Journal of
Applied & Computational Mathematics

Hosseini, J Appl Computat Math 2017, 6:1
DOI: 10.4172/2168-9679.1000344

Research Article Open Access

Laying Chicken Algorithm: A New Meta-Heuristic Approach to Solve Continuous Programming Problems

Eghbal Hosseini*

Philosophy Doctor of Operational Research and Optimization, Department of Mathematics, UAE

Abstract

A concept for the optimization of continuous programming problems using an approach based on behavior of laying chicken, to produce chicken, is introduced. Laying chicken algorithm (LCA) is used for solving different kinds of linear and non-linear programming problems. The presented approach gives efficient and feasible solutions in an appropriate time which has been evaluated by solving test problems. The comparison between LCA and both of meta-heuristic and classic approaches are proposed.

Keywords: Laying chicken algorithm; Meta-heuristic approaches; Optimization problems

Introduction

In the recent decades, optimization and computer scientists have been designing several algorithms based on behavior of animals and insects because the natural systems are very efficient. Swarm Intelligence (SI) was introduced in 1989 as a novel approach in the global optimization [1]. Ant Colony Optimization (ACO) was proposed to solve discrete problems as a new meta-heuristic optimizer [2]. Particle Swarm Optimization (PSO) introduced for solving continuous programming problems [3]. These algorithms have been found acceptable solutions to optimization problems. Therefore, the meta-heuristic algorithms, such as Artificial Bee Colony Algorithm [4], krill herd algorithm [5], Bat Algorithm (BA) [6], social spider optimization [7], Chicken Swarm Optimization (CSO) [8], firefly algorithm [9] have attracted by many researchers.

This paper suggests a novel optimizer for optimization of linear and non-linear programming problems in continuous state. The approach has been discovered in simulating of a natural bi-inspiring model. The paper introduces the laying chicken algorithm concept, strongly discusses the steps of its extension from bi-inspiring simulation model to optimizer. Finally, by proposed numerical results of linear and non-linear test problems, it is easy to see that the simulated model has been succeeded.

Laying chicken algorithm is related to two principle concepts. In general, LCA ties to artificial agent or artificial life obviously, and to, laying fishes, laying turtles, laying snakes and laying chickens in particular (not SI behavior). It comes from both evolutionary programming and genetic algorithms. In this paper, relationships between LCA and above concepts are obvious.

Proposed laying chicken algorithm by the author includes an easy natural theory and concept, and performance of steps can be displayed in some lines of MATLAB code. It needs just an array, to store feasible solutions, and initial mathematical factors. So it has an acceptable computational complexity in both of memory and time. Initial testes have realized the enforcement to be feasible and effective using different classes of problems. In the rest of paper, performances of steps and their MATLAB code will be presented. Finally use of approach to solve several kinds of problems, such as constraint and unconstraint programming problems in different states, is discussed.

Simulation of Laying Chicken Behavior

The hens and their eggs are a great source of food as one of the most extensive tame animals [8]. This paper focuses on behavior of laying hen and answer of this question: "how does she convert the egg to the chicken?" In this paper, same as eggs to the chicken, the feasible solutions have been changed to the optimal solution. In fact, each egg displays a feasible solution in continuous programming problem and a chicken describes optimal solution in the problem.

Farmers use a false egg sometimes to encourage hens to stay in the nest. Because hens often prefer in the same location and not empty nest to lay, in fact they try to do that in the nest that already contain eggs. This is a great idea to create an initial feasible solution and to generate first population near that.

Pheromone of ants in ant colony, individual members or global best in particle swarm optimization, crossover or combination of genes in genetic, are the fundamental concepts of some of the meta-heuristic algorithms. Here hens try to warm their eggs; this concept is base to development of laying chicken algorithm. Same as temperature of eggs objective function of solutions will be improved. Rotation of eggs is the next concept which will be simulated by little change of solutions.

The Laying Chicken Algorithm Concept

The laying chicken algorithm optimizer may the best proposed using describe its conception development. As mentioned, LCA comes from laying hen as an original naturel event, so in this section the main concept of LCA and its relationship with the bio-inspiring event is discussed.

The initial solution

The simulated approach already was written based on two main

***Corresponding author:** Hosseini E, Philosophy Doctor of Operational Research and Optimization, Department of Mathematics, UAE, Tel: 21 2332 0009; E-mail: eghbal_math@yahoo.com

Received March 14, 2016; **Accepted** March 17, 2017; **Published** March 21, 2017

Citation: Hosseini E (2017) Laying Chicken Algorithm: A New Meta-Heuristic Approach to Solve Continuous Programming Problems. J Appl Computat Math 6: 344. doi: 10.4172/2168-9679.1000344

concepts: initial solution and population. Same as the first egg in the nest, the initial feasible solution was necessary. So it has been created randomly. If it is not feasible, a loop in the MATLAB code is repeated to create a feasible one. Initial solution for some optimization problems is created in MATLAB are as follows:

The initial population

In the first iteration, initial population of solutions has been created near the initial feasible solution as possible. In fact, the next factor of the simulation defines "the initial neighborhood," an n-dimensional neighborhood of R^n, this is defined as follows:

$$\|X-Y\| \le k \quad (1)$$

or

$$\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\ldots+(x_n-y_n)^2} \le k$$

Which, X is initial solution, Y is an n-dimensional vectors and k is a positive constant. Here the initial population of eggs has been created randomly in the possible nearest neighborhood of the initial solution.

