Seven-spot Ladybird Optimization for large-scale problems and engineering design optimization

Peng Wang, Shuai Huang, Zhouquan Zhu
Northwestern Polytechnical University, Xi'an 710072, P.R. China
wangpeng305@nwpu.edu.cn

Abstract

Increasing attention is being paid to solving engineering design optimization problems which are generally large-scale or nonlinear or constrained. In this paper, the Seven-spot Ladybird Optimization (SLO) algorithm is presented to solve these problems. The SLO is inspired by recent discoveries on the foraging behavior of a seven-spot ladybird. This paper presents the basic concepts and main steps of the SLO and demonstrates its efficiency. The performance of the SLO is compared with some popular meta-heuristic algorithms by using five different dimensional classical benchmark functions. The simulation results indicate that the proposed algorithm has the ability to get out of a local minimum and can be efficiently used for multivariable, multimodal function optimization. Moreover, three well-known constrained engineering problems are solved by using SLO and numerical results show the efficiency of this proposed method in practical computations.

1 Introduction

In general, large-scale optimization problems are divided into unconstrained optimization problems and constrained optimization problems. In its simplest form, an unconstrained *D*-dimensional real-parameter optimization problem can be formulated as a search for the optimal parameter vector X, which minimizes an objective function $f(X)$ for all $X \in \Omega$, where Ω is a non-empty large finite set serving as the domain of the search. The optimization task is complicated by the existence of non-linear objective functions with multiple local minima [Nasir *et al.*, 2012].

Constrained optimization problems are mathematical programming problems frequently encountered and difficult to solve in applications such as engineering design, VLSI design, structural optimization, location and allocation problems [Lu and Chen, 2008]. As a result, it is important for both academicians and practitioners to solve constrained optimization problems efficiently and effectively. Constrained optimization problems with n variables and m constraints can be described as follows:

min $f(X)$

subject to:

$g_j(X) \leqslant 0$, for $j=1, \cdots, q$

$h_j(X)=0$, for $j=q+1, \cdots, m$

$l_i \leqslant x_i \leqslant u_i$, for $i=1, , \cdots, n$

where $X=(x_1, x_2, \cdots, x_n)$ is the solution vector; $f(X)$ is the objective function to be minimized; $g_j(X) \leqslant 0$ and $h_j(X)=0$ are inequality and equality constraints, respectively; l_i and u_i are the upper and lower bounds of x_i.

Due to the complexity and unpredictability of large-scale optimization problems, especially for constrained optimization problems, a general deterministic solution is hard to find within a reasonable amount of computation time. In recent years, several evolutionary algorithms which are capable of finding the high-precision near-optimal solutions to the large-scale optimization problems have been proposed. Some of the optimization algorithms that are in wide use these days are Genetic Algorithm (GA) [Goldberg, 1989], Simulated Annealing [Kirkpatrick *et al.*, 1983], Tabu Search [Glover and Laguna, 1997], Covariance Matrix Adaptation Evolution Strategies [Hansen and Ostermeier, 1996], Teaching–Learning-Based Optimization [Rao *et al.*, 2011, 2012], Harmony Search [Yildiz, 2008], Immune Algorithm [Yildiz, 2009], Differential Evolution [Zhang and Sanderson, 2009], Particle Swarm Optimization (PSO) [Eberhart and Kennedy, 1995], Ant-Colony Optimization [García *et al.*, 2009], Artificial Bee Colony (ABC) algorithm [Akay and Karaboga, 2012] and some hybrid algorithms [Liu *et al.*, 2010; Zahara and Kao, 2009]. According to the famous No-Free-Lunch theorem for optimization [Wolpert and Macready, 1997], all non-repeating search algorithms have the same mean performance when averaged uniformly over all possible objective functions. But it does not exclude the possibility that some certain algorithms will obtain better results for some certain objective functions. So the requirement to develop new optimization algorithms increasingly continues.

This paper introduces a novel biologically inspired meta-heuristic algorithm called seven-spot ladybird optimization (SLO). SLO is inspired by the foraging behavior of a seven-spot ladybird. In this paper, the performance of the SLO is compared with that of GA, PSO, and ABC by using five different dimensional classical benchmark functions and three well-known constrained engineering problems are solved by using SLO. The results show that SLO has the ability to find the best solution with a comparatively small population size and is suitable for solving optimization problems with lower dimensions. The remainder of this paper is organized as follows: Section 2 presents the foraging behavior of the seven-spot ladybird; Section 3 describes the SLO and the steps in detail; Section 4 discusses the experiments and the results; and Section 5 draws the conclusions.

