Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2014, Article ID 892914, 11 pages
http://dx.doi.org/10.1155/2014/892914

Research Article

Lifecycle-Based Swarm Optimization Method for Numerical Optimization

Hai Shen,[1,2] Yunlong Zhu,[2] and Xiaodan Liang[3]

[1]*College of Physics Science and Technology, Shenyang Normal University, Shenyang 110023, China*
[2]*Laboratory of Information Service and Intelligent Control, Shenyang Institute of Automation, Chinese Academy of Sciences Shenyang, Shenyang 110016, China*
[3]*School of Computer Science & Software Engineering, Tianjin Polytechnic University, Tianjin 300387, China*

Correspondence should be addressed to Yunlong Zhu; ylzhusia@163.com

Received 19 October 2014; Revised 18 November 2014; Accepted 23 November 2014; Published 11 December 2014

Academic Editor: Muhammad Naveed Iqbal

Bioinspired optimization algorithms have been widely used to solve various scientific and engineering problems. Inspired by biological lifecycle, this paper presents a novel optimization algorithm called lifecycle-based swarm optimization (LSO). Biological lifecycle includes four stages: birth, growth, reproduction, and death. With this process, even though individual organism died, the species will not perish. Furthermore, species will have stronger ability of adaptation to the environment and achieve perfect evolution. LSO simulates Biological lifecycle process through six optimization operators: chemotactic, assimilation, transposition, crossover, selection, and mutation. In addition, the spatial distribution of initialization population meets clumped distribution. Experiments were conducted on unconstrained benchmark optimization problems and mechanical design optimization problems. Unconstrained benchmark problems include both unimodal and multimodal cases the demonstration of the optimal performance and stability, and the mechanical design problem was tested for algorithm practicability. The results demonstrate remarkable performance of the LSO algorithm on all chosen benchmark functions when compared to several successful optimization techniques.

1. Introduction

In nature, biology species are divers and an organism is any living thing (such as animal, plant, or microorganism) [1]. All their behaviors can show what kind of biological features they have. Some features are universality, such as foraging, reproduction, mutation, and metabolism. And for some organisms, their features are uniqueness and intelligence [2]. The ant possesses division and cooperation behaviors. Bees have special skills in the process of gathering honey. Birds have unique flight principle. The bacterial flagellums play a role of chemotaxis in their moving. Biologic features enable organisms to adapt to the complex living environment in the best way and long-term survival in nature. Real-world optimization problems are similar to biologic survival environment; they all have complex features. Therefore, with the purpose of solving reality complex problem, researchers begin to mimic the biologic phenomena via defining a set of rules and realize those rules on computer [3]. Those rules are called bioinspired optimization technique.

Currently, the bioinspired optimization techniques possessing abundant research results, and we divide all existing algorithms into three major categories: evolutionary computation, swarm intelligence, and others. Widely concerned algorithms are as follows:

(1) evolutionary computation:

(i) genetic algorithm;
(ii) evolutionary programming;
(iii) evolutionary strategy;
(iv) genetic programming;
(v) differential evolution;
(vi) neuroevolution;

(2) swarm intelligence (SI):

(i) particle swarm optimization;
(ii) ant colony optimization;
(iii) bacterial foraging optimization algorithm;
(iv) artificial bee colony;
(v) shuffled frog leaping algorithm;
(vi) glowworm swarm optimization;
(vii) cuckoo search;
(viii) firefly algorithm;
(ix) harmony search;
(x) bat algorithm;
(xi) wolf search;

(3) other algorithms:

(i) artificial immune algorithm;
(ii) artificial neural networks;
(iii) cellular automata;
(iv) cultural algorithm;
(v) membrane computers;
(vi) brain storm optimization;
(vii) ecoinspired evolutionary algorithm;
(viii) invasive weed optimization;
(ix) dolphin echolocation.

Moreover, these bioinspired optimization algorithms have been widely applied to network optimization [4–7], data mining [8–10], production scheduling [11–14], power system [15, 16], pattern recognition [17, 18], robotics applications [19–21] and so on.

All living organisms have lifecycle, either the commonest ants, butterflies, goldfish around us or the uncommon Antarctic penguins, arctic bear or either the ferocious beast or the meek of poultry. Although different organisms have different lifecycle lengths, they all undergo the process from birth to death. When an original life ends, a new life will generate. The biology evolution of nature follows the "cycle relay" pattern, which is a "life and death alternation" cycle process. This process repeated continuously made the endless life on earth, and biologic evolution become more and more perfect.

Inspired by the idea of lifecycle, in 2002, Krink and Løvbjerg introduced a hybrid approach called the lifecycle model that simultaneously applies genetic algorithms (GAs), particle swarm optimization (PSO), and stochastic hill climbing to create a generally well-performing search heuristics [22]. In this model, authors consider candidate solutions and their fitness as individuals, which, based on their recent search progress, can decide to become either a GA individual, a particle of a PSO, or a single stochastic hill climber.

In 2008, Niu et al. proposed a lifecycle model (LCM) to simulate bacterial evolution from a finite population of *Escherichia coli* (*E. coli*) bacteria [23]. In this simulation study, bacterial behaviors (chemotaxis, reproduction, extinction, and migration) during their whole life cycle are viewed as evolutionary operators used to find the best nutrient concentration which is labeled as a potential global solution of the optimization problem.

In 2011, borrowing the biologic lifecycle theory, the Lifecycle-based swarm optimization (LSO) algorithm was proposed for the first time [24]. Then, 7 unimodal unconstrained optimization test functions and constrained optimization test functions as well as engineering problems that include vehicle routing problem (VRP) and vehicle routing problem with Time Windows (VRPTW) were adopted to test LSO algorithm performance [24–26]. The above experiments demonstrate that LSO is a competitive and effective approach. In order to evaluate the LSO performance accurately, this paper uses 23 unconstrained benchmark functions to study the effectiveness and stability of LSO.

The rest of this paper is organized as follows. Sections 2 and 3 describe the proposed Lifecycle-based swarm optimization (LSO) technique. Sections 4 and 5 present and discuss computational results. The last section draws conclusions and gives directions of future work.

2. Lifecycle-Based Swarm Optimization

2.1. Chemotaxis Operator. Based on the current location, the next movement will be towards the better places. The optimal individual of population selects this foraging strategy. Since the optimal forager in the current iteration possesses the greatest energy, so he has the ability to seek the better location which with more nutrient resources than previous location in global search scope. And the seeking mode taken by optimal foraging individual is not the same as the migration method of nonoptimal individual and also is not a simple migration or position moving, but a rather powerful foraging strategy, such as chaos search. The better solution was found directly using chaos variable.

(1) The current optimization variable is denoted by X_0, and its fitness value is $f(X_0)$.

