the m th subpopulation, and the $m + 1$ st frog is distributed into the first subpopulation in descending order, and the recursion is repeated until all frogs are distributed.

Step 4. According to the Equations (1) and (2) of SFLA and the constraints of solving the problem, meta evolution is carried out in each subpopulation.

Step 5. When each subpopulation evolves to a limited number of $Lmax$, each subpopulation is mixed. After a round of meta evolution for each subpopulation, frogs in each subpopulation are mixed again. It is still in descending order and subpopulation division, and the current global optima solution Xg is updated.

Step 6. Iteration completion criteria. If the convergence condition of the algorithm is satisfied, the execution process of the algorithm stops, otherwise it will jump to Step 3.

SFLA usually adopts three strategies to control the execution time of the algorithm.

- i. After continuous p times global exchange of ideas, the global optima Xg has not been greatly optimized.
- ii. Reach the function evaluation times which is set in advance before the algorithm is executed.
- iii. The fitness value of the optimal converges to the existing standard test results, and the absolute error is less than a certain threshold value.

No matter which stop condition is satisfied, the algorithm must be forced to exit the whole search cycle and output the optima.

2.2.2. Local Search Process

The local search process is the detailed process of step 4 of the global search process.

Step 4-1. Let $im = 0$, where im is the counter of the subpopulation for comparison with the number of subpopulations. Let $in = 0$, where in is the counter of local search evolution for comparison with $Lmax$.

Step 4-2. Xw and Xb are determined in the im th population. After D_S is calculated according to Equation (1), the worst solution is updated in Equation (2) to improve the position of the worst frog in the subpopulation. If the updated fitness value of the worst frog is better than the current fitness value, Xw' is used to replace Xw ; if the updated fitness value is inferior to the current fitness value, the global optima Xg is used to replace Xb . Additionally, then the local search process is performed according to Equations (1) and (2), if the updated fitness value of the worst frog is better than the current one, Xw' is used to replace Xw . Otherwise, a solution is randomly generated to replace Xw for the worst frog. The fitness values of the updated subpopulation are arranged in descending order. Set $in = in + 1$ and repeat this step until $in = Ls$.

Step 4-3. Let $im = im + 1$, and skip to step 4-2 until $im = m$.

Step 4-4. The global information exchange is performed when $im = m$.

3. SFLA Based on SA (SA-SFLA)

The common SA algorithm produces a new solution every time. According to Metropolis criterion, when temperature is T , whether to accept the new state is selected by probability.

$$\begin{cases} E_i' = E_j & (E_j \leq E_i) \\ \begin{cases} E_i' = E_i & (E_j > E_i, \text{rand}() \geq e^{-(E_j - E_i)/kT}) \\ E_i' = E_j & (E_j > E_i, \text{rand}() < e^{-(E_j - E_i)/kT}) \end{cases} \end{cases} \quad (4)$$

where T is temperature, the current state energy is E_i , and the new state energy is E_j . If $E_j < E_i$, the new state is accepted. If $E_j \geq E_i$, whether to accept the new state is selected with a certain probability P . Based on this, the new state with a large energy difference from the current state can be accepted at high temperature, while the new state with a small energy difference from the current state can only be accepted at low temperature. Thus, the algorithm is endowed with the ability to jump out of the local optimas and converge quickly in the process of searching for the optima.

In the early stage of algorithm optimization, the algorithm relies on simulated annealing optimization strategy to jump out the local optimas. Set the initial temperature in T and the temperature attenuation coefficient is α .

$$T' = T \times \alpha \quad (5)$$

When the SFLA judges that the inferior quality $X_{W'}$ is better than the updated inferior quality X_W , it receives the inferior quality with probability according to Equation (4) through Metropolis criterion.

$$\begin{cases} X_i' = X_j & (f(X_j) \leq f(X_i)) \\ \begin{cases} X_i' = X_i & (f(X_j) > f(X_i), \text{rand}() \geq e^{-(f(X_j) - f(X_i))/kT}) \\ X_i' = X_j & (f(X_j) > f(X_i), \text{rand}() < e^{-(f(X_j) - f(X_i))/kT}) \end{cases} \end{cases} \quad (6)$$

where $\text{rand}()$ is the mean and distributed random number between 0 and 1, and K is generally taken as 1.

4. SFLA of Threshold Oscillation Based on SA (SA-TO-SFLA)

As SFLA and SA-SFLA are susceptible to fall into local convergence and local optimas which are adjacent to the global optima in the process of global search on complex functions with high dimension and multi local optimum, SA-TO-SFLA is proposed. The improved algorithm adopts a multiple neighborhood SA strategy joined to the threshold oscillation of SFLA; thus, under the premise of ensuring the optimal convergence time, the global optima convergence time is greatly decreased. In the process of searching for the optima, using SA-TO-SFLA can effectively jump out of the local optimas and converge to the global optima.

In the process of searching for the optima by SFLA, all subpopulations participate in population iteration to accelerate convergence in the early stage of convergence. The form of the solution consists of $[X_1, X_2, \dots, X_k]$, and a solution has k dimensions. Therefore, the standard deviation of each individual frog can be calculated in each solution dimension in the population to judge the convergence degree of the frog population. When the standard deviation of the solution of frog population decreases to the preset threshold in the iterative process $[\sigma_1, \sigma_2, \dots, \sigma_k]$, the algorithm will divide a certain number of subpopulations in the population (the proportion of the number of divided subpopulation to the total subpopulation is ω) which will oscillate in a certain interval near the optimal solution of each subpopulation so as to provide local mutation solution to jump out of the local optima of complex function. The size of oscillation interval C is proportional to the threshold σ_k .