2 Seven-spot Ladybird in the Nature

Many interesting collective behaviours can be observed in insects such as ant, bee, termite, cockroach etc. These individuals usually perform their actions locally with limited knowledge about the entire system. However, when their local actions are combined, they will exhibit complex collective intelligent behaviors and produce global effects [Wong *et al.*, 2008].

The seven-spot ladybird, also known as *Coccinella septempunctata*, has attracted the interest of a growing number of professional entomologists because of its ecological effectiveness and social behaviors. Recent studies have shown that seven-spot ladybirds are more social than we believe them to be [Hodek *et al.*, 1993; Dixon, 2000]. Like most insects, pheromone is the paramount way they communicate and share information with others. Some chemical ecologies of the seven-spot ladybirds, with special attention to semiochemicals involved in social communication and foraging behaviors, have been reviewed in [Pettersson *et al.*, 2005].

The latest researches have shown that the environmental levels of seven-spot ladybirds can be classified into prey, patches, and habitats (Figure 1) [Hodek *et al.*, 1984; Ferran and Dixon, 1993]. This provides a framework for discussing the foraging behaviors of seven-spot ladybirds. In Figure 1, movement between prey within aggregates of aphids is referred to as intensive search which is slow and sinuous. Movement between aggregates within a patch is referred to as extensive search which is relatively linear and fast. Movement between patches is called dispersal and movement from patches to hibernation is called migration.

Seven-spot ladybirds locate their prey via extensive search and then switch to intensive search after feeding. While searching for its prey, a seven-spot ladybird holds its antennae parallel to its searching substratum and its maxillary palpi perpendicular to the substratum. The ladybird vibrates its maxillary palpi and turns its head from side to side. The sideward vibration can increase the area wherein the prey may be located.

How seven-spot ladybirds decide when to leave a patch for another, also known as dispersal, remains unclear. Several authors suggested that beetles decide to leave when the capture rate falls below a critical value or when the time since the last aphid was captured exceeds a certain threshold [Hemptinne *et al.*, 1992; Kindlmann and Dixon, 1993].

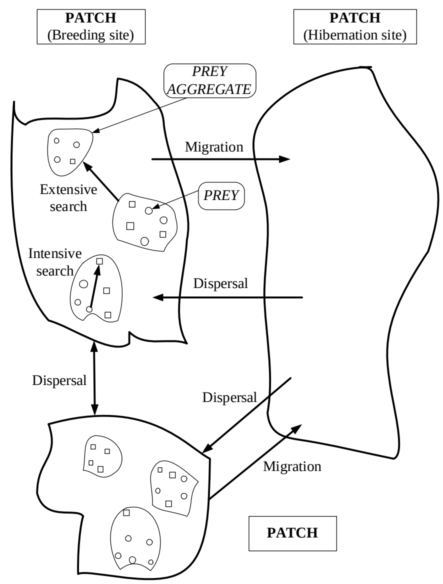

Figure 1. Diagram illustrating how a ladybird might perceive its environment and forage for resources

3 Seven-spot Ladybird Optimization Algorithm

This section describes the proposed seven-spot ladybird optimization (SLO) algorithm, which simulates the foraging behavior of seven-spot ladybirds to solve large scale optimization problems. The main steps are given below:

Step 1: Dividing Patches

Suppose that the search space (environment) is a D-dimensional space. The *i-th* dimensional space is divided into n_i subspaces, and the whole dimensional space is divided into $n = \prod n_i$ subspaces (patches).

Step 2: Initializing Population

Suppose that each seven-spot ladybird is treated as a point in a D-dimensional patch. The *i-th* ladybird is represented as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$, where X_i is a latent solution to the optimized question.

If m is the number of seven-spot ladybirds initialized with random positions in a patch, then the population size of the seven-spot ladybirds is N, $N = m \times n$.

Step 3: Calculating fitness

The fitness values of the seven-spot ladybirds were calculated.

Step 4: Choosing the best ladybird

The current position of each ladybird was compared with its previous best position (*sbest*). If the current value is better than the previous one, then *sbest* = current value.

The current best position of all the ladybirds in a patch was compared with their previous best position (*lbest*). If the current value is better than the previous one, then *lbest* = current value.