(2) Generate n chaotic variables $(X_1, X_2, \ldots, X_n)$ by logistic mapping:

$$X_{i+1} = 4X_i(1 - X_i), \quad i = 0, 1, 2, \ldots, n-1. \tag{1}$$

(3) Transform the chaotic motion traverse range to optimize variable domain:

$$X_i = B_{\text{lo}} + \left(B_{\text{up}} - B_{\text{lo}}\right) X_i, \quad i = 1, 2, \ldots, n, \tag{2}$$

where B_{up} and B_{lo} are the upper and lower boundary of the search space.

(4) Compute fitness values:

$$\left(f(X_1), f(X_2), \ldots, f(X_n)\right) \text{ of } n \text{ chaotic variables } (X_1, X_2, \ldots, X_n). \tag{3}$$

(5) If $f(X_i)$ is better than $f(X_0)$, then $X_0 \Leftarrow X_i$, $f(X_0) \Leftarrow f(X_i)$.

2.2. Transposition Operator. Individuals of selecting nonsocial foraging strategy will randomly migrate within their own energy scope:

$$\begin{aligned} \mathrm{ub}_i &= \frac{X_p}{X_i} \cdot \Delta, \\ \mathrm{lb}_i &= -\mathrm{ub}_i, \\ \varphi &= r_2 \left(\mathrm{ub}_i - \mathrm{lb}_i\right) + \mathrm{lb}_i, \\ X_{i+1} &= X_i + \varphi, \end{aligned} \tag{4}$$

where φ is the migration distance of X_i, $r_2 \in R^n$ is a normal distributed random number with mean 0 and standard deviation 1, ub_i and lb_i are the search space boundary of the i_{th} individual, and Δ is the range of the global search space.

2.3. Assimilation Operator. Individuals of selecting social foraging strategy will perform assimilation operator. They gain resource directly from the optimal individual in the way of using a random step towards the optimal individual:

$$X_{i+1} = X_i + r_1 \left(X_p - X_i\right), \tag{5}$$

where $r_1 \in R^n$ is a uniform random sequence in the range $(0, 1)$, X_P is the best individual of the current population, X_i is the position of an individual who performs assimilation operator, and X_{i+1} is the next position of this individual.

2.4. Crossover Operator. In LSO, the crossover operator selects single-point crossover method. One crossover point is selected, string from beginning of individual to the crossover point is copied from one parent, and the rest is copied from the second parent.

2.5. Selection Operator. According to "the survival of the fittest" theory and for ensuring a fixed population size LSO takes a certain method which can make some individuals be retained and the others be eliminated. In this algorithm, the selection operator performs elitist selection strategy. A number of individuals with the best fitness values are chosen to pass to the next generation.

2.6. Mutation Operator. In this algorithm, the mutation operator performs dimension-mutation strategy. Every individual $X_i \in R^n$, $X_i = (x_{i1}, x_{i1}, \ldots, x_{in})$, one dimension of an individual who was selected according to the probability will re-location in search space:

$$x_{ij} = \mathrm{rand}\,(1)\,(\mathrm{ub} - \mathrm{lb}) + \mathrm{lb}, \tag{6}$$

where ub and lb are the lower and upper boundary of search space. In the N-dimension search space, the x_{ij} is the position of the j_{th} dimension of the i_{th} individual; value j is in $[1, N]$.

3. Algorithm Description

Lifecycle-based swarm optimization is a population-based search technique, evaluation all individuals fitness value, and establishes an iterative process through implementation of six operators proposed above. Each population is composed of a certain number of individuals and meets the clumped distribution. In each iteration, firstly, all individuals need to select foraging strategy and execute foraging operator based on individual's fitness value and foraging probability generated randomly; then, this is followed by the crossover operation, selection operation, and the mutation operation. Finally, generate the next population which can represent the new solutions. In the optimization process, the optimization operation is random, but the optimize performance shown us are not entirely randomly. It can effectively utilize the historical information to speculate the next solutions, which has the possible of closer to optimum. Such process was repeated from generation to generation and finally converges to the individual and this was the most adaptable process to environments and an optimal solution was obtained. Figure 1 shows LSO algorithm flowchart.

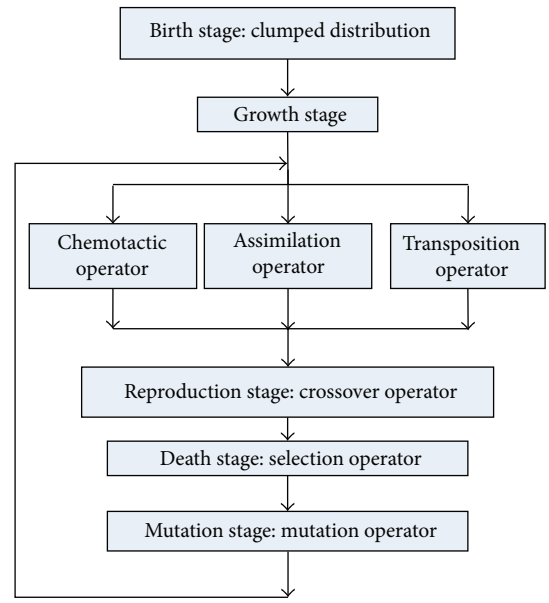

Figure 1: LSO algorithm flowchart.

4. Experiments Setting

4.1. Illustrative Examples. To fully evaluate the performance of the LSO algorithm without bias, we employed 23 benchmark functions which were tested widely in evolutionary computation domain to show the quality solution and the convergence rate [27]. These test functions were listed in appendix. In those functions, functions f_1 to f_7 are unimodal functions, functions f_8 to f_{13} are multimodal functions with many local minima, and functions f_{14} to f_{23} are multimodal functions with few local minima.

In order to verify the efficiency of our approach to settle practical problem and test the goodness of LSO, the mechanical design optimization problem was selected as the testing case, which included pressure vessel and schematic diagram of welded beam problem. These are the hybrid system optimization problems.

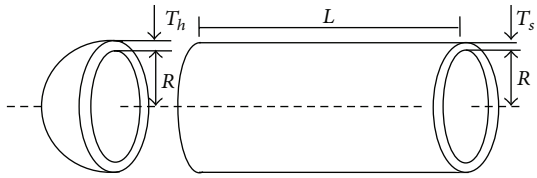

Figure 2: Pressure vessel.

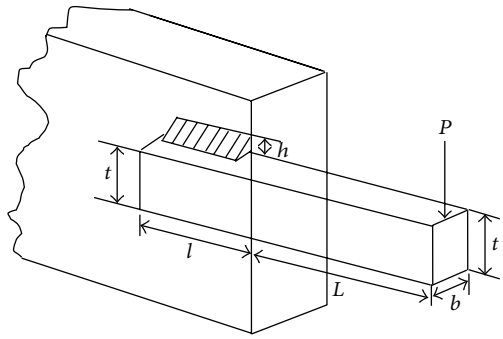

Figure 3: Schematic diagram of welded beam.