Each member of initial population has to be in this neighborhood of the initial solution. We try to generate solutions very near the initial solution. This is because hens usually like to stay in their nest with their eggs. In fact, they prefer to convert their eggs to chicken than other animal eggs. Figure 1 shows 500 eggs (feasible solutions) near to the initial solution for a given problem with k=1 in R^2 (Figure 1).

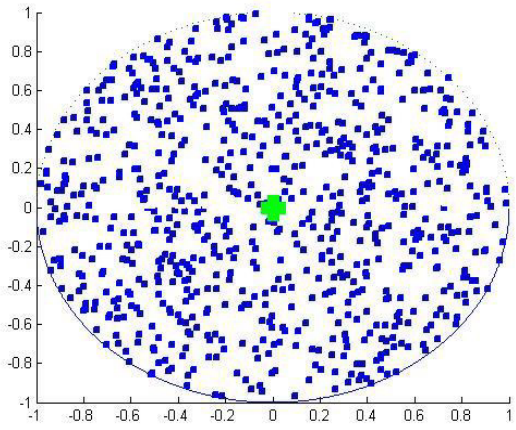

Figure 1: 500 eggs (blue points) have been created near initial solution (green point).

The algorithm will be more efficient when k be very small. This is because it does not miss many solutions near initial solution small k.

Improving of population

Each solution in population, which its objective function is not better than objective function of initial solution, should be changed in direction initial solution while it will be better than initial solution. In fact, value of particles have been changed in direction vector which connects its and the initial solution. These solutions have been modified as follows:

$$x_{j+1}=x_j+\alpha d_{j0} \quad (2)$$

Which, d_{j0} is the vector from xj to x_0 and $f(x_j)< f(x_0)$, $0 \le \alpha \le 2k$ in maximization problems.

All states of α have been described in Table 1 and according to that, the feasible interval for α as follows:

$$0 \le \alpha \le 2k \quad (3)$$

It is easy to show that $\alpha \to 0$ does not change solutions very well, so interval $0<\alpha<k/4$ has been removed and the following is the best:

$$k/4<\alpha \le 2k \quad (4)$$

But according to the gradient theorem in Figure 2a objective function of blue points are not better than initial solution (large red point) and small red points are better than it, in a given problem that its optimal solution is in right hand side of the initial solution. So interval of α has been modified as follows:

$$k \le \alpha \le 2k \quad (5)$$

This is because the author wants to move all blue solutions in Figure 2a in direction initial solution such that they will be better than it. Green points in Figure 2b are these solutions after their movement. By this stage all solution in population will be better than initial solution Figure 2c. The best solution in this iteration will be initial solution in the next iteration. So in the next population and after this step, all solution will be better than the best solution of current population. This is the main idea of the algorithm which every population is better than previous population. Pseudo code of this stage has been shown in Figure 3.

Changing the solutions

The last trait of the simulated method has been inspired from rotation of the eggs by the hen. She rotates the eggs three or four times every day. In this stage except the best solution, all member

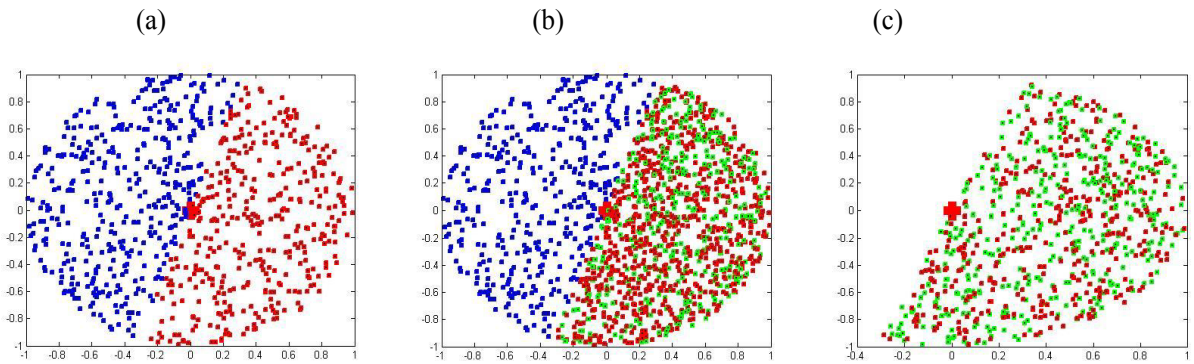

Figure 2: Process of changing solutions which are not better than initial solution in direction it.

```
begin
    Objective function f(x), x = (x1, ..., xd)T
    Generate initial population of eggs xi (i = 1, 2, ..., n) near x0
    Temperature Ei at xi is determined by f(xi)
        for j = 1 : n (all n eggs)
            while (Ej < E0), warm egg j;
            end
            Evaluate new solutions and update temperature
        end
end
```

Figure 3: Pseudo code of improving of solution.

```
begin
        for j=1 : n (all n eggs)
          if (f(xj) < objective function of the best solution), change xj a
              little;
          end
            Evaluate new solutions and update temperature(objective
            function)
        end
end
```

Figure 4: MATLAB code of changing the solutions.

of population have been little changed as follows. ε is a given small positive number.