The current best position of all the ladybirds in the population was compared with their previous best position (*gbest*). If the current value is better than the previous one, then *gbest* = current value.

Step 5: Dispersal

In the SLO, if a position does not improve in a predetermined number of cycles, then a new position is produced in the patch where *gbest* exists, replacing the abandoned position. The new position is produced near the *gbest* to share the information of the best ladybird in the whole particle. The value of the predetermined number of cycles (*limit*) is an important control parameter in the SLO.

If the abandoned position is X_i and $j \in \{1, 2, \cdots, D\}$, then the seven-spot ladybird discovers a new position X_i' as follows:

$$x_{i,j}' = x_{gbest,j} + \phi w \tag{1}$$

where w is the neighborhood space of *gbest* and ϕ is a random number between [-1, 1].

Step 6: Updating positions

The position of a ladybird is updated associated with its previous movement. If a ladybird has done extensive search, then the position of the ladybird is changed as follows:

$$V = c * r_1 * (S_i(t) - X_i(t)) + \varepsilon_1 \tag{2}$$

$$X_i(t+1) = X_i(t) + V, |V| \le V_{\max} \tag{3}$$

After intensive search, a ladybird switches to extensive search. The position is updated according to the following equations:

$$V = c * r_2 * (L_i(t) - X_i(t)) + \varepsilon_2 \tag{4}$$

$$X_i(t+1) = X_i(t) + V, |V| \le V_{\max} \tag{5}$$

In Equation (2) and (4), r_1 and r_2 are two random numbers uniformly distributed from 0 to 1 and the positive constant c is used for adjusting the search step and search direction in each iteration. In Equation (3) and (5), the velocities of the ladybirds in each dimension are limited to the maximum velocity V_{max}, which decides the search precision of the ladybirds in a solution space. If V_{max} is too high, then the ladybirds will possibly fly over the optimal solution. However, if the V_{max} is too low, then the ladybirds will fall into the local search space and have no method to carry on with the global search. Typically, V_{max} is set as follows:

$$V_{\max} = 0.2(ub - lb) \tag{6}$$

where ub and lb are the upper and lower bounds of each patch, respectively.

From equations above, we can see that the velocity updating rule is composed of three parts. The first part, known as **intensive search**, is inspired by the slow and sinuous movements of ladybirds. The second part, known as **extensive search**, is derived from the relatively linear and fast movement behavior of ladybirds. The third part imitates the **sideward vibration** of ladybirds to increase the search area where the potential solution may exist. The parameter ε_1 and ε_2 are usually set as relatively small random numbers.

Step 7: Inspecting termination condition

If the termination condition is satisfied, i.e., the SLO has achieved the maximum iteration number, then the SLO is terminated; otherwise, it returns to Step 3.

4 Experiments and Results

4.1 Large-scale unconstrained optimization problems

In order to verify the efficiency and analyze the performance of Seven-spot Ladybird Optimization algorithm, five typical benchmark test functions are given as following:

Griewank Function

$$f_2(X) = \sum_{t=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10), \quad x_i \in [-15, 15] \tag{7}$$

The value of Griewank function is 0 at its global minimum (0, 0, ⋯ , 0), and it has a product term that introduces interdependence among its variables. The aim is to overcome the failure of the techniques that optimize each variable independently.

Rastrigin Function

$$f_3(X) = \sum_{t=1}^{D} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2, \quad x_i \in [-15, 15] \tag{8}$$

The value of Rastrigin function is also 0 at its global minimum (0, 0, ⋯, 0), and its contour is made up of a large number of local minima whose value increases with the distance to the global minimum.

Rosenbrock Function

$$f_3(X) = \sum_{t=1}^{D} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2, \quad x_i \in [-15, 15] \tag{9}$$

The value of Rosenbrock function is 0 at its global minimum (1, 1, …, 1). The global optimum is inside a long, narrow, parabolic-shaped flat valley. Since it is difficult to converge to the global optimum, this problem is repeatedly used to test the performance of the optimization algorithms.

Ackley Function

$$f_4(X) = 20 + e - 20e^{\left(-0.2\sqrt{\frac{1}{D}\sum_{t=1}^{D} x_i^2}\right)} - e^{\frac{1}{D}\sum_{t=1}^{D}\cos(2\pi x_i)}, \quad x_i \in [-32.786, 32.786] \tag{10}$$

The value of Ackley function is 0 at its global minimum (0, 0, ⋯, 0), and it has an exponential term that covers its surface with numerous local minima. A search strategy must combine the exploratory and exploitative components efficiently to obtain good results for the Ackley function.