(1) Pressure Vessel. As shown in Figure 2, pressure vessel was designed to minimize the total pressure vessel weight. There are four design variables: the shell thickness $T_s = x_1$, the thickness of the head $T_h = x_2$, the inner radius $R = x_3$, and the length of the cylindrical section $L = x_4$. x_1 and x_2 are discrete values which are integer multiples of 0.0625 in and x_3 and x_4 are continuous. The pressure vessel problem is stated as follows:

$$\begin{aligned}
\text{Minimize:}\quad & f(X) = 0.6224X_1X_3X_4 + 1.7781X_2X_3^2 \\
& \quad + 3.1661X_1^2X_4 + 19.84X_1^2X_3 \\
\text{subject to:}\quad & g_1(X) = 0.0193X_3 - X_1 \le 0 \\
& g_2(x) = 0.00954X_3 - X_2 \le 0 \\
& g_3(X) = 1,296,000 - \pi X_3^2 X_4 \\
& \quad - \frac{4}{3}\pi X_3^3 \le 0 \\
& g_4(x) = X_4 - 240 \le 0 \\
& 0.0625 \le X_1, \quad X_2 \le 6.1875, \\
& 10 \le X_3, \quad X_4 \le 200.
\end{aligned} \tag{7}$$

(2) Schematic Diagram of Welded Beam. As shown in Figure 3, schematic diagram of welded beam problem was designed to minimize the total cost of welded beam materials. There are four design variables: the welding thickness $h = X_1$, weld joint length $l = X_2$, the width of the beam $t = X_3$, and the thickness of the beam $b = X_4$. X_1 and X_2 are discrete values which are integer multiples of 0.0625 in and X_3 and X_4 are continuous. The schematic diagram of welded beam problem is stated as follows:

$$\begin{aligned}
\text{Minimize:}\quad & f(x) = 1.1047x_1^2x_2 \\
& \quad + 0.04811x_3x_4(14.0 + x_2) \\
\text{subject to:}\quad & g_1(X) = \tau(X) - 13000 \le 0 \\
& g_2(X) = \sigma(X) - 30000 \le 0 \\
& g_3(X) = x_1 - x_4 \le 0 \\
& g_4(X) = 0.10471x_1^2 + 0.04811x_3x_4 \\
& \quad \times (14.0 + x_2) - 5 \le 0 \\
& g_5(X) = 0.125 - x_1 \le 0 \\
& g_6(X) = \delta(X) - 0.25 \le 0 \\
& g_7(X) = 6000 - P_C(X) \le 0 \\
& \tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \\
& \tau' = \frac{6000}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J}, \\
& M = 6000\left(L + \frac{x_2}{2}\right), \\
& R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \\
& \delta(X) = \frac{504000}{x_3^3x_4}, \\
& J = 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \\
& \sigma(X) = \frac{2.1952}{x_4x_3^2}, \\
& P_c(X) = 64746.022(1 - 0.0282346x_3)x_3x_4^3 \\
& 0.1 \le X_1 \le 2.0, \quad 0.1 \le X_2 \le 10, \\
& 0.1 \le X_3 \le 10, \quad 0.1 \le X_4 \le 2.0.
\end{aligned} \tag{8}$$

4.2. Settings for Involved Algorithms. We compared the optimization performance of LSO with the well-known algorithms: the standard PSO and the standard GA. In 2006, He et al. proposed group search optimizer (GSO) inspired by the scrounging strategies of house sparrows and employed especially animal scanning mechanism [28]. This algorithm appeared to be overpowering compared to the GA, PSO, EP, and ES on 23 benchmark functions used in this paper. Therefore, LSO was also compared with GSO.

The parameter settings of every algorithm were manually tuned. Each of the experiments was repeated 30 runs, and the max iterations in a run $T_{\max} = 3000$. In every run,

Table 1: Results for all algorithms on benchmarks functions f_l to f_7.

Fun. number ($f_{\min}$)	LSO		GSO		PSO		GA	
	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean Best (Rank)	Std (Rank)
1	$1.20E-11$	$4.14E-11$	$9.37E-07$	$4.70E-06$	$3.33E+02$	$1.83E+03$	$1.58E+00$	$6.14E-01$
(0)	(1)	(1)	(2)	(2)	(4)	(4)	(3)	(3)
2	$8.60E-08$	$8.09E-08$	$4.09E-04$	$1.82E-03$	$1.24E+01$	$1.16E+01$	$4.48E-01$	$8.82E-02$
(0)	(1)	(1)	(2)	(2)	(4)	(4)	(3)	(3)
3	$5.95E-09$	$1.40E-08$	$1.27E+02$	$1.59E+02$	$9.93E+03$	$9.78E+03$	$2.88E+03$	$1.15E+03$
(0)	(1)	(1)	(2)	(2)	(4)	(4)	(3)	(3)
4	$4.72E-07$	$5.33E-07$	$3.10E+00$	$1.96E+00$	$1.85E-01$	$8.21E-02$	$3.74E-01$	$6.00E-02$
(0)	(1)	(1)	(2)	(4)	(3)	(3)	(4)	(2)
5	$2.76E+01$	$1.84E-01$	$1.57E+01$	$1.37E+01$	$1.52E+04$	$3.41E+04$	$1.92E+02$	$9.77E+01$
(0)	(2)	(1)	(1)	(2)	(4)	(4)	(3)	(3)
6	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$	$3.44E+02$	$1.82E+03$	$1.53E+00$	$1.38E+00$
(0)	(1)	(1)	(1)	(1)	(3)	(3)	(2)	(2)
7	$5.81E-04$	$5.28E-04$	$5.74E-01$	$2.79E-01$	$3.60E+00$	$7.97E+00$	$5.02E-01$	$3.06E-01$
(0)	(1)	(1)	(3)	(2)	(4)	(4)	(2)	(3)
Total rank	1	1	2	2	4	4	3	3

with the purpose of making the comparison fairly, the initialization populations for all the considered algorithms were generated using the same population which satisfied the normal distribution. The same population size was $S = 50$. The other specific settings for each of the algorithms are described below.

5. Results and Discussion

A lot of experimental data come from printed research papers have shown that PSO and GA can find the optimum of some functions. But in this paper, it becomes powerless. The cause is that the way of generating initialization population is changed from random distribution method to clumped distribution method. In a sense, the clumped distribution is the special form of random distribution. But as stated before, random distribution is rare in reality and clumped distribution is the commonest. So, the finally optimum solution generated via initialization population of random distribution cannot be applied to illustrate the algorithms perform for solving reality and complex optimization problems.