Some of solutions have been selected randomly and changed as follows:

$(xi+1,yi+1)=(x_i \mp \varepsilon, yi \mp \varepsilon)$

Each solution j which $x_j < x_{best}$ has been changed as follows:

$(x_j+1, y_j+1)=(x_j+\varepsilon, y_j)$

Each solution k which $x_k > x_{best}$ has been changed as follows:

$(x_{k+1}, y_{k+1})=(x_k-\varepsilon, y_k)$

At each iteration, the best solution has been saved and other solutions selected near that in the next population. There are two states for current stage: If this stage creates the better solution from the best one (best in this iteration), it will substitute the best and in the next iteration solutions should be selected near that. Otherwise the best solution will not change. In fact by this stage, the best solution will be better or not changed. Figure 4 shows code of this step.

This stage is useful because it causes to generate more random solutions except the best solution. In fact, the algorithm has more choices to select the best solution by more random solutions.

Steps of the algorithm

The main steps of the algorithm in R^2 as follows:

- The initial feasible solution (x_0,y_0), is created. Number of iteration, N, and an arbitrary small positive number, ε_1 are given.
- Initial population near (x_0,y_0), is generated.
- Each solution in step 2, which its objective function is not better than (x_0,y_0), should be changed in direction (x_0,y_0) and found the best solution (x_{best},y_{best})
- All solutions, except the best one, have been very little changed.
- Objective function of solutions and the best solution is updated. Let $(x_0,y_0)=(x_{best},y_{best})$, go back to step 2.
- If $|f(x_{ibest})-f(x_{(i+1)best})|<\varepsilon 1$ or the number of iteration is more than M the algorithm will be finished, x_{ibest}, $x_{(i+1)best}$ are the best solutions in two consecutive generations. Figure 5 shows the process of the algorithm to gain optimal solution from a given feasible solution. Explanation of Figure 5 as follows: initial solution is generated in eqn (1), Red point is the optimal solution and the green point is an arbitrary feasible solution. Eqn (2) shows first population near initial solution with k=1, red points are better than the initial solution and blue points are not. Blue points move in eqn (3) and convert to green points which are better than initial solution. The algorithm continues with eqn (4). All solutions except the best solution have been little changed according eqn (5). Next population will be created near the best solution.

The process of the algorithm in R^3 has been shown in Figure 6.

Convergence

Convergent parameter set includes initial solution, small positive number ε and constants k and α. BBA is run several times to determine convergence rate and convergent parameter set of the algorithm. The convergence rate is top if various results are gained by more

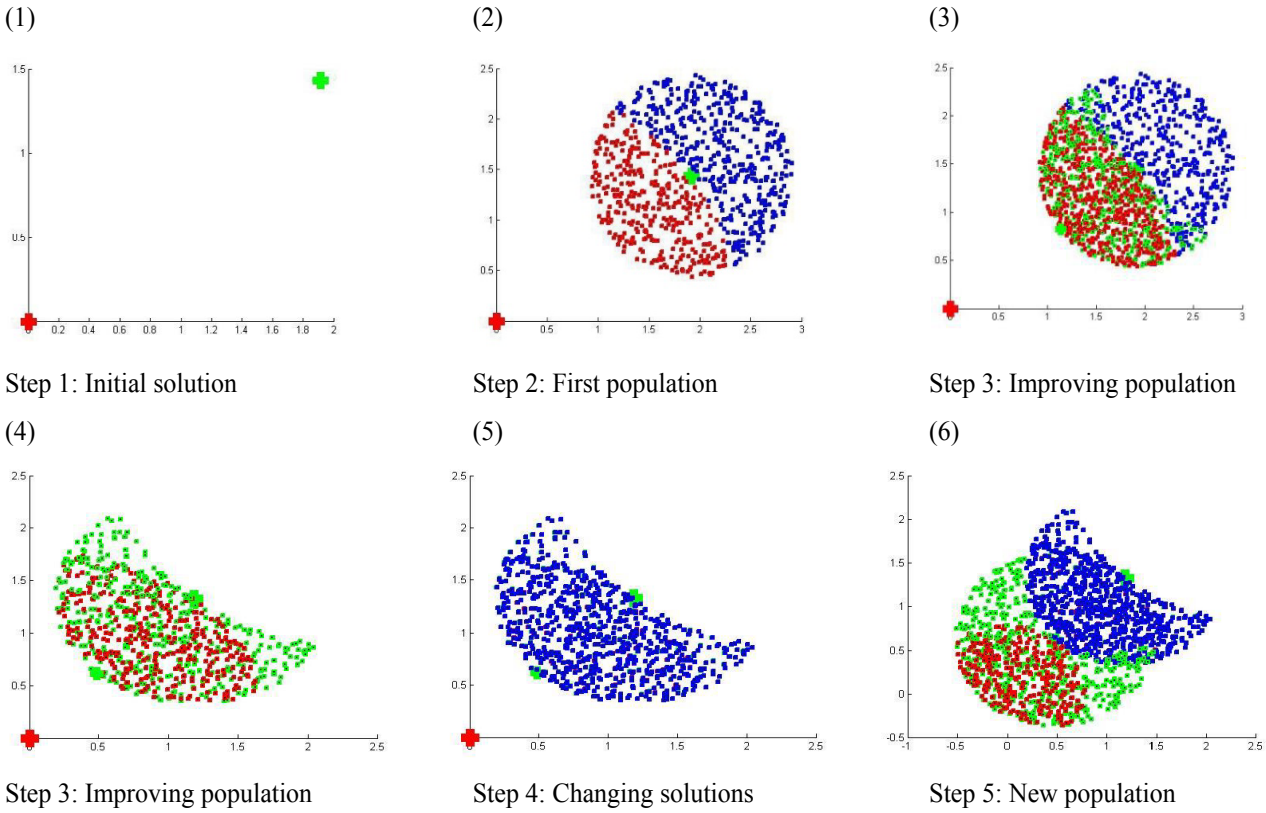

Figure 5: Steps of the algorithm to obtain optimal solution R^2.