The remaining subpopulations still converge on the optima within each subpopulation. If there is a solution better than the current global optima in the subpopulation with oscillation strategy, the solution is immediately replaced by the optima in M^1 subpopulation.

When the standard deviation of the S -dimensional solution space of frog population meets the threshold $[\sigma_1, \sigma_2, \dots, \sigma_k]$, m frog populations are ranked from good to bad according to the quality of the best solution in their own population. Then, among the m sorted populations, the one in the middle is selected, and $m \times \omega - 1$ populations are selected from the top and bottom of the selected population. The sampling populations is defined as the regional oscillation subpopulation.

$$\begin{cases} U(t) = [M^{m/2+k} | (k = -\frac{m \times \omega}{2}, -\frac{m \times \omega}{2} + 1, \dots, \frac{m \times \omega}{2}), k \neq 0, m \times \omega \% 2 = 0)] \\ U(t) = [M^{m/2+k} | (k = \frac{-m \times \omega + 1}{2}, \frac{-m \times \omega + 1}{2} + 1, \dots, \frac{m \times \omega - 1}{2}), m \times \omega \% 2 = 1)] \end{cases} \quad (7)$$

In general, the selection of σ_k can be set according to the percentage of the total standard deviation S_0 after the frog population is initialized. For example, $S_0 = 1000$, the number of threshold values is selected according to the complexity of the function, such as $\sigma_1 = 0.3 \times S_0$, $\sigma_2 = 0.1 \times S_0$, and $\sigma_3 = 0.05 \times S_0$. The threshold number is determined according to the solution dimension of the function and the complexity of the function.

In the regional oscillatory subpopulation M^a , before each update evolution within the subpopulation, the current optima X_b will be added with random noise of a certain interval C , and then the worst solution will be closed to the optima after the oscillation; that is, the worst solution X_w of the subpopulation will be updated according to Equation (8).

$$X_b' = X_b + r = [X_{b1} + r_1, X_{b2} + r_2, \dots, X_{bs} + r_s | r_i \in (-c, c)] \quad (8)$$

If the updated X_b' or X_w is better than the current global optima, then the optima X_b of M^1 subpopulation is exchanged with X_b' or X_w of oscillatory subpopulation M^a , so as to play the role of partial subpopulation information exchange. Subpopulation M^1 is regarded as the optimal frog subpopulation. Through the "selective elimination mechanism", the oscillatory subpopulation continuously provides excellent individuals to lead the whole population to escape from the local optima.

$$ans = X_b, X_b = X_b', X_b' = ans \quad (X_b \in M^1, X_b' \in M^a, X_b' < X_b, X_b' \leq X_w) \quad (9)$$

$$ans = X_b, X_b = X_w, X_w = ans \quad (X_b \in M^1, X_w \in M^a, X_w < X_b, X_b' > X_w) \quad (10)$$

The M^a subpopulation is rearranged in descending order to carry out the evolution iteration of the next subpopulation. The flow chart of the algorithm is shown in Figures 1 and 2.

It is worth pointing out that in the original frog leaping rules of the standard SFLA, the frog imitation mechanism only imitates the jump from the worst frog to the best frog. The new position of the worst frog is limited to the straight line between the current position and the best frog position. In its iteration process, it may never evolve to the best position. It not only increases the convergence time but will lead to premature convergence. However, in

the above oscillation strategy, the Xb of partial subpopulations is dynamic in each iteration process, which leads to Xw being unrestricted by the frog leaping rules mentioned above. Therefore, it reduces the restrictions of frog leaping rules to a certain extent.

Figure 1. Algorithm global search flowchart.

Figure 2. Algorithm partial search flowchart.

The improvement of SA-TO-SFLA combines the advantages of SA based on material science and PSO based on genetic and on social behavior. It achieves a good balance between extensive search of multidimensional solution space and fast convergence of potential global optima. In SFLA, when the population evolves to the later stage, the involution occurs continuously in the population, a large number of invalid optimization processes are added, and the calculation cost remains unchanged while the income decreases, so it is difficult to make a big breakthrough. However, the SA-TO-SFLA can help the population to find a better evolutionary direction in the later stage.

5. Numerical Simulation Experiment Design

In order to verify the effectiveness of the SA-TO-SFLA, this experiment will test SFLA, SA-SFLA, TO-SFLA, and SA-TO-SFLA. This experiment uses professional algorithms to test the function [39,40] convergence accuracy and convergence time in Table 2.

Table 2. Definition of the multi-dimensional symmetric functions.

Name	Definition	Scope	Minimum Value
Drop-wave	$f_1 = -\frac{1+\cos(12\sqrt{x_1^2+x_2^2})}{0.5(x_1^2+x_2^2)+2}$	$[-5.12, 5.12]$	-1
Schaffer N.2	$f_2 = 0.5 + \frac{\sin^2(x_1^2-x_2^2)-0.5}{[1+0.001(x_1^2+x_2^2)]^2}$	$[-100, 100]$	0
Rastrigin	$f_3 = 20 + \sum_{i=1}^2 [x_i^2 - 10 \cos(2\pi x_i)]$	$[-5.12, 5.12]$	0
Griewank	$f_4 = \sum_{i=1}^2 \frac{x_i^2}{4000} - \prod_{i=1}^2 \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-5, 5]$	0

The drop-wave function, Schaffer function N.2, Rastrigin function, and Griewank function are used in order of difficulty of function test from high to low [39,40]. Performance tests are performed on SFLA, SA algorithm, SA-SFLA, and SA-TO-SFLA. The function images are shown in Figure 3.