5.1. Unimodal Functions f_1 to f_7. Table 1 presents the optimization results for unimodal functions f_1–f_7 obtained by all algorithms. Obviously, LSO performs best and finds the global optimum or very near optimum in all cases expect function f_5. Functions f_1–f_4 have consistent performance pattern across all algorithms. LSO is the best, GSO is almost good, and PSO and GA failed. Function f_6 is the step function and consists of plateaus, which has one minimum and is discontinuous. It is obvious that finding the optimum solution by LSO and GSO is easy, but it is difficult for PSO. Function f_7 is a noisy quartic function, where random $[0, 1)$ is a uniformly distributed random variable in $[0, 1)$. On this function, LSO can find the exact optimum, whereas other algorithms cannot do so.

Generally speaking, unimodal benchmark functions f_1 to f_7 are relatively easy to be optimized. They were mainly used for testing the convergence rate of algorithm, and the satisfactory accuracy is not a major issue. On this kind of functions, LSO has the best optimization accuracy and the fastest convergence rate. It can converge exponentially fast toward the fitness optimum. This conclusion can be illustrated via Figure 4, which shows the progress of the mean best solutions found by these algorithms over 30 runs for all unimodal functions, expect for function f_5. From Figures 4(a) to 4(f), it can be seen that LSO has the best convergence speed, followed by GSO, GA, and PSO. From the beginning of iterations, the convergence rate of LSO is faster and the convergence curve declined rapidly. At the 500th iteration, LSO has found the optimum solution. Moreover, with increasing the number of iterations, the optimum solution was also approached continuously by LSO at a fast rate. Either the convergence curve of other algorithms is much slower or looks like a horizontal line and seems to stagnate.

5.2. Multimodal Functions with Many Local Minima f_8 to f_{13}. Table 2 presents the optimization results on multimodal functions with many local minima f_8 to f_{13}. These functions were mainly used to test the capability of global seeking optimum and escaping from the local optimum. The quality of the final results is more crucial element. It can also be found from Table 2 that, for 4 out of 6 functions, LSO generated better results than the other four algorithms. The two exceptions are functions f_8 and f_{13}. On f_8, although LSO performs slightly worse than GA, the standard deviation of LSO is highly superior to that of GA. On f_{13}, GSO performs moderately better than LSO. Figure 5 shows the convergence curves of

Table 2: Results for all algorithms on benchmarks functions f_8 to f_{13}.

Fun. number ($f_{\min}$)	LSO		GSO		PSO		GA	
	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)
8	$-1.26E+04$	$1.09E-01$	$-1.23E+04$	$9.01E+02$	$-8.74E+03$	$1.04E+03$	$-1.28E+04$	$4.57E+02$
(-12569.5)	(2)	(1)	(3)	(3)	(4)	(4)	(1)	(2)
9	$1.17E-23$	$3.73E-23$	$2.28E+01$	$8.64E+01$	$1.00E+03$	$6.13E+02$	$1.67E-01$	$1.99E-01$
(0)	(1)	(1)	(3)	(3)	(4)	(4)	(2)	(2)
10	$3.09E-07$	$2.78E-07$	$6.61E-01$	$3.62E+00$	$3.23E+00$	$5.12E+00$	$5.45E-01$	$1.45E-01$
(0)	(1)	(1)	(3)	(3)	(4)	(4)	(2)	(2)
11	$5.02E-04$	$2.75E-03$	$5.36E-02$	$1.59E-01$	$3.04E-02$	$3.97E-02$	$9.58E-01$	$9.83E-02$
(0)	(1)	(1)	(3)	(4)	(2)	(2)	(4)	(3)
12	$2.16E-01$	$5.43E-05$	$2.27E-01$	$6.04E-02$	$1.27E+01$	$1.25E+01$	$4.30E-01$	$1.48E-01$
(0)	(1)	(1)	(2)	(2)	(4)	(4)	(3)	(3)
13	$1.11E-03$	$5.76E-04$	$3.32E-10$	$1.69E-09$	$4.75E-01$	$1.04E+00$	$9.14E-01$	$3.68E-01$
(0)	(2)	(2)	(1)	(1)	(3)	(4)	(4)	(3)
Total rank	1	1	2	3	4	4	3	2

Table 3: Results for all algorithms on benchmarks functions f_{14} to f_{23}.

Fun. number ($f_{\min}$)	LSO		GSO		PSO		GA	
	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)	Mean best (Rank)	Std (Rank)
14	$9.98E-01$	$8.56E-13$	$9.98E-01$	$2.26E-16$	$9.98E-01$	$1.13E-16$	$9.98E-01$	$2.80E-07$
(1)	(1)	(2)	(1)	(1)	(1)	(1)	(1)	(3)
15	$5.61E-03$	$3.37E-03$	$5.72E-03$	$6.12E-03$	$1.09E-02$	$1.08E-02$	$2.16E-02$	$1.40E-02$
($3.075E-4$)	(1)	(1)	(2)	(2)	(3)	(3)	(4)	(4)
16	$-1.03E+00$	$1.04E-08$	$-1.03E+00$	$5.07E-16$	$-1.03E+00$	$6.78E-16$	$-1.03E+00$	$1.19E-04$
(-1.0316)	(1)	(2)	(1)	(1)	(1)	(1)	(1)	(3)
17	0.3984	$1.20E-02$	0.4012	$1.90E-02$	0.4396	$3.73E-02$	0.4001	$6.80E-02$
(0.398)	(1)	(1)	(3)	(2)	(4)	(3)	(2)	(4)
18	$3.00E+00$	$1.42E-07$	$3.00E+00$	$6.35E-14$	$3.00E+00$	$1.33E-15$	$3.91E+00$	$4.94E+00$
(3)	(1)	(3)	(1)	(1)	(1)	(2)	(2)	(4)
19	$-3.86E+00$	$2.75E-09$	$-3.86E+00$	$1.71E-15$	$-3.86E+00$	$1.44E-03$	$-3.84E+00$	$1.41E-01$
(-3.86)	(1)	(2)	(1)	(1)	(1)	(3)	(2)	(4)
20	$-3.27E+00$	$5.94E-02$	$-3.29E+00$	$5.56E-02$	$-3.20E+00$	$1.61E-01$	$-3.29E+00$	$5.56E-02$
(-3.32)	(2)	(2)	(1)	(1)	(3)	(3)	(1)	(1)
21	$-1.02E+01$	$2.90E-07$	$-6.45E+00$	$3.01E+00$	$-7.11E+00$	$2.99E+00$	$-6.95E+00$	$3.17E+00$
(-10)	(1)	(1)	(4)	(2)	(2)	(3)	(3)	(4)
22	$-9.96E+00$	$1.69E+00$	$-7.24E+00$	$3.09E+00$	$-8.45E+00$	$2.61E+00$	$-6.05E+00$	$3.00E+00$
(-10)	(1)	(1)	(3)	(4)	(2)	(2)	(4)	(3)
23	$-9.64E+00$	$2.32E+00$	$-6.70E+00$	$3.29E+00$	$-8.39E+00$	$2.93E+00$	$-6.23E+00$	$3.21E+00$
(-10)	(1)	(1)	(3)	(4)	(2)	(2)	(4)	(3)
Total rank	1	1	2	2	2	3	3	4

