I.J. Modern Education and Computer Science, **2014, 8, 18-24**
Published Online August 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2014.08.03

An Improved Chaotic Bat Algorithm for Solving Integer Programming Problems

Osama Abdel-Raouf
Department of Operations Research, Faculty of Computers and Information, Menoufia University, Menoufia, Shebin-El-come, Egypt.
Email:osamabd@hotmail.com.

Mohamed Abdel-Baset
Department of Operations Research, Faculty of Computers and Informatics, Zagazig University, El-Zera Square, Zagazig, Sharqiyah, Egypt.
Email:henawy2000@yahoo.com.

Ibrahim El-henawy
Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, El-ZeraSquare, Zagazig, Sharqiyah, Egypt.
Email:analyst_mohamed@yahoo.com.

Abstract—Bat Algorithm is a recently-developed method in the field of computational intelligence. In this paper is presented an improved version of a Bat Meta-heuristic Algorithm, (IBACH), for solving integer programming problems. The proposed algorithm uses chaotic behaviour to generate a candidate solution in behaviors similar to acoustic monophony. Numerical results show that the IBACH is able to obtain the optimal results in comparison to traditional methods (branch and bound), particle swarm optimization algorithm (PSO), standard Bat algorithm and other harmony search algorithms. However, the benefits of this proposed algorithm is in its ability to obtain the optimal solution within less computation, which save time in comparison with the branch and bound algorithm (exact solution method).

Index Terms—Bat algorithm; meta-heuristics; optimization; chaos; integer programming.

I. Introduction

The real world optimization problems are often very challenging to solve, and many applications have to deal with NP-hard problems [1]. To solve such problems, optimization tools have to be used even though there is no guarantee that the optimal solution can be obtained. In fact, for NP problems, there are no efficient algorithms at all. As a result of this, many problems have to be solved by trial and errors using various optimization techniques [2]. In addition, new algorithms have been developed to see if they can cope with these challenging optimization problems. Among these new algorithms, many algorithms such as particle swarm optimization, cuckoo search and firefly algorithm, have gained popularity due to their high efficiency. In this paper we have used IBACH algorithm for solving integer programming problems. Integer programming is NP-hard problems [3-10].The name “linear integer programming “is referred” to the class of combinatorial constrained optimization problems with integer variables, where the objective function is a linear function and the constraints are linear inequalities.” The Linear Integer Programming (also known as LIP) optimization problem can be stated in the following general form:

$$\text{Max } cx \tag{1}$$
$$\text{s.t. } Ax \leq b, \tag{2}$$
$$x \in Zn \tag{3}$$

where the solution $x \in Zn$ is a vector of n integer variables: $x = (x_1, x_2, \ldots, x_n)^T$ and the data are rational and are given by the $m \times n$ matrix A, the $1 \times n$ matrix c, and the $m \times 1$ matrix b. This formulation includes also equality constraints, because each equality constraint can be represented by means of two inequality constraints like those included in eq. (2).

Integer programming addresses the problem raised by non-integer solutions in situations where integer values are required. Indeed, some applications do allow a continuous solution. For instance, if the objective is to find the amount of money to be invested or the length of cables to be used, other problems preclude it: the solution must be discrete [3]. Another example, if we are considering the production of jet aircraft and x1 = 8.2 jet airliners, rounding off could affect the profit or the cost by millions of dollars. In this case we need to solve the problem so that an optimal integer solution is guaranteed.

The possibility to obtain integer values is offered by integer programming: as a pure integer linear programming, in which all the variables must assume an integer value, or as a mixed-integer linear programming which allows some variables to be continuous, or a 0-1 integer model, all the decision variables have integer values of zero or one[10].

A wide variety of real life problems in logistics, economics, social sciences and politics can be formulated as linear integer optimization problems. The combinatorial problems, like the knapsack-capital budgeting problem, warehouse location problem, travelling salesman problem, decreasing costs and machinery selection problem, network and graph problems, such as maximum flow problems, set covering problems, matching problems, weighted matching problems, spanning trees problems and many scheduling problems can also be solved as linear integer optimization problems [11-14].