The parameters are set as follows.

i. When the four combined frog leaping algorithms are on Griewank and Rastrigin functions, $f = 100$ frogs are divided into $m = 10$ subpopulations, the number of frogs in the subpopulation is $n = 10$, the number of iterations within the subpopulation is $L_{max} = 10$, and the number of global information exchange is $SF = 250$. Among them, the $T_0 = 200$ in SA-SFLA, the attenuation coefficient α is a symmetrical random number between 0.3 and 0.5. $\sigma_1 = 0.5, \sigma_2 = 1.3, \sigma_3 = 3.4, \sigma_4 = 4.6$ in the SFLA with oscillation strategy, and the oscillation strategy interval is, respectively 0.01, 0.05, 0.1, or 0.2 of the value range of independent variable.

When the four combined frog leaping algorithms are on Schaffer and drop-wave functions, $f = 100$ frogs are divided into $m = 10$ populations, the number of frogs in the subpopulation is $n = 10$, the number of iterations within the population is $L_{max} = 10$, and the number of global information exchange is $SF = 50$. Among them, the $T_0 = 200$ in SA-SFLA, $\sigma_1 = 0.5, \sigma_2 = 1.3, \sigma_3 = 3.4, \sigma_4 = 4.6$ in drop-wave function, and the oscillation strategy interval is, respectively 0.01, 0.05, 0.1, or 0.2 of the value range of independent variable. The oscillation strategy interval is, respectively 0.01, 0.05, 0.1, or 0.2 of the value range of independent variable in Schaffer N.2 function, and $\sigma_1 = 70, \sigma_2 = 90, \sigma_3 = 120, \sigma_4 = 140$.

ii. Parameter setting of each improved algorithm. All algorithms use the parameters set in (i) above; that is, if there are corresponding parameters in the algorithm, they will be adopted, if not, they will not be used.

The performance test method is as follows: fixed convergence precision of 0.001, and fixed maximum global iteration times of 200 (except SA, as it is not SIEA class algorithm). When the algorithm reaches the convergence precision within the maximum number of global iterations, it exits the iteration, and records the convergence time and the number of global iterations. If the algorithm still fails to converge to the convergence precision when

the maximum number of global iterations is reached, it is the convergence failure, so as to test the convergence accuracy and convergence time of the algorithm.

Figure 3. (a) The image of simple level test Griewank function f_4 in the value range of $X_1 = (-5,5)$, $X_2 = (-5,5)$; (b) The image of middle level test Rastrigin function f_3 in the value range of $X_1 = (-5,5)$, $X_2 = (-5,5)$; (c) Detailed image of local optimas and global optima of Rastrigin function in the value range of $X_1 = (-5,5)$, $X_2 = (-5,5)$; (d) The image of mid high level test Schaffer function N.2 f_2 in the value range of $X_1 = (-5,5)$, $X_2 = (-5,5)$; (e) Detailed image of Schaffer function N.2 in the value range of $X_1 = (-2,2)$, $X_2 = (-2,2)$; (f) The image of high level test drop-wave function f_1 in the value range of $X_1 = (-5,5)$, $X_2 = (-5,5)$; (g) Detailed image of drop-wave function in the value range of $X_1 = (-2,2)$, $X_2 = (-2,2)$.

6. Results and Discussion

6.1. Comparison among SFLA, SA, SA-SFLA, TO-SFLA, and SA-TO-SFLA

Table 3 shows the experimental results of convergence test of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA in the drop-wave function. From the data, it can be seen that the SA-TO-SFLA has the highest convergence accuracy, reaching 99.0%, while the SA-SFLA has the lowest convergence accuracy, which is only 57%; in terms of the median convergence time, SA-SFLA has the shortest convergence time, reaching 32.06 s.

Table 3. Drop-wave function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	76.56	75.23	2.98	40.57	63.67	86.01	243	90.0
	Epoches	11.68	19.53	1.00	6.00	8.00	11.00	50	
SA	Time(/s)	169.45	13.58	146	157	168	180	196	66.8
	Epoches	Epoches = 600, T = 1000							
SA-SFLA	Time(/s)	49.75	39.20	8.80	24.75	32.06	59.63	194.68	57.0
	Epoches	14.01	11.54	2.00	7.00	9.00	17.00	48.00	
SA-TO-SFLA	Time(/s)	57.99	34.35	5.89	33.34	48.64	72.18	180.2	99.0
	Epoches	13.65	8.08	1.00	8.00	11.00	17.00	43.00	
TO-SFLA	Time(/s)	46.64	29.12	6.29	22.95	39.36	62.76	143.18	98.1
	Epoches	12.45	8.04	1.00	6.25	10.5	17.00	38.00	

Table 4 shows the experimental results of convergence test of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA on Schaffer function N.2. It can be seen from the data that the convergence accuracy of SA algorithm is only 85.5%, and the convergence accuracy of other algorithms is 100%. However, from the median of the convergence time, the convergence time of SA algorithm is only 0.52 s, and other algorithms are more than 32 s.

Table 4. Schaffer function N.2 test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	68.52	42.16	8.98	44.13	59.71	81.75	281.32	100
	Epoches	9.55	5.55	1.00	6.00	9.00	11.00	42.00	
SA	Time(/s)	0.83	0.84	0.004	0.21	0.52	1.23	3.89	85.8
	Epoches	Epoches = 600, T = 1000							
SA-SFLA	Time(/s)	34.86	17.38	3.03	22.22	33.22	42.68	105.92	100
	Epoches	8	6.87	3.422	1.00	5.00	7.00	200	
SA-TO-SFLA	Time(/s)	33.04	14.33	2.80	23.26	32.07	41.81	82.66	100
	Epoches	6.83	2.95	1.00	5.00	7.00	9.00	17.00	
TO-SFLA	Time(/s)	56.96	32.37	9.09	34.27	49.84	74.10	184.48	100
	Epoches	9.01	4.39	1.00	6.00	8.00	11.00	30.00	

Table 5 shows the experimental results of convergence test of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA on Rastrigin function. It can be seen from the data that the convergence accuracy of SFLA is the lowest, only 78.0%, while the accuracy of SA-TO-SFLA and SA algorithm is 100%. From the median of convergence time, SA algorithm has the shortest convergence time, which is only 4.35 s, while TO-SFLA has the most convergence time, which is 27.76 s.