Schwefel Function

$$f_5(X) = D * 418.9829 + \sum_{t=1}^{D} -x_i \sin(\sqrt{|x_i|}), \quad x_i \in [-500, 500] \tag{11}$$

The value of Schwefel function is 0 at its global minimum (420.9867, 420.9867, ⋯, 420.9867), and it is composed of a great number of peaks and valleys. The function has a second best minimum far from the global minimum where many search algorithms are trapped.

Results of the SLO algorithm were taken for D = 5, 10 and 30 and were compared to those of the GA, PSO and ABC algorithms. Common control parameters of the algorithms are population size and number of maximum generation. In the experiments, the population size was 50, and maximum generations were 750, 1000 and 1500 for Dimensions 5, 10 and 30, respectively. Other parameters employed for GA, PSO and ABC are consistent with [Karaboga and Basturk, 2007; Dipti and Seow, 2003]. In SLO, every dimension is divided into 2 equal parts, so there are totally 2^D patches. In each patch, the initial population of ladybirds is set to 20. The parameter *limit* is 100 and *w* is 1. The parameter *c* in Equation (2) and (4) decreases linearly from 10 to 2. The sideward vibration ε_1 is Rand*10e-4 and ε_2 is Rand*10e-8.

Each of the experiments was repeated 30 times with different random seeds, and the mean, best and standard values produced by different algorithms are presented in Table 1-5. In order to make the comparison clearer, values below e-12 were assumed to be 0.

According to the best function values obtained using the different algorithms with D=5, the SLO can find the global optimization solution with values close to the theoretical solution and has the same search ability as PSO. This indicates the proposed SLO algorithm has the ability to find the best solution with a comparatively small population size. Based on the mean results of all experiments, the proposed SLO has better performance than GA for Griewank function, Ackley function and Schwefel function. However, when dimension is 30, the result of SLO is no better than that of PSO and ABC. Considering the No Free Lunch Theorem, if we compare two searching algorithms with all possible functions, the performance of any two algorithms will be, on average, the same. In this paper, the proposed SLO is suitable for solving optimization problems with lower dimensions.

Algorithm	Dimension	Mean	Best	SD
SLO	5	1.29E-01	7.40E-03	9.12E-02
	10	3.10E-01	2.46E-02	5.29E-01
	30	1.47E+00	6.20E-03	1.79E+00
GA	5	1.22E+01	1.22E+01	1.23E-10
	10	1.46E+01	6.56E+00	4.71E+00
	30	1.41E-02	1.91E-10	2.47E-02
PSO	5	2.37E-02	7.40E-03	1.24E-02
	10	7.69E-02	2.70E-02	3.36E-02
	30	1.32E-02	0.00E+00	1.49E-02
ABC	5	6.88E-04	0.00E+00	2.10E-03
	10	2.80E-03	0.00E+00	4.60E-03
	30	6.35E-04	0.00E+00	3.50E-03

Table 1 Results of the Griewank function

Algorithm	Dimension	Mean	Best	SD
SLO	5	0.00E+00	0.00E+00	0.00E+00
	10	2.28E+01	9.98E-01	1.47E+01
	30	3.61E+02	2.31E+02	5.90E+01
GA	5	6.30E-01	2.07E-09	6.12E-01
	10	7.96E-01	3.33E-08	8.43E-01
	30	3.08E+00	2.51E-06	2.37E+00
PSO	5	6.63E-02	0.00E+00	2.52E-01
	10	1.73E+00	0.00E+00	1.17E+00
	30	2.90E+01	1.79E+01	8.56E+00
ABC	5	0.00E+00	0.00E+00	0.00E+00
	10	0.00E+00	0.00E+00	0.00E+00
	30	0.00E+00	0.00E+00	0.00E+00

Table 2 Results of the Rastrigin function

Algorithm	Dimension	Mean	Best	SD
SLO	5	1.33E+00	1.62E-08	8.78E-01
	10	1.69E+01	6.70E+00	2.36E+01
	30	1.61E+04	2.56E+03	9.16E+03
GA	5	5.08E-02	9.20E-03	2.01E-02
	10	7.06E-01	1.06E-01	5.61E-01
	30	2.34E+01	7.50E-02	2.23E+01
PSO	5	2.20E-01	3.61E-04	3.95E-01
	10	2.61E+00	3.98E-02	1.38E+00
	30	4.11E+01	1.12E+01	2.78E+01
ABC	5	5.13E-02	4.90E-03	5.67E-02
	10	5.22E-02	1.90E-03	5.11E-02
	30	6.07E-02	3.64E-04	7.53E-02