Exact integer programming techniques such as cutting plane techniques [15-17]. The branch and the bound both have high computational cost, in large-scale problems [18-19]. The branch and the bound algorithms have many advantages over the algorithms that only use cutting planes. One example of these advantages is that the algorithms can be removed early as long as at least one integral solution has been found and an attainable solution can be returned although it is not necessarily optimal. Moreover, the solutions of the LP relaxations can be used to provide a worst-case estimate of how far from optimality the returned solution is. Finally, the branch method and the bound method can be used to return multiple optimal solutions.

Since integer linear programming is NP-complete, for that reason many problems are intractable. So instead of the integer linear programming, the heuristic methods must be used. For example, Swarm intelligence metaheuristics, amongst which an ant colony optimization, artificial bee colony optimization particle swarm optimization [20-24]. Also Evolutionary algorithms, differential evolution and tabu search were successfully applied into solving integer programming problems [25-27]. Heuristics typically have polynomial computational complexity, but they do not guarantee that the optimal solution will be captured. In order to solve integer programming problems, most of the heuristics truncate or round the real valued solutions to the nearest integer values. In this paper, Bat algorithm is applied to integer programming problems and the performance was compared with other harmony search algorithms.

This paper is organized as follows: after introduction, the original Bat Algorithm is briefly introduced in section 2. Section 3 introduces the meaning of chaos. In section 4, the proposed algorithm is described, while the results are discussed in section 5. Finally, conclusions are presented in section 6.

II. The Original Bat Algorithm

Bat Algorithm has been developed by Xin-She Yang in 2010 [28]. The algorithm exploits also called echolocation of bats. Bats use sonar echoes to detect and avoid obstacles. It is generally known that sound pulses are transformed to frequency which is reflected from obstacle. Bats can use time delay from emission for reflection and use it for navigation. They typically emit short and loud sound impulses and the pulse rate is usually defined as 10 to 20 times per second. After hitting and reflecting, bats transform their own pulses to useful information to gauge how far away the prey is. Bats use wavelengths, that vary from range (0.7, 17) mm or inbound frequencies (20,500) kHz. By implementation, pulse frequency and pulse rates have to be defined. Pulse rate can be simply determined from range 0 to 1, where 0 meaning there is no emission and 1 meaning bats are emitting maximum (5-8). This behaviour can be used to formulate the new bat algorithm. Yang [28] used three generalized rules for Bat Algorithm:

I All bats use echolocation to sense distance, and they also predict the difference between food/prey and background barriers in some magical way.

II Bats fly randomly with velocity vi at position xi with a fixed frequency fmin, varying wavelength λ and loudness A0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission r ∈[0, 1], depending on the proximity of their target.

III III. Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) A0 to a minimum constant value Amin. Initialization of the bat population is performed randomly. Generating the new solutions is performed by moving virtual bats according the following equations:

$$f_i = f_{min} + (f_{max} - f_{min})\beta\ , \tag{4}$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - best)f_i, \tag{5}$$

$$x_i^t = x_i^{t-1} + v_i^t\ , \tag{6}$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. Here X is the current global best solution which is located after comparing all the solutions among all the bats.

For the local search part, once a solution is selected among the current solutions, a new solution for each bat is generated located using random walk [29-33].

$$x_{new} = x_{old} + \varepsilon A_t \tag{7}$$

where ε is the scaling factor and A_i^t is the loudness, the loudness A_0 and the rate r_i of pulse emission have to be updated accordingly as the iterations proceed. These equations are:

$$A_i^{t+1} = \alpha A_i^t \tag{8}$$

$$r_i^{t+1} = r^0{}_i\ [1- exp(-\gamma t)] \tag{9}$$

where α and γ are constants.

The basic steps of BA can be summarized as the pseudocode shown in Figure 1.