Table 5. Rastrigin Function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	28.05	41.58	3.75	10.18	12.93	17.96	198.98	78.0
	Epoches	34.30	53.30	4.00	12.00	15.00	20.00	246.00	
SA	Time(/s)	4.41	0.28	4.06	4.21	4.35	4.54	5.70	100
	Epoches	Epoches = 200, T = 200							
SA-SFLA	Time(/s)	15.16	7.6	2.38	9.8	13.4	19.1	45.93	89.0
	Epoches	16.44	6.14	3.00	12.00	15.00	20.00	31.00	
SA-TO-SFLA	Time(/s)	38.72	32.16	5.38	19.12	24.06	45.59	193.80	100
	Epoches	43.47	34.92	6.00	21.00	32.00	52.00	210	
TO-SFLA	Time(/s)	37.10	30.39	3.01	17.99	27.76	43.61	175.96	99.5
	Epoches	39.96	31.98	3.00	19.00	30.00	49.00	195.00	

Table 6 shows the experimental results of convergence test of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA on Girewank function. It can be seen from the data that the convergence accuracy of SFLA is the lowest, which is 92.5%, while the accuracy of TO-SFLA and SA algorithm are both as high as 99.7% and 98%, respectively. From the median of convergence time, SA-SFLA has the shortest convergence time, which is only 2.36 s, while SFLA has the longest convergence time, which is 8.74 s.

Table 6. Griewank Function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	31.00	74.00	1.13	5.53	8.74	12.62	578	92.5%
	Epoches	17.13	39.55	1.00	3.00	5.00	7.00	223.00	
SA	Time(/s)	5.85	0.95	5.08	5.29	5.60	6.00	10.49	98%
	Epoches	Epoches = 200, T = 200							
SA-SFLA	Time(/s)	3.44	3.13	0.48	1.37	2.36	4.50	23.26	94.7%
	Epoches	3.24	4.88	1.00	2.00	3.00	5.00	13.00	
SA-TO-SFLA	Time(/s)	4.80	6.05	0.47	1.40	3.04	5.61	53.94	97.5%
	Epoches	4.47	6.19	1.00	2.00	3.00	5.00	59.00	
TO-SFLA	Time(/s)	5.79	12.44	0.48	1.52	3.30	5.94	140.85	99.7%
	Epoches	5.04	13.52	1.00	2.00	3.00	5.00	149.00	

i. Convergence accuracy analysis. From Figure 4, it can be seen that TO-SFLA and SA-TO-SFLA with threshold oscillation strategy can achieve convergence accuracy of more than 97.5% in four test functions with increasing difficulty level in turn. While maintaining a high accuracy, the accuracy curve is extremely stable. Additionally, obviously, with the increasing complexity of functions, the accuracy of SA-TO-SFLA is still high, while the accuracy of TO-SFLA has begun to show a downward trend. The curve of the SFLA has greater volatility, and its performance is the worst on the Girewank function. It can be seen from Figure 3a that, although the function has only four local optimas, the difference between the local optimas and the global optima is very small, which leads to the SFLA often converging with the local optimas due to the lack of a strong enough jump out mechanism. However, on the Rastrigin function, it can be seen from Figure 3b that there appears a large number of local optimas, and the global optima are surrounded by multiple identical local optimas, which has strong confusion for SFLA and SA-SFLA, and making the algorithm often fall into the local optimas as shown in Figure 3c. On Schaffer N.2, each algorithm performs better. In the drop-wave function, as the function fluctuates violently, the convergence of the algorithm takes a long time, and there are many iterations. Therefore, before the algorithm finds the global optima, the temperature of the simulated annealing strategy has dropped to the extent that it is not able to accept the inferior quality. However,

the simulated annealing strategy cannot jump out or has not yet converged in the specified round in the early stage due to receiving a large number of poor solutions. On the contrary, it has a restraining effect on the algorithm, which leads to a significant decrease in the accuracy of the algorithm.

Figure 4. On the premise that 0.001 is the convergence precision and 200 rounds is the maximum number of iterations, the convergence accuracy of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA is significantly different after being tested by four functions: Girewank, Rastrigin, Schaffer, and drop-wave.

ii. Convergence time analysis. As can be seen from Figure 5, in the Girewank function, SFLA takes the longest time to find convergence, which is the same as above. In this function, there are four identical local optima approximate to the global optima, which makes the frog in the algorithm jump back and forth among multiple optima and prolong the convergence time, while other algorithms have similar time-consuming issues. In the Rastrigin function, the convergence time of SFLA and SA-SFLA is better than that of TO-SFLA and SA-TO-SFLA, as when testing with SFLA or SA-SFLA, at the later stage of convergence, all frog individuals gather around a point (the global optima or local optima), while the TO-SFLA and SA-TO-SFLA divide a certain proportion of subpopulation to implement the oscillation strategy, and the remaining subpopulations converge. By comparing that, the number of frogs converging in the TO-SFLA and SA-TO-SFLA is less than that in the SFLA and SA-SFLA, so the convergence time of the TO-SFLA and SA-TO-SFLA increases slightly. However, SA-TO-SFLA can jump out of local optima solutions faster than TO-SFLA due to the simulated annealing strategy in the early stage, so SA-TO-SFLA is faster than TO-SFLA. In Schaffer function N.2 and drop-wave functions, SFLA needs to spend a lot of time jumping between different local optima. The convergence time of SA-TO-SFLA and SA-SFLA tested on Schaffer N.2 function is similar, as only the central region of the function has a large range of violent fluctuations. Relying on the poor solution acceptability of simulated annealing strategy, a frog population can quickly converge to the central region, and the central global optima is in a large range. Thus, it provides a good convergence direction for both algorithms, which makes the convergence time short. However, TO-SFLA does not have the global acceptability of poor solutions of simulated annealing strategy in the early stage, so it will spend more time on some local optima, which makes the convergence time long. In the drop-wave function, the convergence time of TO-SFLA is shorter than that of SA-TO-SFLA, while SA-SFLA still maintains a shorter