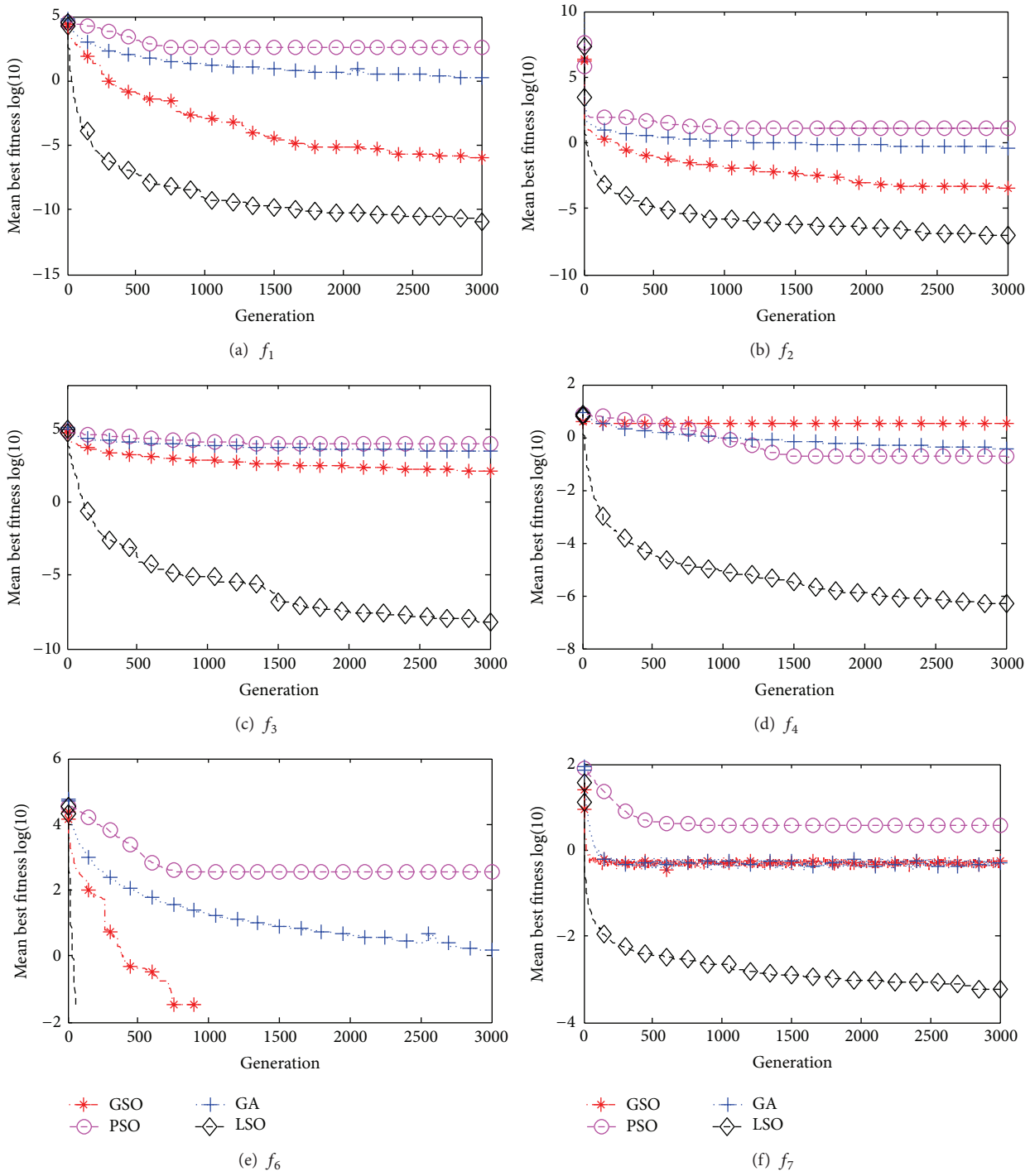

FIGURE 4: Convergence results of unimodal functions f_1, f_2, f_3, f_4, f_6, and f_7.

functions f_8 to f_{13}. LSO converges very fast to good values near the optimum. In summary, the search performance of the four algorithms tested here can be ordered as LSO > GSO > GA > PSO.

5.3. Multimodal Functions with Few Local Minima f_{14} to f_{23}. Functions f_{14} to f_{23} are multimodal functions with few local minima and possess rather unique features, which can verify the adaptation of algorithms to the different optimization environment. Table 3 presents the optimization results for functions f_{14} to f_{23}. It can be concluded from Table 3 that the order of the search performance of these four algorithms is LSO > GSO > PSO > GA.

For these ten functions, in terms of testing the indicators, LSO was ranked the first on functions f_{15}, f_{17}, f_{21}, f_{22}, and f_{23}. For example, the problem f_{21} shown in Figure 6(a) has