Bat Algorithm

Begin
Objective function f (x), x = (x_1, ...,x_d)^T
Initialize the bat population x_i and v_i for (i = 1,2, ...,n)
Define pulse frequency f_i at x_i
Initialize pulse rates r_i and the loudness A_i
While*(t <Max number of iterations)*
Generate new solutions by adjusting frequency and,
Updating velocities and locations/solutions (equations 4 to 6)
if*(rand (0,1)> r_i)*
Select a solution among the best solutions
Generate a local solution around the best solution
End if
Generate a new solution by flying randomly
if*(rand (0,1)< A_i &f(x_i) < f(x))*
Accept the new solutions
Increase r_i and reduce A_i
End if
Rank the bats and find the current best
End while
Post process results and visualization
End

Fig. 1 Pseudo code of the bat algorithm

III. Chaos Theory

Generating random sequences with longer periods and good consistency is very important for easily simulating complex phenomena, sampling, numerical analysis, decision making and especially in heuristic optimization [34]. Its quality determines the reduction of storage and computation time to achieve a desired accuracy [35]. Chaos is a deterministic, random-like process found in a nonlinear, dynamical system, which is non-period, non-converging and non-bounded. Moreover, it depends on its initial condition and parameters [36-38]. Applications of chaos has several disciplines including operations research, physics, engineering, economics, biology, philosophy and computer science [39-41].

Recently chaos has been extended to various optimization areas because it can more easily escape from local minima and improve global convergence in comparison with other stochastic optimization algorithms [37-42]. Using chaotic sequences in Bat Algorithm can be helpful to improve the reliability of the global optimality, and they also enhance the quality of the results.

A. Chaotic maps

At random-based optimization algorithms, the methods using chaotic variables instead of random variables are called chaotic optimization algorithms (COA) [37]. In these algorithms, due to the non-repetition and ergodicity of chaos, it can carry out overall searches at higher speeds than stochastic searches that depend on probabilities [43-52]. To resolve this issue, herein one-dimensional and non-invertible maps (are mathematical systems that model a single variable as it evolves over discrete steps in time) are utilized to generate chaotic sets. We will illustrate some of well-known one-dimensional maps as:

1 . Logistic map

The Logistic map is defined by:

$$Y_{n+1} = \mu Y_n(1 - Y_n)\ Y\in(0,1)\ 0 < \mu \le 4 \quad (10)$$

2 . The Sine map

The Sine map is written as the following equation:

$$Y_{n+1} = \frac{\mu}{4}\sin(\pi Y_n)\ Y\epsilon\ (0,1)\ 0 < \mu \le 4 \quad (11)$$

3 . Iterative chaotic map

The iterative chaotic map with infinite collapses is described as:

$$Y_{n+1} = \sin\left(\frac{\mu\pi}{Y_n}\right)\ \mu \in (0,1) \quad (12)$$

4 . Circle map

The Circle map is expressed as:

$$Y_{n+1} = Y_n + \alpha - \left(\frac{\beta}{2\pi}\right)\sin(2\pi Y_n)\ \bmod 1 \quad (13)$$

5 . Chebyshev map

The family of Chebyshev map is written as the following equation:

$$Y_{n+1} = \cos(k\cos^{-1}(Y_n))\ \ Y \in (-1,1) \quad (14)$$

6 . Sinusoidal map

This map can be represented by

$$Y_{n+1} = \mu Y_k^2 \sin(\pi Y_n) \quad (15)$$

7. Gauss map

The Gauss map is represented by:

$$Y_{n+1} = \begin{cases} 0 & Y_n = 0 \\ \frac{\mu}{Y_n}\ \bmod 1 & Y_n \neq 0 \end{cases} \quad (16)$$

8. Sinus map

Sinus map is formulated as follows:

$$Y_{n+1} = 2.3(Y_n)^{2\sin(\pi Y_n)} \quad (17)$$

9. Dyadic map

Also known as the dyadic map bit shift map, 2x mod 1 map, Bernoulli map, doubling map or saw tooth map. Dyadic map can be formulated by a mod function:

$$Y_{n+1} = 2Y_n\ \bmod 1 \quad (18)$$

10 . Singer map

Singer map can be written as:

$$Y_{n+1} = \mu(7.86Y_n - 23.31Y_n^2 + 28.75Y_n^3 - 13.3Y_n^4 \quad (19)$$

μ between 0.9 and 1.08

11.Tent map

This map can be defined by the following equation:

$$Y_{n+1} = \begin{cases} \mu Y_n & Y_n < 0.5 \\ \mu(1 - Y_n) & Y_n \geq 0.5 \end{cases} \tag{20}$$

IV. The Proposed Algorithm (IBACH) for Solving Integer Programming Problems

In the proposed chaotic Bat algorithm, we used chaotic maps to tune the Bat algorithm parameters and improve the performance [40]. The steps of the proposed chaotic Bat Algorithm for solving integer programming problems are as follows:

Step 1 Set the initial conditions: population x_i (i = 1, 2 ...n) and V_i, pulse frequency fiat x_i and pulse rates r_i and the loudness A_i

Step 2 Calculate the average position and the optimal position of the bat colony.