convergence time. However, it can be seen from Figure 4 that the convergence accuracy of both algorithms has decreased, as the algorithm falls into the local optima, and the convergence time is decreased by sacrificing the convergence accuracy.

Figure 5. On the premise that 0.001 is the convergence precision and 200 rounds is the maximum number of iterations, the convergence time and standard deviation curve of SFLA, SA, SA-SFLA, SA-TO-SFLA, and TO-SFLA on four functions: Girewank, Rastrigin, Schaffer, and drop-wave.

iii. The strategy of SA-TO-SFLA has strong robustness and effectiveness, and its convergence time has been greatly decreased on the SFLA. Although the convergence time is slightly longer than SA-SFLA, the convergence accuracy is more stable and the convergence accuracy is higher than TO-SFLA and SA-SFLA. In more complex functions, the advantages of SA-TO-SFLA algorithm will be more obvious.

6.2. Comparison with Other Improved Algorithms

Table 7 shows the experimental results of convergence test of SFLA, ISFLA, MSFLA, SA-TO-SFLA and DSFLA on drop-wave function. It can be seen from the data that the convergence accuracy of MSFLA is the lowest, only 32.4%, while that of SA-TO-SFLA is 99%. In terms of the median convergence time, ISFLA has the shortest convergence time, which is 34.1 s, while SFLA has the longest convergence time, which is 63.6 s.

Table 7. Drop-wave function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	76.56	75.23	2.98	40.57	63.67	86.01	243	90.0
	Epoches	11.68	19.53	1.00	6.00	8.00	11.00	50	
ISFLA	Time(/s)	48.12	42.89	6.13	25.61	34.17	54.43	260	57.6
	Epoches	11.00	9.16	1.00	6.00	9.00	11.00	48.00	
MSFLA	Time(/s)	57.23	54.86	9.89	24.60	34.89	63.14	289.54	32.4
	Epoches	12.37	12.20	2.00	5.00	7.00	12.00	47.00	
SA-TO-SFLA	Time(/s)	57.99	34.35	5.89	33.34	48.64	72.18	180.2	99.0
	Epoches	13.65	8.08	1.00	8.00	11.00	17.00	43.00	
DSFLA	Time(/s)	56.46	45.77	6.60	24.81	35.99	76.45	193.26	66.7
	Epoches	12.20	12.96	1.00	5.00	7.00	11.00	48.00	

Table 8 shows the experimental results of convergence test of SFLA, ISFLA, MSFLA, SA-TO-SFLA, and DSFLA on the Schaffer function. From the data, it can be seen that the convergence accuracy of all the algorithms is 100%. From the median of convergence time, SA-TO-SFLA has the shortest convergence time, which is 32 s, while SFLA has the longest convergence time, which is 59.71 s.

Table 8. Schaffer function N.2 test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	68.52	42.16	8.98	44.13	59.71	81.75	281.32	100
	Epoches	9.55	5.55	1.00	6.00	9.00	11.00	42.00	
ISFLA	Time(/s)	55.8	45.0	3.60	33.68	49.23	75.08	196.32	100
	Epoches	7.14	3.81	1.00	4.00	7.00	9.00	26.00	
MSFLA	Time(/s)	45.29	22.56	9.81	32.28	38.76	55.17	158.97	100
	Epoches	6.44	3.43	1.00	5.00	6.00	8.00	24.00	
SA-TO-SFLA	Time(/s)	33.04	14.33	2.80	23.26	32.07	41.81	82.66	100
	Epoches	6.83	2.95	1.00	5.00	7.00	9.00	17.00	
DSFLA	Time(/s)	44.16	23.86	4.12	27.32	37.53	55.63	124.15	100
	Epoches	5.17	2.42	1.00	3.75	5.00	6.00	18.00	

Table 9 shows the experimental results of convergence test of SFLA, ISFLA, MSFLA, SA-TO-SFLA, and DSFLA on the Rastrigin function. It can be seen from the data that the convergence accuracy of MSFLA is the lowest, which is only 53.3%, while the SA-TO-SFLA has the highest convergence accuracy of 100%. From the median of convergence time, SFLA has the shortest convergence time, which is 12.93 s, while SA-TO-SFLA has the longest convergence time, which is 24.06 s.