Table 3 Results of the Rosenbrock function

Algorithm	Dimension	Mean	Best	SD
SLO	5	0.00E+00	0.00E+00	0.00E+00
	10	9.34E-02	0.00E+00	3.61E-01
	30	9.27E-01	0.00E+00	2.33E+00
GA	5	1.91E-05	8.74E-07	1.47E-05
	10	2.89E-05	4.12E-06	1.22E-05
	30	7.65E-05	5.40E-05	1.22E-05
PSO	5	0.00E+00	0.00E+00	0.00E+00
	10	0.00E+00	0.00E+00	0.00E+00
	30	1.71E-08	1.59E-09	2.11E-08
ABC	5	0.00E+00	0.00E+00	0.00E+00
	10	0.00E+00	0.00E+00	0.00E+00
	30	0.00E+00	0.00E+00	0.00E+00

Table 4 Results of the Ackley function

Algorithm	Dimension	Mean	Best	SD
SLO	5	3.25E+02	6.36E-05	1.73E+02
	10	1.25E+03	7.50E+02	2.79E+02
	30	5.87E+03	4.71E+03	7.61E+02
GA	5	2.08E+03	2.08E+03	5.23E-11
	10	4.15E+03	4.15E+03	9.49E-11
	30	1.25E+04	1.25E+04	7.32E-10
PSO	5	2.96E+02	6.36E-05	1.31E+02
	10	5.67E+02	2.37E+02	1.67E+02
	30	2.86E+03	1.60E+03	4.04E+02
ABC	5	6.36E-05	6.36E-05	0.00E+00
	10	1.27E-04	1.27E-04	0.00E+00
	30	3.10E-03	3.82E-04	1.42E-02

Table 5 Results of the Schwefel function

4.2 Constrained mechanical design optimization problems

To handle these constraints, many different approaches have been proposed. The penalty function method is the most popular technique due to its simple principle. It converts a constrained optimization problem to an unconstrained one by adding a penalty item to the objective function [Kitayama *et al.*, 2006]. In these experiments, the penalty function method is employed to handle three well-known constrained engineering problems: pressure vessel, tension/ compression spring and speed reducer.

Problem 1: pressure vessel

The first example is minimization of the total cost comprising of material, forming and welding costs of a cylindrical vessel as shown in Figure 2. The four design variables are x_1(thickness T_S of the shell), x_2(thickness T_H of the head), x_3(inner radius R) and x_4(length L of the cylindrical section of the vessel, not including the head). x_1 and x_2 are to be in integral multiples of 0.0625 inch which are the available thicknesses of rolled steel plates. The radius x_3 and the length x_4 are continuous variables.

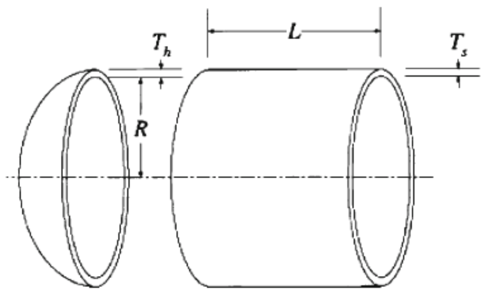

Figure 2. The pressure vessel problem

$$
\begin{aligned}
\min_X\ & f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
& \quad + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
\textit{subject to}\ & g_1(X): -x_1 + 0.0193x_3 \le 0 \\
& g_2(X): -x_2 + 0.00954 \le 0 \\
& g_3(X): -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\
& g_4(X): x_4 - 240 \le 0
\end{aligned} \tag{12}
$$

where $X=(x_1, x_2, x_3, x_4)^T$. The ranges of the design parameters are $0 \leqslant x_1, x_2 \leqslant 99$, $10 \leqslant x_3, x_4 \leqslant 200$.

Problem 2: tension/ compression spring

The tension/compression problem deals with the minimization of the weight of the tension/compression spring shown in Figure 3, subject to constraints on the minimum deflection, shear stress, surge frequency, diameter and design variables. The design variables are the wire diameter, $d(x_1)$, the mean coil diameter, $D(x_2)$, and the number of active coils, N (x_3).