Step 3 Using the equations 4 to 6 update velocities and locations/solutions and Generate new solutions by adjusting frequency.

$$f_i = f_{min} + (f_{max} - f_{min})Si \text{ ,where } s_i \equiv \text{chaotic map} \tag{21}$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - best) f_i * S_i \tag{22}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{23}$$

Step 4 If ($rand > r_i$) then select a solution among the best solutions and generate a local solution around the selected best solution with the following equation

$$x_{new} = x_{old} + \varepsilon A_t \tag{24}$$

Whereε∈[-1,1] If not, skip this step.

Step 5 If *($rand < A_i$ & $f(x_i) < f(x)$)*then accept the new solutions. Increase r_i and reduce A_i with the following two equations

$$A_i^{t+1} = \alpha A_i^t \tag{25}$$

$$r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)] \tag{26}$$

If not, skip this step.

Step 6 Rank the bats and find the current best X.

Step 7 If the iterations attain to the maximum number, then stopped and output the global optimal solution. If not, go to ***step 2*** to continue the search.

A. Handling Constraints

One of the well-known techniques of handling constraints is using penalty function, which transforms constrained problem into unconstrained ones, consisting of a sum of the objective and the constraints weighted by penalties. By using penalty function methods, the objectives are inclined to guide the search toward the feasible solutions. Hence, in this paper the corresponding objective function used in is defined and described as:

$$\min F(x) = f(x) + \lambda \sum_{n=1}^{K} \max(0, g_n) \tag{27}$$

where f(x) is the objective function for assignment problem, λ is the penalty coefficient and it is set to 107 in this paper ,K is the number of constraints and gn the constraints of the problem.

V. Numerical Results

Several examples have been done to verify the weight of the proposed algorithm. The initial parameters setting of the algorithms is as follows: HMS=50 and itermax=1000, HMCR = 0.9; PAR_{max} = 1; PAR_{min} =0.1; bw_{max} = 1; bw_{min} = 0.01; n= 40, f_{min}= 0 ,f_{max}= 2, A_0 = 0.5 ,r = 0.5. The results of IBACH algorithm are conducted from 50 independent runs for each problem and measured according to the best values in these runs .The selected chaotic map for all examples is the Sinusoidal map, whose equation is shown below:

$$Y_{n+1} = \mu Y_k^2 \sin(\pi Y_n) \tag{28}$$

Where n is the iteration number, all the experiments were performed on a Windows 7 Ultimate 64-bit operating system; processor Intel Core i5 760 running at 2.81 GHz; 4 GB of RAM and codes were implemented in C#.

A. Test Problem 1

Max $z = 7x_1 + 9x_2$
s.t.
$-x_1 + 3x_2 \leq 6$
$7x_1 + x_2 \leq 35$
$x_1, x_2 \geq 0$ *and integer.*

B. Test Problem 2

Max $w = 4x_1 + 6x_2 + 2x_3$
s.t.
$4x_1 - 4x_2 \leq 5$
$-x_1 + 6x_2 \leq 5$
$-x_1 + x_2 + x_3 \leq 5$
$x_1, x_2, x_3 \geq 0$ *and integer.*

C. Test Problem 3

Min $z = -5x_1 + 7x_2 + 10x_3 - 3x_4 + x_5$
s.t.
$x_2 - 2x_3 - x_4 + x_5 \leq -2$
$x_1 + 3x_2 - 5x_3 + x_4 + 4x_5 \geq 0$
$2x_1 + 6x_2 - 3x_3 + 2x_4 + 2x_5 \geq 4$
$x_i = 0$ *or 1 ,i=1,2,…,5.*