Table 9. Rastrigin function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	28.05	41.58	3.75	10.18	12.93	17.96	198.98	78.0
	Epoches	34.30	53.30	4.00	12.00	15.00	20.00	246.00	
ISFLA	Time(/s)	22.91	13.40	7.79	13.65	18.92	29.31	106.86	69.8
	Epoches	13.14	6.85	4.00	10.00	12.00	14.25	48.00	
MSFLA	Time(/s)	21.24	13.80	2.71	13.39	17.54	24.90	119.49	53.3
	Epoches	10.11	6.49	1.00	7.00	9.00	10.00	44.00	
SA-TO-SFLA	Time(/s)	38.72	32.16	5.38	19.12	24.06	45.59	193.80	100
	Epoches	43.47	34.92	6.00	21.00	32.00	52.00	210	
DSFLA	Time(/s)	22.90	10.86	7.80	15.76	18.78	28.42	75.62	79.5
	Epoches	9.18	4.59	4.00	7.00	9.00	10.00	45.00	

Table 10 shows the experimental results of convergence test of SFLA, ISFLA, MSFLA, SA-TO-SFLA, and DSFLA on the Girewank function. It can be seen from the data that the convergence accuracy of SFLA is the lowest, which is only 92.5%, while the convergence accuracy of SA-TO-SFLA is also not high, reaching 97.5%, and the other algorithms are 100%. From the median of convergence time, SA-TO-SFLA has the shortest convergence time, which is 3.04 s, while SFLA has the longest convergence time, which is 8.74 s.

i. Convergence accuracy analysis. In Figure 6, the accuracy of ISFLA, MSFLA, and DSFLA tested on the Girewank functions is better than SA-TO-SFLA and SFLA, but it is obvious that the accuracy of SFLA performs worse, while SA-TO-SFLA still has an accuracy of 97.5%. On the Rastrigin function, the convergence accuracy of SA-TO-SFLA is much better than that of other algorithms as the global is spread over the local optimas. Other algorithms focus on the improvement of convergence accuracy and the decrease in convergence time, but the convergence accuracy is much worse. On the Schaffer N.2 function, the

convergence accuracy of all algorithms is as high as 100%. However, on the more complex drop-wave function, the convergence accuracy of each algorithm is obviously improved. The convergence accuracy of the SA-TO-SFLA is still very high, but the convergence accuracy of other algorithms is not even as high as that of the SA-SFLA, as the improved algorithms are aimed at the convergence time. However, from their improved strategies, it can be seen that the faster the frog jumps, the longer the step size is, and it is easy to skip the high-quality solution. The differential algorithm introduced by DSFLA makes frogs randomly generated in the whole frog population, rather than in the global random generation. This will lead to that when the algorithm is in a complex function, if it converges to the local optima temporarily in the process of convergence and the frog population does not include the global optima in the frog population range, the algorithm will no longer have the ability to jump out of the local optima. This is why DSFLA performs better than SFLA, ISFLA, and MSFLA on Griewank, Rastrigin, and Schaffer N.2 functions, but has a very low accuracy in drop-wave function.

ii. Convergence time analysis. In Figure 7, the convergence time of SA-TO-SFLA on the simple Griewank function keeps the first place with a slight advantage. This is as the difference between the local optima and the global optima of this function is small, and it has great confusion for other algorithms, making the frog population jump back and forth among multiple the optima, while the SA-TO-SFLA adopts the measure of partial subpopulation convergence and partial subpopulation oscillation to make the convergence and jump out of local optima division parallel, so its convergence time is optimal. In the next three functions, except Schaffer N.2 function, the convergence time of SA-TO-SFLA is not outstanding in other algorithms, as other algorithms focus on the decrease in convergence time. From Figure 7, it can also be seen that the convergence time of the SFLA decreases greatly as the function becomes more complex.

iii. Summary. Compared with SFLA, ISFLA, MSFLA, and DSFLA, SA-TO-SFLA has a strong advantage in the convergence accuracy, and the convergence accuracy fluctuates little with the function complexity, while the accuracy of other algorithms decreases greatly with the increase in function complexity; that is, the advantage of SA-TO-SFLA in convergence accuracy will be more obvious and prominent in the more complex functions. In terms of convergence time, although the convergence time of SA-TO-SFLA is not as short as other algorithms, the convergence time interval between algorithms is very large and, compared with SFLA, it has a greater time decrease. Therefore, SA-TO-SFLA is very excellent. It overcomes the shortcomings of long convergence time and low convergence accuracy of SFLA under certain precision. It has great application potential in the fields of robot foraging, group control, path planning, etc.

Table 10. Griewank function test data.

Algorithm	Metric	Mean	Std	Min	25%	50%	75%	Max	Acc (%)
SFLA	Time(/s)	31.00	74.00	1.13	5.53	8.74	12.62	578	92.5
	Epoques	17.13	39.55	1.00	3.00	5.00	7.00	223.00	
ISFLA	Time(/s)	4.41	2.30	1.02	2.66	4.32	5.84	12.7	100
	Epoques	2.04	1.45	1.00	2.00	3.00	4.00	6.00	
MSFLA	Time(/s)	5.52	2.45	1.51	3.55	5.64	7.52	12.13	100
	Epoques	1.71	1.16	1.00	2.00	3.00	4.00	6.00	
SA-TO-SFLA	Time(/s)	4.80	6.05	0.47	1.40	3.04	5.61	53.94	97.5
	Epoques	4.47	6.19	1.00	2.00	3.00	5.00	59.00	
DSFLA	Time(/s)	4.91	2.39	1.37	3.29	5.02	6.08	13.02	100
	Epoques	1.17	1.19	1.00	2.00	3.00	3.00	6.00

Figure 6. On the premise that 0.001 is the convergence precision and 200 rounds is the maximum number of iterations, it shows the significant difference between the convergence accuracy of SFLA, ISFLA, MSFLA, SA-TO-SFLA, and DSFLA on the four functions: Girewank, Rastrigin, Schaffer, and drop-wave.

Figure 7. On the premise that 0.001 is the convergence precision and 200 rounds is the maximum number of iterations, the convergence time and standard deviation curve of SFLA, ISFLA, MSFLA, SA-TO-SFLA, and DSFLA on four functions: Girewank, Rastrigin, Schaffer, and drop-wave.