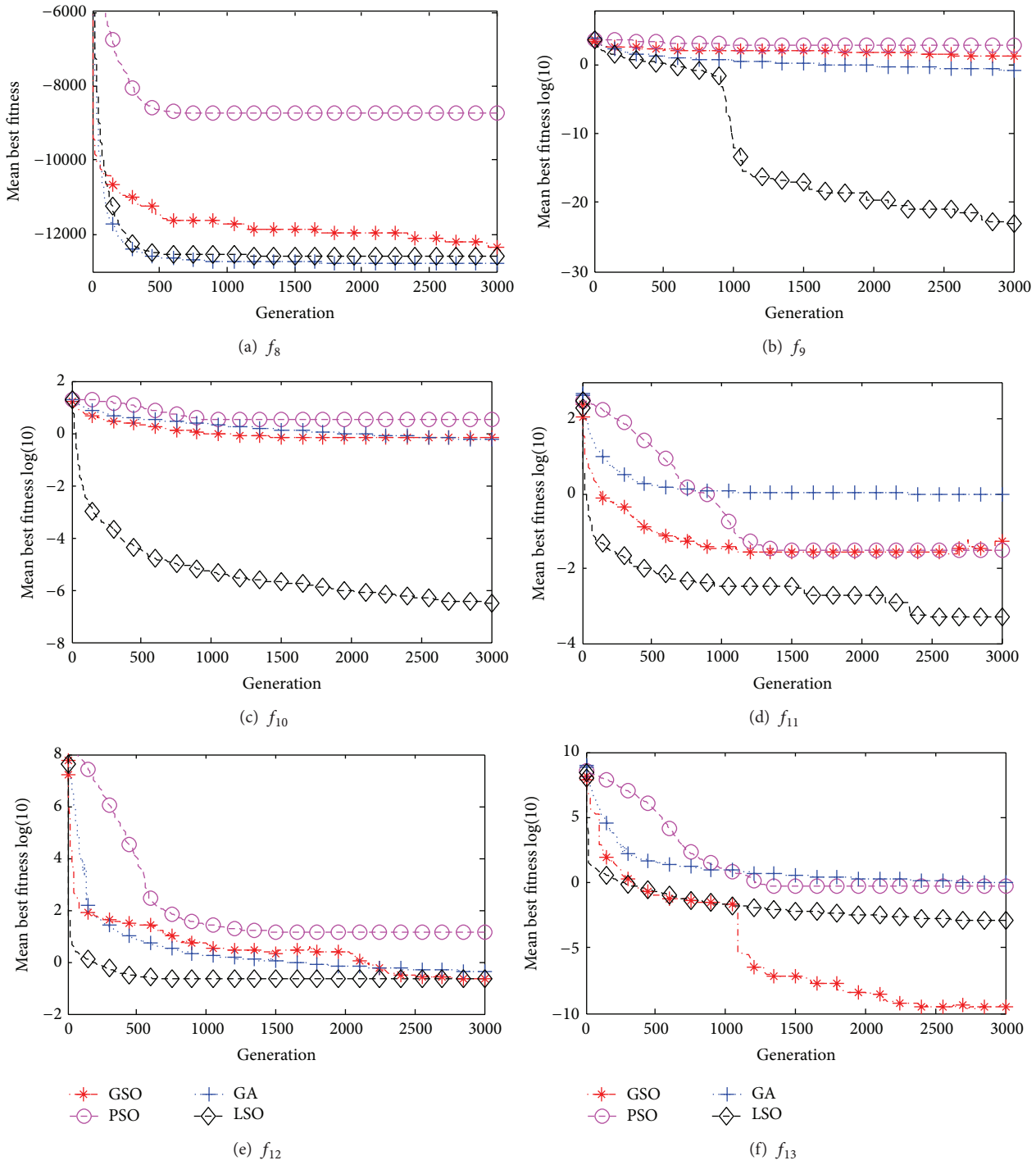

Figure 5: Convergence results of functions f_8, f_9, f_{10}, f_{11}, f_{12}, and f_{13}.

five extremes; the bottom point at the deepest hole is the global optimal position and the other holes are deceptive. Figure 6(b) shows convergence results of four algorithms. All algorithms have been quickly in the early iterations. But the GSO, PSO, and GA stagnate before finding the global optimum, and LSO stagnates until it finds it. At the beginning of the searching, there is a number of promising "fox holes," so the convergence rate of these algorithms is fast. But after a short period, owing to lacking the ability of jumping out of the local extreme, the solutions obtained by GSO, PSO, and GA fall into the "fox holes" deeply, and the evolutionary curve tends to stop. The optimization tactics make LSO escape from the deceptive region and migrate towards the global one. The properties of function f_{22} and f_{23} are similar to that of function f_{21}. Figure 7 shows the same convergence results on functions f_{22} and f_{23}.

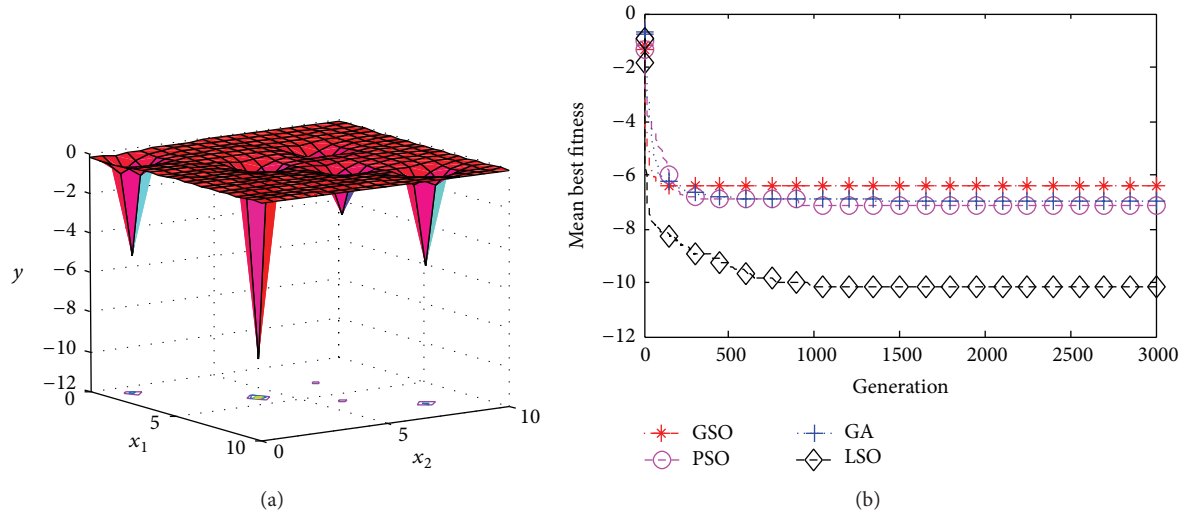

FIGURE 6: Function f_{21}. (a) Function graphs with a dimension of 2 and (b) convergence results of all algorithms.

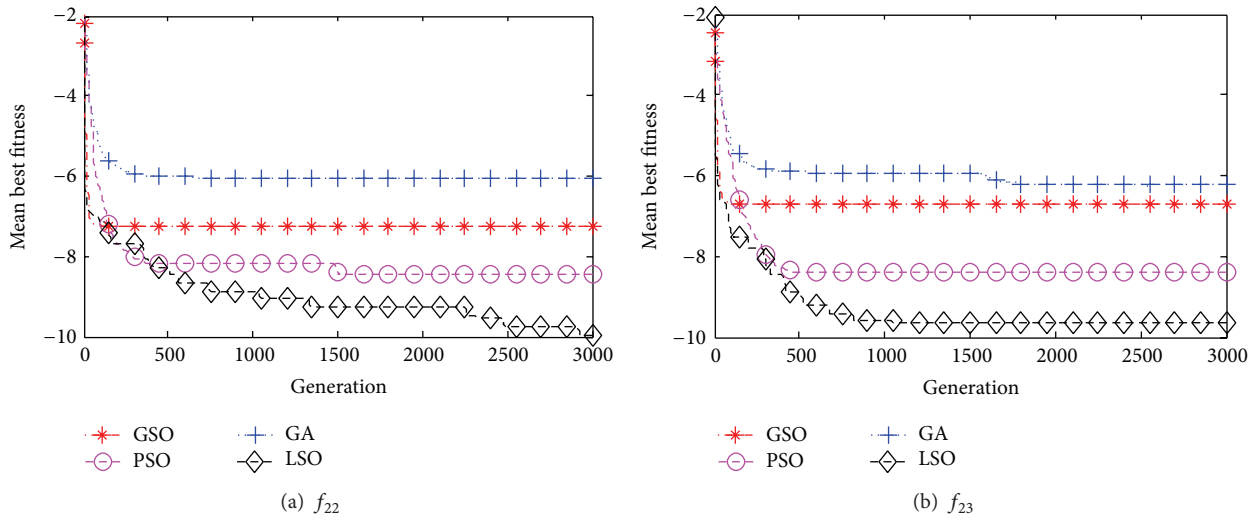

FIGURE 7: Convergence results of functions f_{22} and f_{23}.