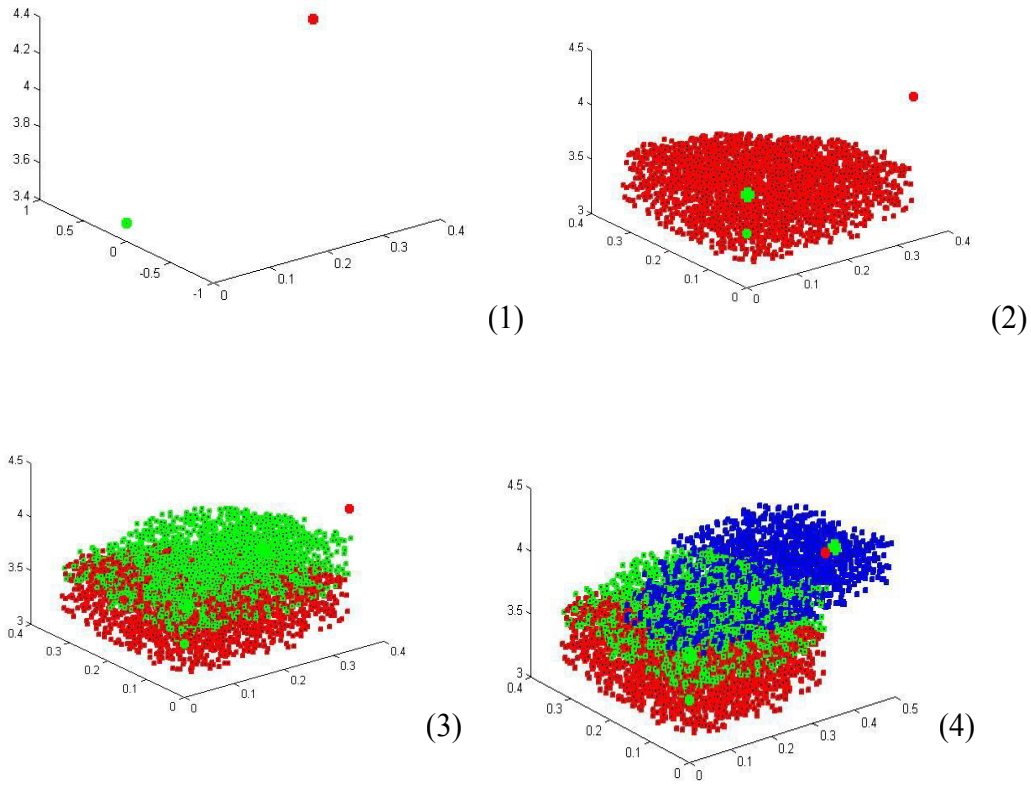

Figure 6: Steps of the algorithm to obtain optimal solution in R^3.

performances. Large number of eggs and slow convergent parameter set must be used in this state. The convergence rate is low if after large number of iterations same result is gained. Small number of eggs and quick convergent parameter set should be used here. Finally, if suitable results are gained, convergence rate is well and the common parameter set should be used. According to the computational results, convergence rate of the algorithm is completely high rather than other proposed meta-heuristic approaches.

Computational Results

Example 1 [9]

Consider the following problem:

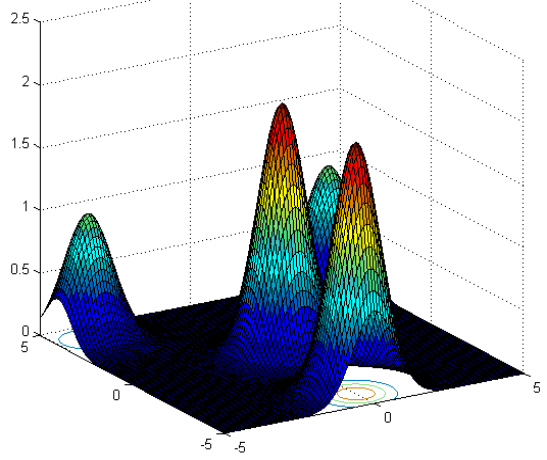

Figure 7: Behavior of the function: Example 1.

$$\min \exp(-(x-4)^2-(y-4)^2)+\exp(-(x+4)^2-(y-4)^2)+2\exp(-x^2-y^2)+2\exp(-x^2--(y+4)^2)$$

Figure 7 shows behavior of objective function in Example 1. To solve the problem, all efficient factors to obtain optimal solution are: number of eggs, stochastic constant (k), small positive number ε to change solutions, and initial feasible solution. According to the Table 2 the proposed meta-heuristic approach has presented a solution with less time and number of eggs than firefly algorithm. Behavior of agents to obtain optimal solution has been shown in Figure 8.