7. Conclusions

This paper aims to overcome the shortcomings of the SFLA by proposing SA-TO-SFLA. SFLA is easy to converge prematurely and easily falls into local optimas. Thus, the threshold oscillation strategy is applied to the intermediate subpopulation in the later stage of the SA-TO-SFLA convergence, so that the high-quality subpopulation continues to search for the current local optimas to ensure the convergence time, while the oscillatory subpopulation continuously provides the most variant solution, and when the high-quality solution is obtained, it will immediately exchange information with the optimal subpopulation. The

results show that the convergence time of SA-TO-SFLA on complex functions is decreased greatly with a certain degree of security, and the more complex the function is, the better the performance of the algorithm is. The effectiveness of the algorithm is also verified by the later simulation data.

With the rapid development of AI, SIEA and ANN both perform well in different fields. One of AI's significant functions is to liberate the human labor force. The combination of SIEA and ANN is a trend. The optimization algorithm of ANN can be replaced by SIEA. In this paper, the SA-TO-SFLA is proposed to optimize the maximum and minimum value of the objective function by simulating the evolution process of frog population, and the convergence time is very short and convergence accuracy are very high.

Potential direction of further work is to combine this algorithm with ANN from two aspects: the first point is to change the optimization algorithm of ANN to the SA-TO-SFLA to test the training time, training accuracy, and generalization ability of ANN; the second point is to take each ANN model as an individual of a frog population and build a small ANN subpopulation. Each ANN model constantly learns from each other, exchanging information, then realizing the collective evolution, and verifying the training time, training accuracy, and generalization ability of the optimal ANN model in the population. According to this, the effectiveness and reliability of SA-TO-SFLA are further verified by combining with the actual application scenarios.

Author Contributions: Data curation, W.G.; methodology, X.D.; software, J.W.; validation, X.G.; supervision, F.L.; project administration, L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China (No. 41774147), the National Key R&D Project (No. 2017YFC0602100) and Sichuan Science and Technology Program (2021YJ0522). We thank the assistance of Sichuan Provincial Key Lab of Applied Nuclear Techniques in Geosciences, we also thank CDUT Team 203 for the English language review.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled, "An improved shuffled frog leaping algorithm of threshold oscillation based on simulated annealing".

References

1. Acharya, U.R.; Fujita, H.; Sudarshan, V.K.; Oh, S.L.; Muhammad, A.; Koh, J.E.; Tan, J.H.; Chua, C.K.; Chua, K.P.; San Tan, R. Application of empirical mode decomposition (EMD) for automated identification of congestive heart failure using heart rate signals. *Neural Comput. Appl.* **2017**, *28*, 3073–3094. [\[CrossRef\]](#)
2. Lei, X.; Ding, Y.; Fujita, H.; Zhang, A. Identification of dynamic protein complexes based on fruit fly optimization algorithm. *Knowl.-Based Syst.* **2016**, *105*, 270–277. [\[CrossRef\]](#)
3. Bench-Capon TJ, M.; Dunne, P.E. Argumentation in artificial intelligence. *Artif. Intell.* **2007**, *171*, 619–641. [\[CrossRef\]](#)
4. Hassabis, D.; Kumaran, D.; Summerfield, C.; Botvinick, M. Neuroscience-Inspired Artificial Intelligence. *Neuron* **2017**, *95*, 245–258. [\[CrossRef\]](#)
5. Ramos, C.; Augusto, J.C.; Shapiro, D. Ambient Intelligence—The Next Step for Artificial Intelligence. *IEEE Intell. Syst.* **2008**, *23*, 15–18. [\[CrossRef\]](#)
6. Hutson, M. Missing data hinder replication of artificial intelligence studies. *Science* **2018**. [\[CrossRef\]](#)
7. Marshall, J.A. Neural networks for pattern recognition. *Neural Netw.* **1995**, *8*, 493–494. [\[CrossRef\]](#)
8. Chua, L.O.; Yang, L. Cellular Neural Network: Applications. *IEEE Trans. Circuits Syst.* **1988**, *35*, 1273–1290. [\[CrossRef\]](#)
9. West, D. Neural network credit scoring models. *Comput. Oper. Res.* **2000**, *27*, 1131–1152. [\[CrossRef\]](#)
10. Zhou, L.; Tam, K.P.; Fujita, H. Predicting the listing status of Chinese listed companies with multi-class classification models. *Inf. Sci.* **2016**, *328*, 222–236. [\[CrossRef\]](#)