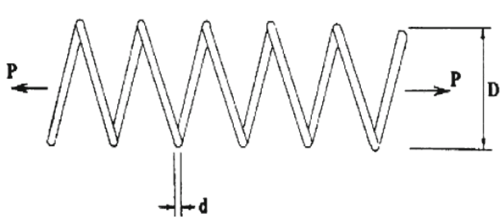

Figure 3. The tension/compression spring problem

The problem is formulated as:

$$
\begin{aligned}
\min_X\ & f(X) = (N+2)Dd^2 \\
\textit{subject to}\ & g_1(X): 1 - \frac{D^3N}{71785d^4} \le 0 \\
& g_2(X): \frac{4D^2 - dD}{12566(Dd^3 - d^4)} + \frac{1}{5108d^2} - 1 \le 0 \\
& g_3(X): 1 - \frac{140.45d}{D^2N} \le 0 \\
& g_4(X): \frac{D+d}{1.5} - 1 \le 0
\end{aligned} \tag{13}
$$

where $X=(d, D, N)^T$, $0.05 \leqslant d \leqslant 2.0$, $0.25 \leqslant D \leqslant 1.3$, $2.0 \leqslant N \leqslant 15.0$.

Problem 3: speed reducer

The aim of the speed reducer design shown in Figure 4 is to minimize the weights of the speed reducer subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shafts. Design parameters of the speed reducer problem, the face width (b), module of teeth (m), number of teeth in the pinion (z), length of the first shaft between bearings (l_1), length of the second shaft between bearings (l_2) and the diameter of the first shaft (d_1) and second shaft (d_2) correspond to $x_1, x_2, \cdots, x_7$, respectively. Some optimization algorithms have been reported to have difficulties in finding the feasible space and is an example of a mixed integer programming problem. The third variable (number of teeth) is of integer value while all other variables are continuous.

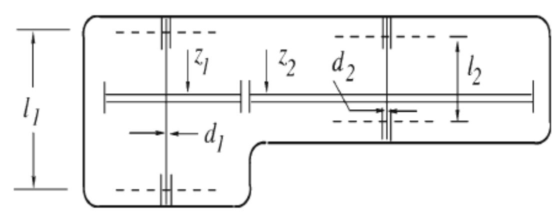

Figure 4. The speed reducer problem

$$
\begin{aligned}
\min_X\ & f(X) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) \\
& \quad - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) \\
\textit{subject to}\ & g_1(X): \frac{27}{x_1x_2^2x_3} - 1 \le 0 \qquad g_2(X): \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0 \\
& g_3(X): \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0 \qquad g_4(X): \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0 \\
& g_5(X): \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times10^6\right)^{1/2}}{110.0x_6^3} - 1 \le 0 \\
& g_6(X): \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 157.5\times10^6\right)^{1/2}}{85.0x_7^3} - 1 \le 0 \\
& g_7(X): \frac{x_2x_3}{40} - 1 \le 0 \qquad g_8(X): \frac{5x_2}{x_1} - 1 \le 0 \\
& g_9(X): \frac{x_1}{12x_2} - 1 \le 0 \qquad g_{10}(X): \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \\
& g_{11}(X): \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0
\end{aligned} \tag{14}
$$

where $2.6 \leqslant x_1 \leqslant 3.6$, $0.7 \leqslant x_2 \leqslant 0.8$, $17 \leqslant x_3 \leqslant 28$, $7.3 \leqslant x_4 \leqslant 8.3$, $7.8 \leqslant x_5 \leqslant 8.3$, $2.9 \leqslant x_1 \leqslant 3.9$, $5.0 \leqslant x_1 \leqslant 5.5$.

In the experiments, results of the SLO are compared with the Society and Civilization Algorithm (SCA), the Evolution Strategy (μ+λ−ES) and the ABC algorithms [Akay and Karaboga, 2012]. The values of the algorithm-specific control parameters are given in [Akay and Karaboga, 2012]. For SLO, the parameters are set the same as those described in section 4.1. Discrete variables were handled by truncating the real value to its closest integer value. We conducted 30 independent experiments for each problem. The best values were recorded to show the ability of an algorithm to find the optimal. Table 6 presents the reference optimal values given in [Akay and Karaboga, 2012] and the best values obtained using SLO. Parameter values of the best solutions to the three optimization problems are given in Table 7, 8 and 9.