D. Test Problem 4

$$\begin{aligned}
&\text{Max } z = 3x_1-2x_2+4x_3+5x_4+x_5+x_6+2x_7-6x_8+x_9-x_{10}\\
&\text{s.t.}\\
&x_1+5x_2+3x_3 \geq 8\\
&x_7+5x_8+2x_9+x_{10} = 7\\
&4x_1+x_2+3x_3+x_4+x_7 \leq 4\\
&6x_1+x_5+x_6+3x_8-2x_9 \leq 5\\
&4x_1-2x_2+3x_3-4x_4+5x_5 \geq 2\\
&-x_1+x_2+2x_3-x_4+x_5+7x_7 \leq 3\\
&-3x_1-x_2+4x_3-x_4+2x_5+5x_6+9x_8+x_{10} = 10\\
&x_i \geq 0,\ i=1,2,\ldots,10 \text{ and integer}
\end{aligned}$$

E. Test Problem 5

$$\begin{aligned}
&\text{Min } z = 3x_1 -2x_2 + 5x_3 + x_4 - x_5 + 2x_6 + 7x_7 + 3x_8 + 11x_9 + 22x_{10} -15x_{11} + 9x_{12} + 7x_{13} -13x_{14} + x_{15} + 8x_{16} + 19x_{17} + 4x_{18} - 9x_{19} + 10x_{20}\\
&\text{s.t.}\\
&x_{16}+3x_{18} + 9x_{19} + x_{20} \geq 11\\
&3x_8 + 7x_{17} + 8x_{19} + x_{20} \geq 18\\
&x_7 + x_{13} + x_{15} + 3x_{16} + 3x_{18} \leq 25\\
&x_1 + x_2 +12x_7 + 5x_8 + 6x_{15} - x_{18} \leq 6\\
&-3x_3 -2x_4 + 7x_5 -9x_6 + x_{13} +5x_{17} \geq 21\\
&x_6 -7x_{10} + 3x_{11} + x_{12} + 8x_{13} -9x_{14} \geq 4\\
&-3x_1 + x_2 + 8x_3 + x_{15} +22x_{19} - x_{20} \leq 83\\
&5x_1 + 8x_2 + 7x_3 + x_6 + 2x_7 + 9x_{12} + 7x_{15} \leq 9\\
&10x_1 + 2x_6 - x_{10} + 6x_{11} + 11x_{13} + 8x_{15} -9x_{19} = 40\\
&2x_1 + 3x_2 -7x_4 + 4x_5 + 18x_9 + x_{11}\ 9x_{15} + 4x_{16} \geq 10\\
&3x_1 -7x_2 -2x_4 + 3x_6 + 8x_8 -4x_{10} + 11x_{12} + 5x_{14} + 7x_{19} \geq 32\\
&22x_1 + 8x_3 + 8x_5 + 18x_7 + 4x_8 -3x_{10} + 19x_{11} -8x_{17} + x_{18} \geq 17\\
&x_1 + 3x_2 - x_3 + 7x_4 + 11x_5 + 18x_8 + 5x_{14} + 11x_{17} + x_{18} +4x_{20} \geq 22\\
&-8x_1 + 5x_3 - x_5 + x_7 + x_9 + 5x_{10} -7x_{11} + x_{13} + x_{15} + 8x_{17} + 2x_{20} \geq 9\\
&4x_1 + 5x_6 + x_8 +14x_{10} + 11x_{12} + x_{13}-5x_{14} + x_{15} + 11x_{16} + 3x_{18} + 3x_{20} \geq 16\\
&x_i \geq 0,\ i=1,2,\ldots,20 \text{ and integer.}
\end{aligned}$$