Example 2 [10]

Consider the following linear programming problem:

$$\min -3x_1+x_2$$

$$x_1+2x_2 \le 4$$

$$-x_1+x_2 \ge 0$$

Comparison LCA and exact methods has been proposed in Table 3. Figure 9 shows to move generations to optimal solution in feasible region.

Example 3 [11]

Consider the following non-linear programming problem:

$$\min -(x_1-4)^2-(x_2-4)^2$$

$$x_1+3 \le 0$$

$$-x_1+x_2-2 \le 0$$

$$x_1+x_2-4 \le 0$$

$$x_1,x_2-2 \ge 0$$

Comparison LCA and other methods by example 3 have been proposed in Table 4. Behavior of generations has been shown in Figure 10.

States of α	$x_{j+1}=x_j+\alpha d_{j0}$	Explanation	Logical decision	P. Infeasible solutions
$\alpha >> 2k$	$\alpha\to\infty \Rightarrow x_{j+1}\to\infty$	x_{j+1} will be infeasible.	This state should not be selected.	100%
$\alpha << 2k$	$\alpha\to-\infty \Rightarrow x_{j+1}\to-\infty$	x_{j+1} will be infeasible.	This state should not be selected.	100%
$\alpha=0$	$x_{j+1}=x_j$	x_{j+1} will not be changed.	α should not be near zero.	–
$\alpha=k$	$x_{j+1}=x_0$	x_0 is already in population.	This state should not be selected.	–
$\alpha=2k$	$x_{j+1}=x_j+kd_{j0}$	x_{j+1} will be feasible.	This state can be selected.	0%
$\alpha<k$	$x_{j+1}=x_j+2kd_{j0}$	x_{j+1} till be feasible.	This state can be selected.	0%

Table 1: States of α description.

Algorithms	N.Eggs/Firefly	N. Iterations	Optimal Solution	F Max	K	ε	x_0
LCA	24	2	(–0.03,–0.02)	1.99	1	0.01	(0.80,0.90)
LCA	20	3	(–0.10,–0.01)	1.95	1	0.01	(1.81,1.90)
FA[9]	25	20	(0,0)	2	–	–	–

Table 2: Comparison of LCA and firefly algorithm: Example 1.

Algorithms	N. Eggs	N. Iterations	Optimal Solution	F Max	K	ε	x_0
LCA	100	4	(3.85,0.59)	–11.50	1	0.01	(0.80,0.90)
Exact methods[10]	–	–	(4,0)	–12	–	–	–

Table 3: Comparison of LCA and exact methods: Example 2.

Algorithms	N. Eggs	N. Iterations	Optimal Solution	F Max	k	ε	x_0
LCA	100	2	(0.04,0.02)	–31.47	1	0.01	(0.81,0.90)
LCA	40	3	(0.04,0.11)	–30.73	1	0.01	(0.81,0.90)
LCA	100	4	(0.00,0.17)	–30.6	1	0.01	(3,1)
Classic methods [11]			(0,0)	–32	–		

Table 4: Comparison of LCA and other methods: Example 3.