Functions f_{14} and f_{16} are all easy problems, and all algorithms can find the exact optimum solutions. On functions f_{18} and f_{19}, LSO, GSO, and PSO yield the exact optimum, while GA yielded the approximate optimum. All algorithms come very close to the global optimum on f_{20}. Figures 8(a) and 8(c) show the convergence results on functions f_{14} and f_{18}. Moreover, we can see that LSO has the fastest convergence speed from Figures 8(b) and 8(d).

5.4. Mechanical Design Optimization Problem. Mechanical design optimization plays an important role in engineering and manufacturing enterprises. In this field, one of the most difficult parts encountered is constraints handling and optimization variables. First, on the test results of proposed algorithm, the best feasible value on these two problems is 6059.72 and 1.7107, respectively. The best feasible solution found by our approach is better than those solutions found by other techniques, listed in other literature [29]. In addition, the standard LSO employed a penalty function to preserve feasibility of the encountered solutions. This proposed method is relatively simple compared to other algorithms introduced to solve constraint the problem, such as the multiobjective evolutionary method, the collaborative evolutionary particle swarm optimization algorithm, dynamic penalty function method, annealing penalty function method, information feedback adaptive penalty function method, multilayer social culture algorithm, and combination of global and local topology particle swarm algorithm.

6. Conclusions

This paper proposed a novel optimization algorithm, LSO, which is based on biologic lifecycle theory. Based on these features of lifecycle, LSO designed six optimization operators:

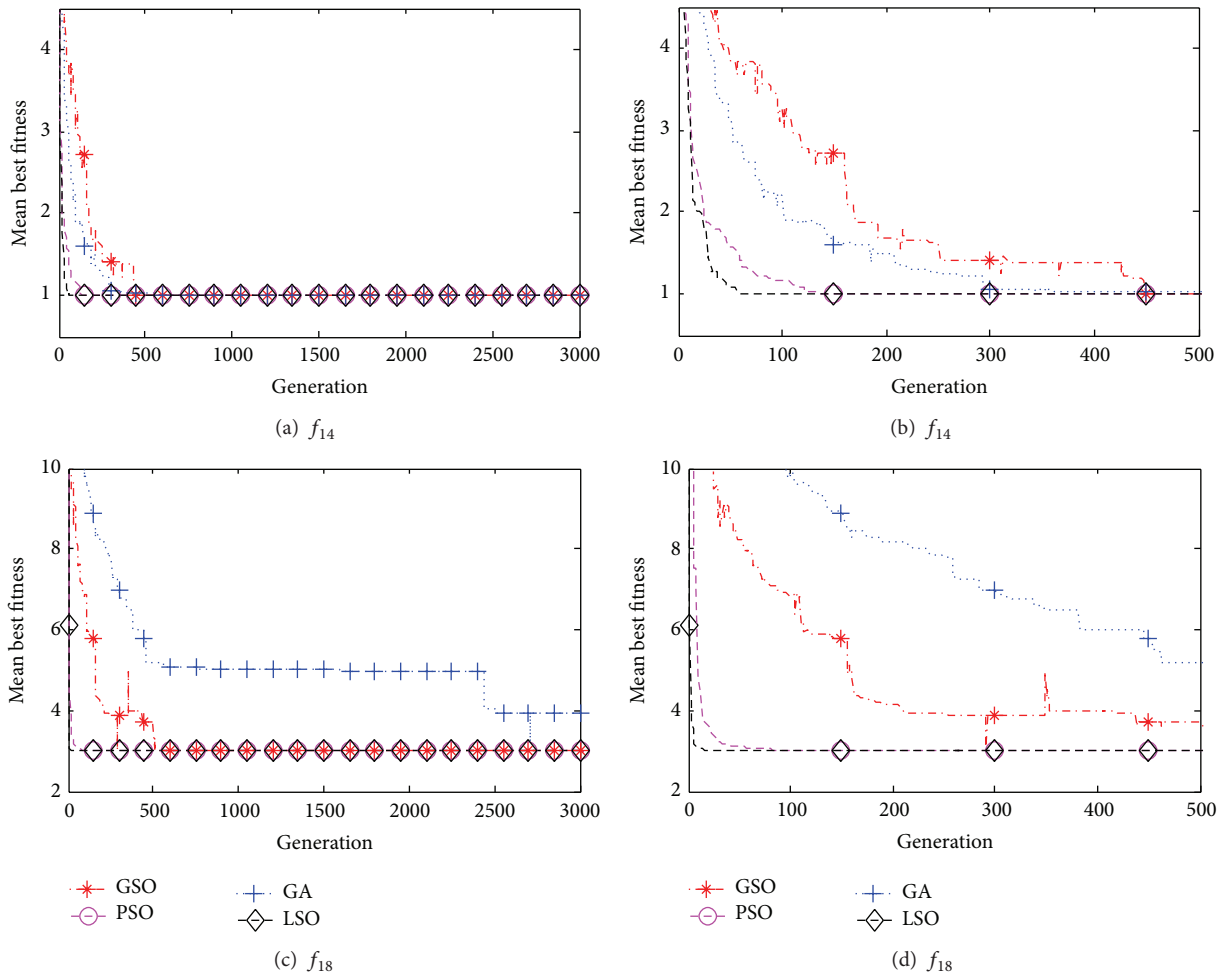

Figure 8: Convergence results of functions f_{14} and f_{18}.

chemotactic, assimilation, transposition, crossover, selection, and mutation. Population is the basic unit of biologic existence. Clumped distribution of population spatial is the commonest pattern. This paper borrows the clumped distribution pattern to generate initialization population. A set of 23 unconstrained benchmark functions and mechanical design optimization problems have been used to test LSO in comparison with GSO, PSO and GA, respectively.

It is worth mentioning that LSO cannot find the optimum on function f_5. Function f_5 is a nonconvex function; its global minimum is inside a long, narrow, parabolic shaped flat valley. However, even though the valley is easy to find, convergence to the global minimum is difficult. So our future work would study how to make LSO has the ability of moving quickly along the narrow valley in the local area to the objective function minimum. For instance, gradient-based method is incorporated in the late stage of optimization.

As part of our future work, LSO also could be studied and tested on real-world problems, such as location problem of manufacturing systems, network routing problem of computer engineering, parameter identification problem of industrial engineering, electrical engineering problem, aerospace engineering problem, and bioengineering problem.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper or financial conflict of interests between the authors and the commercial identity.