F. Test Problem 6

$$\begin{aligned}
&\max Z= x_1 + 2x_2 + 7x_3 + 2x_4 + x_5 + 4x_6 + x_7 -5x_8 + x_9 + 2x_{10} + 4x_{11} + 7x_{12} + 5x_{13} + 3x_{14} + 9x_{15} + 5x_{16} + x_{18} + 8x_{19} + 6x_{20} + x_{21} -3x_{22} + 7x_{23} - x_{24} -3x_{25} + 2x_{26} + x_{27} + 9x_{28} + 7x_{29} + 4x_{30}\\
&\text{s.t.}\\
&2x_1 + 2x_3 + x_{12}+ 9x_{15} -3x_{20} + 3x_{29} \leq 70\\
&2x_1 + 5x_5+ 2x_{13} + 2x_{15} + 2x_{16} + 2x_{28} \leq 19\\
&x_2 + x_3 + x_4 + x_7 + 2x_8 + x_{11}+8x_{14} + 2x_{25} \leq 20\\
&2x_8 - x_{11}-3x_{12} -3x_{13} + 2x_{19} + x_{20} +3x_{22} + x_{23} \leq 11\\
&-3x_{11} + x_{14} + 3x_{15} + 2x_{16} + x_{18}+ 2x_{22} + 2x_{26} \leq 95\\
&2x_{12} + x_{14} + 2x_{15} + 2x_{18} + 2x_{24} + 2x_{25} + 3x_{27} \leq 40\\
&x_2 + x_3 + x_4+ 2x_8 + 2x_{10} + x_{17} + x_{18} + x_{20} + x_{21} + x_{22} \leq 35\\
&4x_1 + 2x_9 + x_{10}+3x_{13} + 3x_{16} -9x_{17} + x_{18} + x_{24} + 5x_{27} + 4x_{29} + x_{30} \leq 40\\
&-3x_5 + x_9 +2x_{12} + 5x_{13} + 4x_{16} + x_{17} + x_{19} + x_{21}+ 4x_{25} -3x_{27} + 2x_{30} \leq 100\\
&5x_1 + x_3 + x_5 + 2x_6 + x_8 + 2x_9 - x_{10}+5x_{12} + x_{14} + x_{15} + 3x_{16}- 9x_{17} + x_{18} \leq 7\\
&x_1+ 2x_6 + 2x_7 + 2x_{14} + 11x_{15} + x_{16} -3x_{21} + 10x_{24} + 2x_{25} + 8x_{26} -3x_{28} + 11x_{29} \leq 62\\
&x_2 + x_4 + x_7 + x_9 + 2x_{11} -9x_{13} + 2x_{17} + 5x_{18} + x_{20} + x_{21}\ 4x_{24} + 3x_{26} + 5x_{27} + 4x_{30} \leq 51\\
&x_1 + x_3 + 2x_4 + 2x_6 + 3x_7 + 2x_8 -2x_{10} + 2x_{13} -5x_{15} + 2x_{19} + 3x_{20} + 4x_{21} + 3x_{23} + 4x_{28} \leq 22\\
&2x_2 + x_4 + 5x_5 + 4x_6 + 2x_7 + 3x_{13} + 8x_{17} + 2x_{19} + x_{21} + 2x_{22} + 2x_{23} + 2x_{24} + 10x_{25} -3x_{26} + 2x_{27} + 3x_{28} + 2x_{29} + x_{30} \leq 60\\
&x_i \geq 0,\ i=1,2,\ldots,30 \text{ and integer.}
\end{aligned}$$

Table 1 Optimal solution of selected problems

	Exact method		The Best Solution				
	Optimal sol.	**Optimal values**	**PSO[24]**	**HS[53]**	**IHS[54]**	**BA[28]**	**IBACH**
problem1	55	X_i=(4,3)	55	55	55	55	55
problem2	26	X_i=(2,1,6)	24	21	25	24	26
problem3	9	X_i=(1,1,0,0,0)	8	7	9	9	9
problem4	9	X_i=(0,2,0,2,3,1,0,0,2,3)	7	6	7	7	9
problem5	16	X_i=(0,0,0,0,0,0,0,0,0,0,1,4,0,4,3,0,2,4,0,3,0)	14	11	13	14	16
problem6	446	X_i=(0,0,0,0,0,0,0,0,0,16,20,4,4,0,3,0,0,0,24,3,0,0,0,0,0,4,0,1,0,8)	443	405	422	440	446

Table 1 shows the results of IBACH algorithm are privileged compared with the results of particle swarm optimization (PSO), Standard harmony search algorithm (HS), standard bat algorithm (BA) and improved harmony search algorithm (IHS). In comparison with exact values we find that the results of IBACH algorithm are very close to the exact values of selected problems under the study. If a large number of variables are to be found, then it is hard to go past the classical methods. More usually, though, users will choose to use the proposed algorithm, to save their own time and to gain reliability. for example when we solved test problem number 6 by proposed algorithm it took time 7 seconds ,but when we solved it by branch and bound(exact method) it took time 396 seconds .