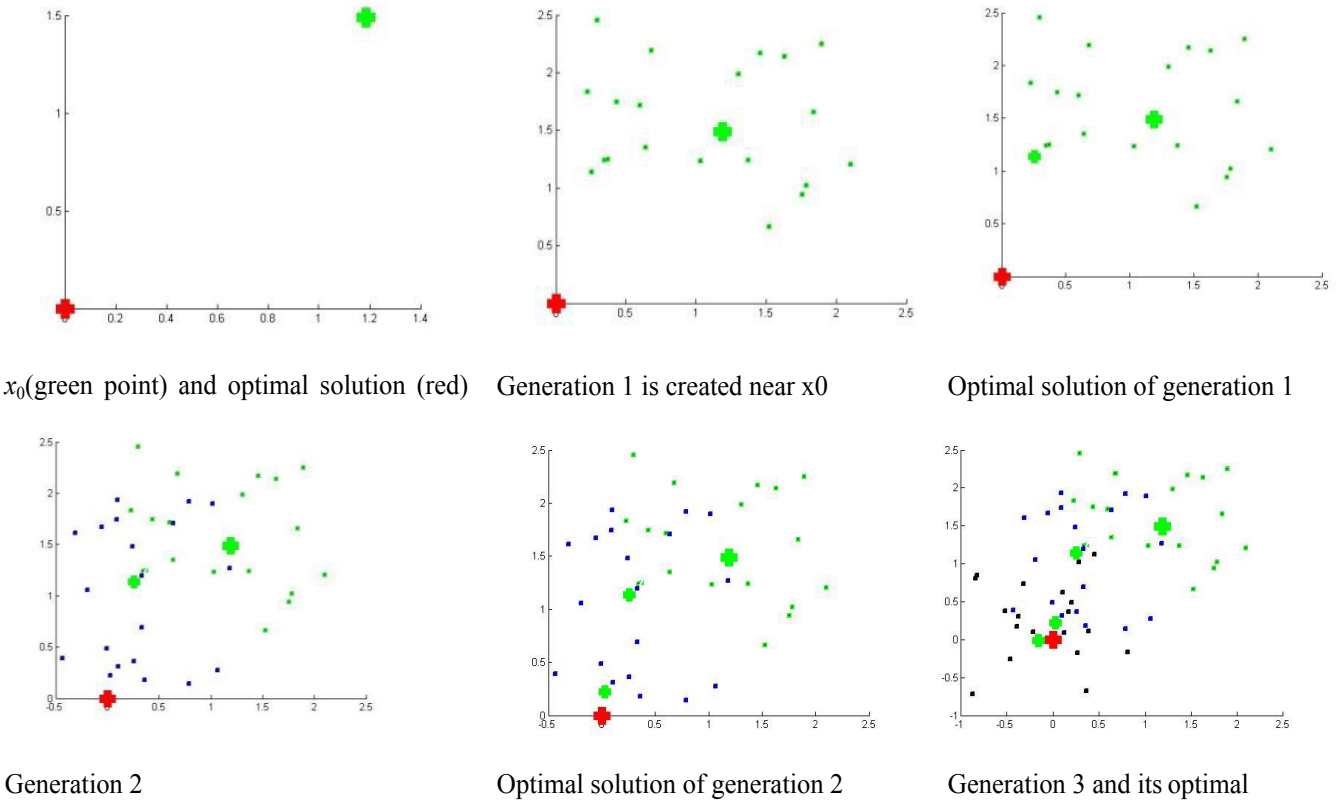

Figure 8: Generations have been moved to find optimal solution: Example 1.

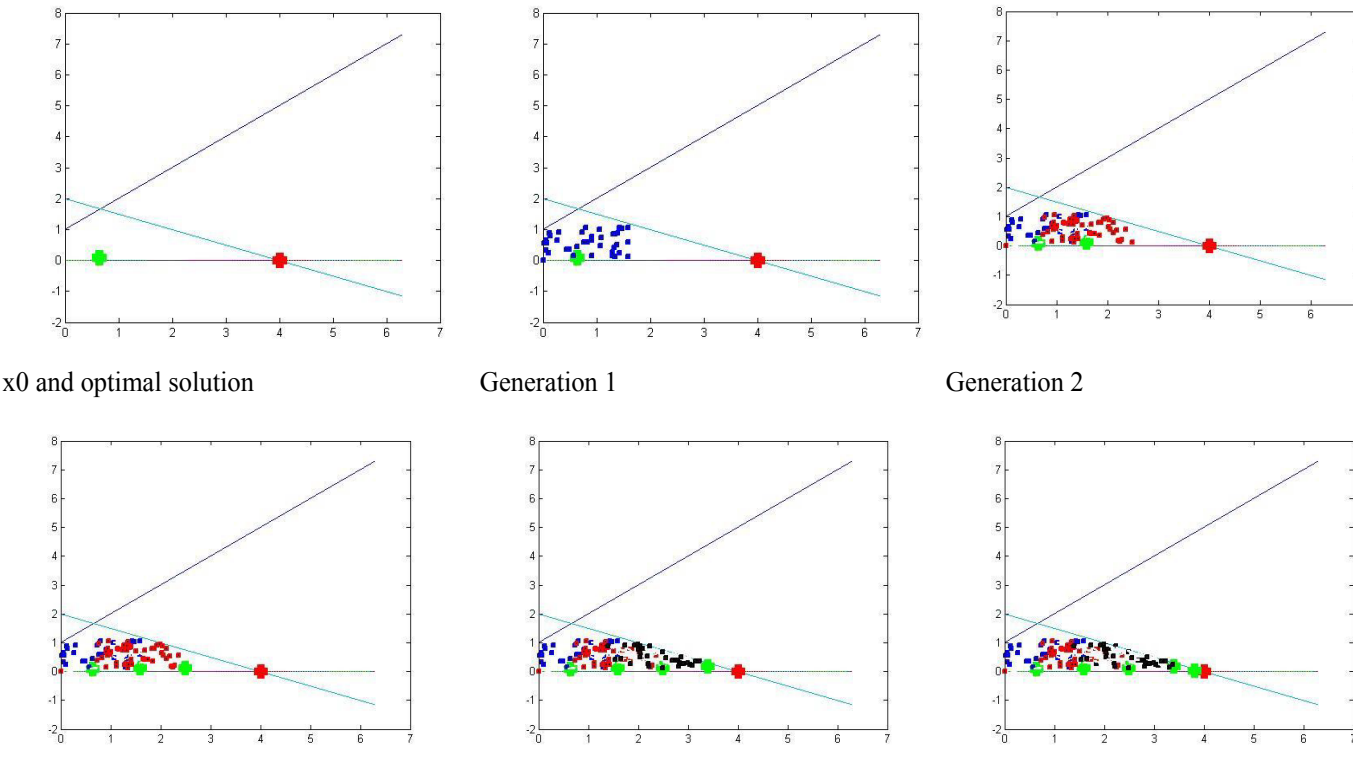

Figure 9: Generations have been created in feasible region and moved to optimal solution: Example 2.

The proposed algorithm is efficient for problems by more than two variables according to the following example.

Example 4 [10]

Consider the following linear programming problem:

Min $x_1+x_2-4x_3$

$x_1+ x_2+2x_3 \le 9$

$x_1+x_2-x_3 \le 2$

$-x_1+x_{2+}x_3 \le 4$

$x_1,x_2,x_3 \ge 0$

Comparison LCA and exact methods has been proposed in Table 5. Behavior of generations to find optimal solution has been shown in Figure 11.