Acknowledgments

This project is supported by the National Natural Science Foundation of China (Grant nos. 61174164, 51205389, 61105067, 71001072, 71271140, and 71240015), the National Natural Science Foundation of Liaoning province of China (Grant no. 201102200), the General Research Project of Liaoning province of China (Grant no. L2012392), and the Natural Science Foundation of Guangdong Province (Grant nos. S2012010008668 and 9451806001002294).

References

[1] V. L. Avila, *Biology: Investigating Life on Earth*, Jones and Bartlett, Boston, Mass, USA, 1995.

[2] P. H. Raven and G. B. Johnson, *Biology*, Hill Companies, Boston, Mass, USA, 5th edition, 1999.

[3] G. W. Flake, *The Computational Beauty of Nature: Computer Explorations of Fractals, Complex Systems, and Adaptation*, MIT Press, Cambridge, Mass, USA, 1998.

[4] L. B. Ribeiro and M. F. de Castro, "BiO4SeL: a bio-inspired routing algorithm for sensor network lifetime optimization," in *Proceedings of the 17th International Conference on Telecommunications (ICT '10)*, pp. 728–734, Doha, Qatar, April 2010.

[5] K. B. Swain, S. S. Solanki, and A. K. Mahakula, "Bio inspired cuckoo search algorithm based neural network and its application to noise cancellation," in *International Conference on Signal Processing and Integrated Networks (SPIN '14)*, pp. 632–635, Noida, India, February 2014.

[6] K. Saleem, N. Fisal, and J. Al-Muhtadi, "Empirical studies of bio-inspired self-organized secure autonomous routing protocol," *IEEE Transactions on Sensors Journal*, vol. 14, no. 7, pp. 2232–2239, 2014.

[7] A. Khalili, A. Rastegarnia, and M. K. Islam, "Bio-inspired cooperative algorithm for distributed source localization with mobile nodes," in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3515–3518, 2013.

[8] J.-H. Qu, Z.-Z. Shao, and X.-Y. Liu, "PSO clustering algorithm based on cooperative evolution," *Journal of Donghua University*, vol. 27, no. 2, pp. 285–288, 2010.

[9] L. Goel, D. Gupta, and V. K. Panchal, "Hybrid bio-inspired techniques for land cover feature extraction: a remote sensing perspective," *Applied Soft Computing Journal*, vol. 12, no. 2, pp. 832–849, 2012.

[10] Y. Hendrawan and H. Murase, "Bio-inspired feature selection to select informative image features for determining water content of cultured Sunagoke moss," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14321–14335, 2011.

[11] R. Chaukwale and S. S. Kamath, "A modified ant colony optimization algorithm with load balancing for job shop scheduling," in *Proceedings of the 15th International Conference on Advanced Computing Technologies (ICACT '13)*, pp. 1–5, Rajampet, India, September 2013.

[12] K. Sun, G. Yang, and C. Pan, "Solving hot rolling scheduling problem by a new population-based extremal optimization algorithm," in *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications*, pp. 1189–1193, September 2010.

[13] R. M. Chen and Y. A. Chen, "Heuristics based particle swarm optimization for solving vehicle routing problems," in *Proceedings of the International Symposium on Computer, Consumer and Control*, pp. 360–363, 2014.

[14] F. A. Toader, "Production scheduling by using ACO and PSO techniques," in *Proceedings of the International Conference on Development and Application Systems*, pp. 170–175, 2014.

[15] A. Mozaffari, M. Azimi, and M. Gorji-Bandpy, "Ensemble mutable smart bee algorithm and a robust neural identifier for optimal design of a large scale power system," *Journal of Computational Science*, vol. 5, no. 2, pp. 206–223, 2014.

[16] W. Peres, E. J. Oliveira, J. A. P. Filho, D. N. Arcanjo, I. C. Silva, and L. W. Oliveira, "Power system stabilizers tuning using bio-inspired algorithm," in *Proceedings of the IEEE Grenoble Conference PowerTech (POWERTECH '13)*, pp. 1–5, IEEE, Grenoble, France, June 2013.

[17] E. Cuevas, V. Osuna-Enciso, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "Multi-threshold segmentation based on artificial immune systems," *Mathematical Problems in Engineering*, vol. 2012, Article ID 874761, 20 pages, 2012.

[18] Z. Ding, J. Tang, X. Zhang, and B. Luo, "A local elitism based membrane evolutionary algorithm for point pattern matching," *Advances in Intelligent Systems and Computing*, vol. 212, pp. 873–882, 2013.

[19] M. Turduev, G. Cabrita, M. Kirtay, V. Gazi, and L. Marques, "Experimental studies on chemical concentration map building by a multi-robot system using bio-inspired algorithms," *Autonomous Agents and Multi-Agent Systems*, vol. 28, no. 1, pp. 72–100, 2014.

[20] F. Cordella, L. Zollo, E. Guglielmelli, and B. Siciliano, "A bio-inspired grasp optimization algorithm for an anthropomorphic robotic hand," *International Journal on Interactive Design and Manufacturing*, vol. 6, no. 2, pp. 113–122, 2012.

[21] O. Castillo, H. Neyoy, J. Soria, M. García, and F. Valdez, "Dynamic fuzzy logic parameter tuning for ACO and its application in the fuzzy logic control of an autonomous mobile robot," *International Journal of Advanced Robotic Systems*, vol. 10, article 47, 2013.

[22] T. Krink and M. Løvbjerg, "The lifecycle model: combining particle swarm optimization, genetic algorithms and hillclimbers," in *Lecture Notes in Computer Science*, vol. 2439, pp. 621–630, 2002.

[23] B. Niu, Y. L. Zhu, X. X. He, H. Shen, and Q. H. Wu, "A lifecycle model for simulating bacterial evolution," *Neurocomputing*, vol. 72, no. 1–3, pp. 142–148, 2008.

[24] H. Shen, Y. Zhu, L. Jin, and H. Guo, "Lifecycle-based swarm optimization method for constrained optimization," *Journal of Computers*, vol. 6, no. 5, pp. 913–922, 2011.

[25] H. Shen, B. Niu, Y. L. Zhu, and H. N. Chen, "Optimization algorithm based on biology life cycle theory," in *Intelligent Computing Theories and Technology*, vol. 7996 of *Lecture Notes in Computer Science*, pp. 561–570, Springer, Berlin, Germany, 2013.

[26] Z. W. Yin, H. Shen, Y. Deng, and M. Zhang, "Lifecycle-based swarm optimization for constrained problem of engineering," *Applied Mechanics and Materials*, vol. 281, pp. 710–714, 2013.

[27] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[28] S. He, Q. H. Wu, and J. R. Saunders, "A novel group search optimizer inspired by animal behavioral ecology," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1272–1278, 2006.

[29] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, 2002.