Problem	SLO	SCA	(μ+λ)-ES	ABC
Pressure vessel	6106.8097	6171.0000	6059.7143	6059.7143
Tension/ compression spring	0.0127	0.0127	0.0127	0.0127
Speed reducer	2897.4658	2994.7442	2996.3481	2997.0584

Table 6 Best results of the SLO, SCA, (μ+λ)-ES and ABC algorithms

	SLO	SCA	(μ+λ)-ES	ABC
x_1	0.8757	0.8125	0.8125	0.8125
x_2	0.5	0.4375	0.4375	0.4375
x_3	45.3754	41.9768	42.0984	42.0984
x_4	139.8608	182.2845	176.6366	176.6366
$f(x)$	0.8757	6171.0000	6059.7143	6059.7143

Table 7 Parameter values for pressure vessel problem

	SLO	SCA	(μ+λ)-ES	ABC
x_1	0.0505	0.0521	0.0528	0.0517
x_2	0.3293	0.3682	0.3849	0.3582
x_3	13.1136	10.6484	9.8077	11.2038
$f(x)$	0.0127	0.0127	0.0127	0.0127

Table 8 Parameter values for tension/compression spring problem

	SLO	SCA	(μ+λ)-ES	ABC
x_1	3.6000	3.5000	3.4999	3.4999
x_2	0.7500	0.7000	0.6999	0.7
x_3	17	17	17	17
x_4	7.8000	7.3276	7.3000	7.3000
x_5	8.3000	7.7153	7.8000	7.8000
x_6	3.4	3.3503	3.3502	3.3502
x_7	5.0	5.2867	5.2867	5.2878
$f(x)$	2897.4658	2994.7442	2996.3481	2997.0584

Table 9 Parameter values for speed reducer problem

Depending on the results in Table 6, for pressure vessel problem, the best solutions are produced by (μ+λ)-ES and ABC algorithms. On tension/compression spring problem, the SLO, SCA, (μ+λ)-ES and ABC algorithms deliver equal performances. As for the speed reducer problem with seven parameters, SLO algorithm is more successful. In general, the performance of these four algorithms are roughly the same. But considering the population size, the proposed SLO algorithm is more efficient.

5 Conclusions

This paper investigated the foraging behaviors of seven-spot ladybirds and proposed a novel biologically inspired meta-heuristic algorithm called SLO. The performance of SLO algorithm was compared with those of GA, PSO and ABC optimization algorithms by using five numerical benchmark functions. The simulated results show that SLO has the ability to find the best solution with a comparatively small population size and is suitable for solving optimization problems with lower dimensions. Moreover, the proposed SLO algorithm has been applied to solve three constrained engineering problems. The above experiments indicate that the proposed SLO algorithm is an effective method for handling constraints in practical computations. However, it should be noted that the proposed algorithm is not as efficient as ABC. According to the No Free Lunch Theorem [Wolpert and Macready, 1997], "Any elevated performance over one class of problems is offset by performance over another class." Future studies will focus on improving the SLO.

References

[Akay and Karaboga, 2012] Bahriye Akay and Dervis Karaboga. Artificial bee colony algorithm for large-scale problems and engineering design optimization. *Journal of Intelligent Manufacturing*. 23(4):1001-1014, August 2012.

[Dipti and Seow, 2003] Srinivasan Dipti and Tian Hon Seow. Particle swarm inspired evolutionary algorithm (PS-EA) for multiobjective optimization problems. In *The 2003 Congress on Evolutionary Computation*, pages 2292-2297, Canberra, ACT, Australia, December 2003. IEEE.

[Dixon, 2000] A.F.G. Dixon. *Insect predator-prey dynamics: ladybird beetles and biological control*. Cambridge University Press, New York, 2000.

[Eberhart and Kennedy, 1995] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995*, pages 39-43, Nagoya, Japan, Oct 1995. IEEE.

[Ferran and Dixon, 1993] Andre Ferran and Anthony F. G. Dixon. Foraging behaviour of ladybird larvae (Coleoptera: Coccinellidae). *European Journal of Entomology*. 90:383-402,

[García *et al.*, 2009] Salvador García, Daniel Molina, Manuel Lozano, and Francisco Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*. 15(6):617-644, December 2009.

[Glover and Laguna, 1997] Fred Glover and Manuel Laguna. *Tabu search*. Kluwer Academic Publishers, Boston, MA, July 1997.