Conclusion

Laying chicken algorithm is an easy meta-heuristic approach which optimizes different kinds of functions and optimization programming problems. Also, it seems efficient according to the examples. LCA was proposed as a natural event algorithm, not based on swarm intelligence unlike most of pervious meta-heuristic approaches. It ties to behavior of hen in process of produce chickens from eggs. In fact, LCA relates to both of biological and evolution computation because of its evolution and stochastic process.

LCA was successful because it does not miss the great solutions near initial solution particularly when k is small. The number of generations would be less according to a suitable feasible solution such as x_0 in fact consuming time to find optimal solution is much better than other meta-heuristic approaches.

Finally, there are many different NP-Hard problems which can be solved by meta-heuristic approaches especially using laying chicken algorithm. The simple MATLAB code of the LCA can be interested in the future researches especially for problems in large size. However, the proposed solution by LCA is near to optimal solution, but it is an approximate approach and the better algorithms can be proposed in the future researches.

Algorithms	N. Eggs	N. Iterations	Optimal Solution	F Max	K	ε
LCA	27	7	(0.31,0.00,4.29)	–16.87	1	0.01
Exact methods[10]	–	–	(0.33,0,4.33)	-17	–	–

Table 5: Comparison of LCA and other methods: Example 4.

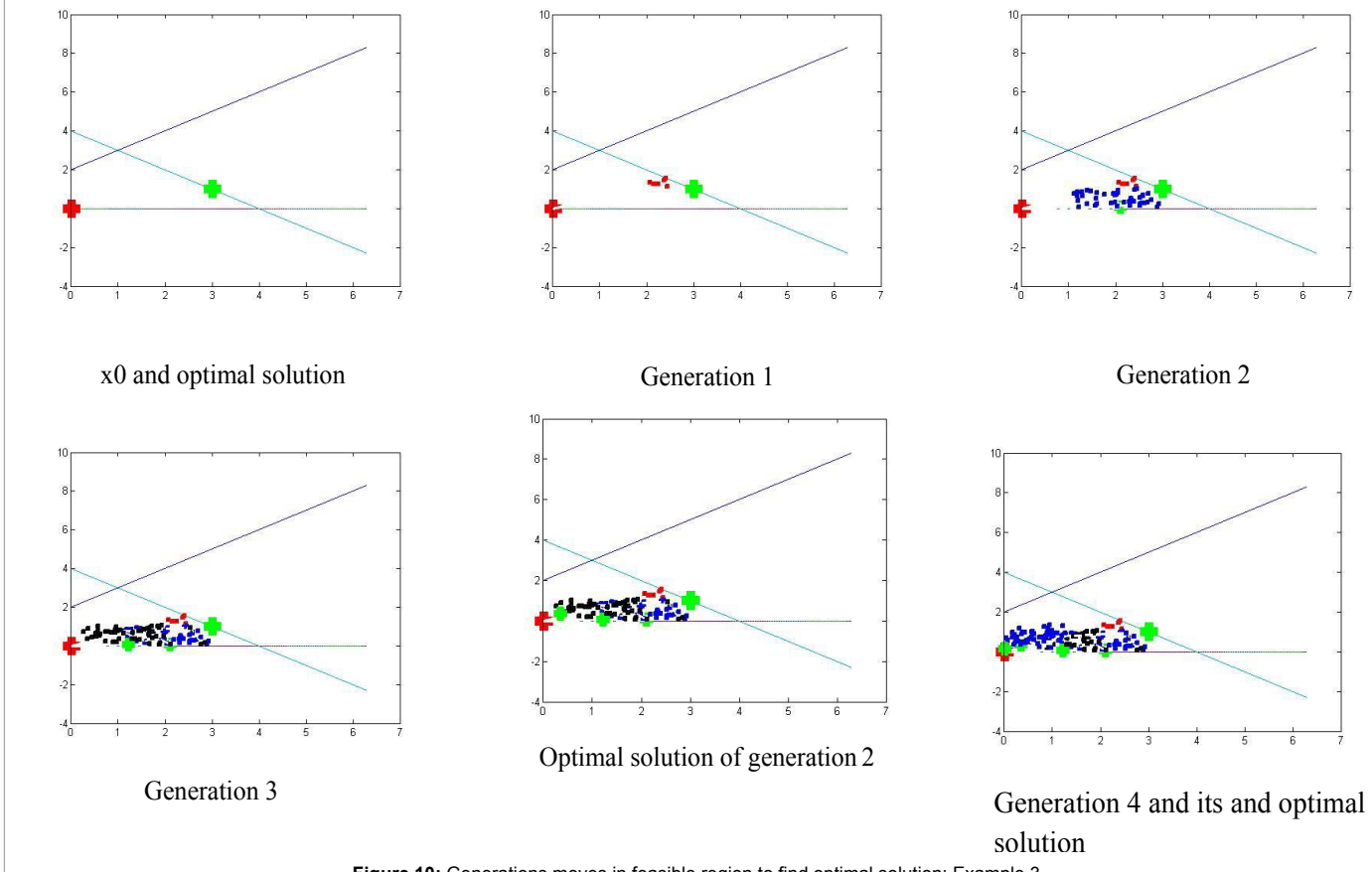

Figure 10: Generations moves in feasible region to find optimal solution: Example 3.