11. Zhao, L.; Zhou, Y.; Lu, H.; Fujita, H. Parallel computing method of deep belief networks and its application to traffic flow prediction. *Knowl.-Based Syst.* **2019**, *163*, 972–987. [\[CrossRef\]](#)
12. Fujita, H.; Kurematsu, M.; Hakura, J. Virtual Doctor System (VDS): Aspects on Reasoning Issues. *Front. Artif. Intell. Appl.* **2011**, *231*, 293–304. [\[CrossRef\]](#)
13. Ali, M.; Dapoigny, R. Local search algorithm for unicost set covering problem. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2006; Chapter 34; pp. 302–311. [\[CrossRef\]](#)
14. Pobil, A.P.D.; Mira, J.; Ali, M. Tasks and Methods in Applied Artificial Intelligence. In Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA-98-AIE Benicàssim, Castellón, Spain, 1–4 June 1998.
15. Gao, P.; Yuan, R.; Wang, F.; Xiao, L.; Fujita, H.; Zhang, Y. Siamese attentional keypoint network for high performance visual tracking. *Knowl.-Based Syst.* **2020**, *193*, 105448. [\[CrossRef\]](#)
16. Shihabudheen, K.V.; Pillai, G.N. Recent advances in neuro-fuzzy system: A survey. *Knowl.-Based Syst.* **2018**, *152*, 136–162. [\[CrossRef\]](#)
17. Wang, J.; Xu, C.; Tavoosi, J. A Novel Nonlinear Control for Uncertain Polynomial Type-2 Fuzzy Systems (Case Study: Cart-Pole System). *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2021**, *29*, 753–770. [\[CrossRef\]](#)
18. Mohammadi Moghadam, H.; Mohammadzadeh, A.; Hadjiaghaie Vafaie, R.; Tavoosi, J.; Khooban, M.-H. A type-2 fuzzy control for active/reactive power control and energy storage management. *Trans. Inst. Meas. Control* **2021**. [\[CrossRef\]](#)
19. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–489. [\[CrossRef\]](#) [\[PubMed\]](#)
20. Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. Mastering the game of Go without human knowledge. *Nature* **2017**, *550*, 354–359. [\[CrossRef\]](#) [\[PubMed\]](#)
21. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. *arXiv* **2020**, arXiv:2005.14165.
22. Ding, W.; Jiang, H.; Ali, M.; Li, M. Modern Advances in Intelligent Systems and Tools. In Proceedings of the 25th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE 2012), Dalian, China, 9–12 June 2012; Springer Publishing Company, Incorporated: Berlin/Heidelberg, Germany, 2011.
23. Li, Z.; Guo, S.; Wang, F.; Lim, A. Improved GRASP with Tabu Search for Vehicle Routing with Both Time Window and Limited Number of Vehicles. In *International Conference on Innovations in Applied Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2004.
24. Van, D.A.; Gary, V.; Lamont, B. Multiobjective Evolutionary Algorithm Research: A History and Analysis. *Evol. Comput.* **1998**, *8*, 1–88.
25. Han, K.H.; Kim, J.H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE Trans. Evol. Comput.* **2002**, *6*, 580–593. [\[CrossRef\]](#)
26. Chaves-Gonzalez, J.M.; Perez-Toledano, M.A.; Navasa, A. Software requirement optimization using a multiobjective swarm intelligence evolutionary algorithm. *Knowl.-Based Syst.* **2015**, *83*, 105–115. [\[CrossRef\]](#)
27. Purnomo, H.D.; Wee, H.M. Soccer Game Optimization: An Innovative Integration of Evolutionary Algorithm and Swarm Intelligence Algorithm. In *Research Methods Concepts Methodologies Tools & Applications*; IGI Global: Hershey, PA, USA, 2015.
28. Ding, M.; Dong, W. Multi-Working Mode Product Color Planning Using Evolutionary Algorithm and Swarm Intelligence. *J. Comput. Theor. Nanoence* **2013**, *10*, 2906–2911. [\[CrossRef\]](#)
29. Zhang, X.; Sun, B.; Mei, T.; Wang, R. A Novel Evolutionary Algorithm Inspired by Beans Dispersal. *Int. J. Comput. Intell. Syst.* **2013**, *6*, 79–86. [\[CrossRef\]](#)
30. Eusuff, M.M.; Lansey, K.E. Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm. *Water Resour. Plan. Manag.* **2003**, *129*, 210–225. [\[CrossRef\]](#)
31. Eusuff, M.; Lansey, K.; Pasha, F. Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. *Eng. Optim.* **2006**, *38*, 129–154. [\[CrossRef\]](#)
32. Zhang, X.; Hu, X.; Cui, G.; Wang, Y.; Niu, Y. An improved shuffled frog leaping algorithm with cognitive behavior. In Proceedings of the 2008 7th World Congress on Intelligent Control and Automation, Chongqing, China, 25–27 June 2008.
33. Wang, Q.; Hao, Y.; Sun, X. Modified shuffled frog leaping algorithm with convergence of update process in local search. In Proceedings of the 2011 First International Conference on Instrumentation, Measurement, Computer, Communication and Control, Beijing, China, 21–23 October 2011.
34. Metropolis, R.; Rosenbluth, A.; Rosenbluth, M.; Teller, A.; Teller, E. Simulated annealing. *J. Chem. Phys.* **1953**, *21*, 1087–1092. [\[CrossRef\]](#)
35. Du, J.; Kong, X.; Zuo, X.; Zhang, L.; Ouyang, A. Shuffled Frog Leaping Algorithm for Hardware/Software Partitioning. *J. Comput.* **2014**, *9*, 2752–2760. [\[CrossRef\]](#)
36. Jaballah, S.; Rouis, K.; Abdallah, F.B.; Tahar, J.B.H. An improved Shuffled Frog Leaping Algorithm with a fast search strategy for optimization problems. In Proceedings of the 2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 4–6 September 2014.

37. Ahandani, A.M. A diversified shuffled frog leaping: An application for parameter identification. *Appl. Math. Comput.* **2014**, *239*, 1–16. [[CrossRef](#)]
38. Wang, H.B.; Zhang, K.P.; Tu, X.Y. A mnemonic shuffled frog leaping algorithm with cooperation and mutation. *Appl. Intell.* **2015**, *43*, 32–48. [[CrossRef](#)]
39. Harrison, K.R.; Engelbrecht, A.P.; Ombuki-Berman, B.M. Inertia weight control strategies for particle swarm optimization. *Swarm Intell.* **2016**, *10*, 267–305. [[CrossRef](#)]
40. Molga, M.; Smutnicki, C. Test Functions for Optimization Needs. 2005. Available online: <http://zsd.ict.pwr.wroc.pl/> (accessed on 15 November 2021).