[Goldberg, 1989] David Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, January 1989.

[Hansen and Ostermeier, 1996] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation 1996*, pages 312-317, Nagoya, Japan, May 1996. IEEE.

[Hemptinne *et al.*, 1992] J. L. Hemptinne, A. F. G. Dixon, and J. Coffin. Attack strategy of ladybird beetles (Coccinellidae): factors shaping their numerical response. *Oecologia*. 90(2):238-245, May 1992.

[Hodek *et al.*, 1984] I. Hodek, S. Chakrabarti, and M. Rejmanek. The effect of prey density on food intake by adultCheilomenes sulphurea [Col.: Coccinellidae]. *Entomophaga*. 29(2):179-184, June 1984.

[Hodek *et al.*, 1993] Ivo Hodek, Gabriel Iperti, and Magdalena Hodkova. Long-distance flights in Coccinellidae (Coleoptera). *European Journal Of Entomology*. 90(4):403-414, December 1993.

[Karaboga and Basturk, 2007] Dervis Karaboga and Bahriye Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of global optimization*. 39(3):459-471, November 2007.

[Kindlmann and Dixon, 1993] Pavel Kindlmann and Anthony F.G. Dixon. Optimal foraging in ladybird beetles (Coleoptera: Coccinellidae) and its consequences for their use in biological control. *European Journal of Entomology*. 90(4):443-450, 1993.

[Kirkpatrick *et al.*, 1983] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simmulated annealing. *science*. 220(4598):671-680, May 1983.

[Kitayama *et al.*, 2006] S. Kitayama, M. Arakawa, and K. Yamazaki. Penalty function approach for the mixed discrete nonlinear problems by particle swarm optimization. *Structural and Multidisciplinary Optimization*. 32(3):191-202, September 2006.

[Liu *et al.*, 2010] Hui Liu, Zixing Cai, and Yong Wang. Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Applied Soft Computing*. 10(2):629-640, March 2010.

[Lu and Chen, 2008] Haiyan Lu and Weiqi Chen. Self-adaptive velocity particle swarm optimization for solving constrained optimization problems. *Journal of global optimization*. 41(3):427-445, July 2008.

[Nasir *et al.*, 2012] Md Nasir, Swagatam Das, Dipankar Maity, Soumyadip Sengupta, Udit Halder, and P. N. Suganthan. A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization. *Information Sciences*. 209(0):16-36, November 2012.

[Pettersson *et al.*, 2005] Jan Pettersson, Velemir Ninkovic, Robert Glinwood, Michael A. Birkett, and John A. Pickett. Foraging in a complex environment-semiochemicals support searching behaviour of the seven spot ladybird. *European Journal of Entomology*. 102(3):365, 2005.

[Rao *et al.*, 2011] R. V. Rao, V. J. Savsani, and D. P. Vakharia. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*. 43(3):303-315, March 2011.

[Rao *et al.*, 2012] R. V. Rao, V. J. Savsani, and D. P. Vakharia. Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences*. 183(1):1-15, January 2012.

[Wolpert and Macready, 1997] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*. 1(1):67-82, April 1997.

[Wong *et al.*, 2008] Li-Pei Wong, Malcolm Yoke Hean Low, and Chin Soon Chong. A Bee Colony Optimization Algorithm for Traveling Salesman Problem. In *Second Asia International Conference on Modeling & Simulation*, pages 818-823, Kuala Lumpur, Malaysia, May 2008. IEEE.

[Yildiz, 2008] Ali Rıza Yildiz. Hybrid Taguchi-harmony search algorithm for solving engineering optimization problems. *International Journal of Industrial Engineering: Theory, Applications and Practice*. 15(3):286-293, 2008.

[Yildiz, 2009] Ali Rıza Yildiz. A novel hybrid immune algorithm for global optimization in design and manufacturing. *Robotics and Computer-Integrated Manufacturing*. 25(2):261-270, April 2009.

[Zahara and Kao, 2009] Erwie Zahara and Yi-Tung Kao. Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems. *Expert Systems with Applications*. 36(2, Part 2):3880-3886, March 2009.

[Zhang and Sanderson, 2009] Jingqiao Zhang and A. C. Sanderson. JADE: Adaptive Differential Evolution With Optional External Archive. *IEEE Transactions on Evolutionary Computation*. 13(5):945-958, October 2009.