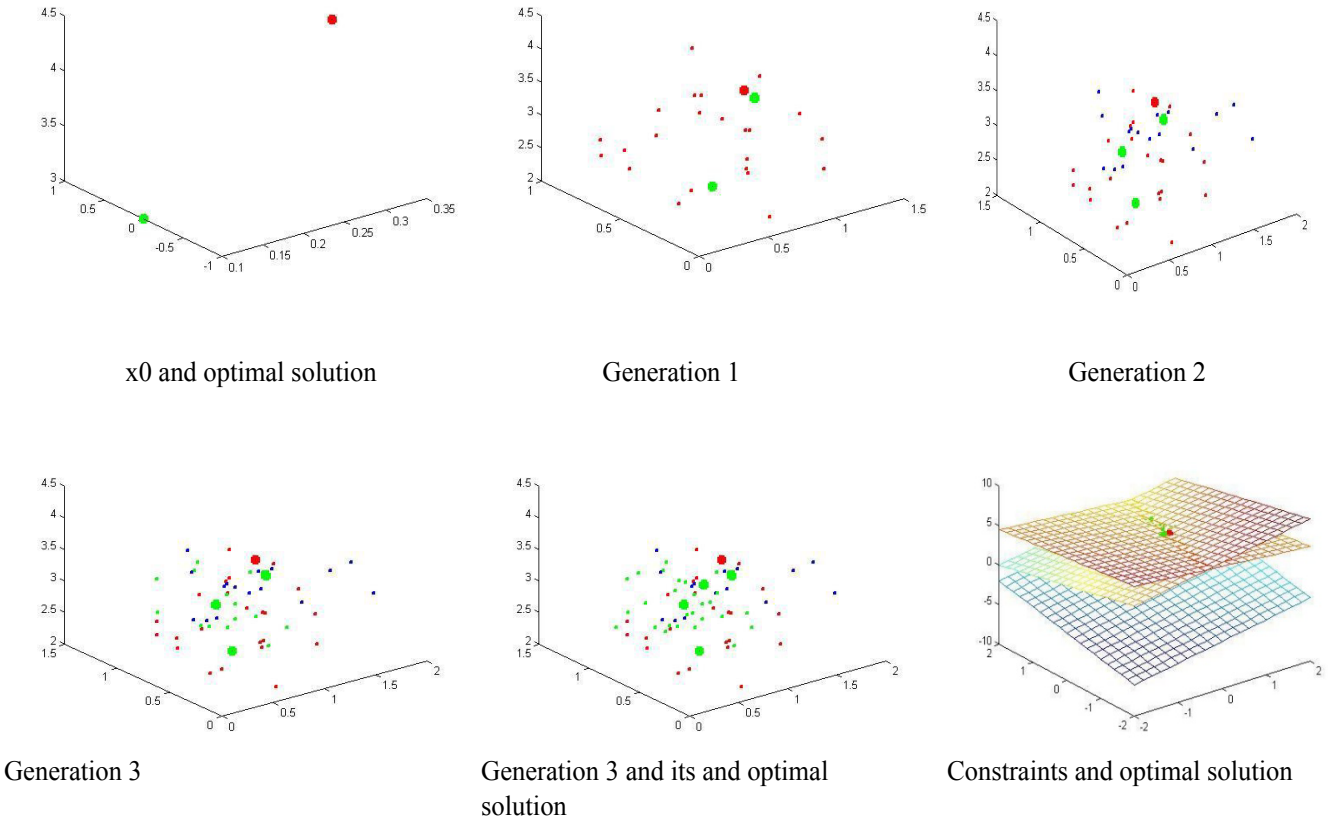

Figure 11: Generations moves in the feasible region to ind optimal solution: Example 4.

References

1. Beni G, Wang J (1989) Swarm intelligence in cellular robotic systems. In NATO Advanced Workshop on Robots and Biological Systems, Il Ciocco, Tuscany, Italy, pp: 703-712.
2. Colorni A, Dorigo M, Maniezzo V, Trubian M (1994) Ant system for job-shop scheduling. Belgian Journal of Operations Research. Statistics and Computer Science 34: 39-53.
3. Shi Y (2001) Particle swarm optimization: developments, applications and resources. In evolutionary computation 2001. Proceedings of the 2001 Congress 1: 81-86.
4. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization 39: 459-471.
5. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Communications in Nonlinear Science and Numerical Simulation 17: 4831-4845.
6. Yang XS, He X (2013) Bat algorithm: literature review and applications. International Journal of Bio-Inspired Computation 5: 141-149.
7. Cuevas E, Cienfuegos M, Zaldívar D, Pérez-Cisneros M (2013) A swarm optimization algorithm inspired in the behavior of the social-spider. Expert Systems with Applications 40: 6374-6384.
8. Meng X, Liu Y, Gao X, Zhang H (2014) A new bio-inspired algorithm: chicken swarm optimization. In International Conference in Swarm Intelligence. Springer International Publishing, pp: 86-94.
9. Yang XS (2010) Nature-inspired metaheuristic algorithms. Luniver press.
10. Bazzara M (2010) linear programming and Network Flows, Wiley. Inc, New York.
11. Bazzara M (2007) Non-linear programming Theory and Algorithms. Wiley Inc. New York.