The reason for getting better results than the other algorithms considered is that the search power of bat algorithm. Adding to this, using chaos improves the performance of the algorithm.

VI. Conclusions

This paper has introduced an improved Bat Algorithm by blending with chaos for solving integer programming problems. Several examples have been used to prove the effectiveness of the proposed methods. The proposed algorithm managed to solve a large scale of problems that traditional method could not solve due to exponential growth in time and space complexities. The solution procedure will not face the same time waste in going through non-converging iterations as traditional methods do. IBACH algorithm is superior to both HS and IHS in terms of both efficiency and success rate. This implies that IBACH is potentially more powerful in solving NP-hard problems.

References

[1] L. A. Wolsey, "Integer programming," IIE Transactions, vol. 32, pp. 2-58, 2000.

[2] G. B. Dantzig, Linear programming and extensions: Princeton university press, 1998.

[3] G. L. Nemhauser and L. A. Wolsey, Integer and combinatorial optimization vol. 18: Wiley New York, 1988.

[4] E. Beale, "Integer programming," in Computational Mathematical Programming, ed: Springer, 1985, pp. 1-24.

[5] C. H. Papadimitriou and K. Steiglitz, Combinatorial optimization: algorithms and complexity: Courier Dover Publications, 1998.

[6] H. Williams, "Logic and Integer Programming, International Series in Operations Research & Management Science," ed: Springer, 2009.

[7] A. Schrijver, Theory of linear and integer programming: Wiley. com, 1998.

[8] D. Bertsimas and R. Weismantel, Optimization over integers vol. 13: Dynamic Ideas Belmont, 2005.

[9] J. K. Karlof, Integer programming: theory and practice: CRC Press, 2005.

[10] M. Jünger, T. Liebling, D. Naddef, G. Nemhauser, W. Pulleyblank, G. Reinelt, et al., 50 Years of Integer Programming 1958–2008: Springer, Berlin, 2010.

[11] D.-S. Chen, R. G. Batson, and Y. Dang, Applied integer programming: modeling and solution: Wiley. com, 2011.

[12] K. L. Hoffman and M. Padberg, "Solving airline crew scheduling problems by branch-and-cut," Management Science, vol. 39, pp. 657-682, 1993.

[13] J. D. Little, K. G. Murty, D. W. Sweeney, and C. Karel, "An algorithm for the traveling salesman problem," Operations research, vol. 11, pp. 972-989, 1963.

[14] M. Grotschel and L. Lovász, "Combinatorial optimization," Handbook of combinatorics, vol. 2, pp. 1541-1597, 1995.

[15] R. E. Gomory, "Outline of an algorithm for integer solutions to linear programs," Bulletin of the American Mathematical Society, vol. 64, pp. 275-278, 1958.

[16] R. E. Gomory, "An algorithm for integer solutions to linear programs," Recent advances in mathematical programming, vol. 64, pp. 260-302, 1963.

[17] R. E. Gomory, "Early integer programming," Operations Research, pp. 78-81, 2002.

[18] J. Tomlin, "Branch and bound methods for integer and non-convex programming," Integer and Nonlinear Programming, American Elsevier Publishing Company, New York, pp. 437-450, 1970.

[19] S. Rouillon, G. Desaulniers, and F. Soumis, "An extended branch-and-bound method for locomotive assignment," Transportation Research Part B: Methodological, vol. 40, pp. 404-423, 2006.

[20] M. Tuba, "Swarm intelligence algorithms parameter tuning," in Proceedings of the 6th WSEAS international conference on Computer Engineering and Applications, and Proceedings of the 2012 American conference on Applied Mathematics, 2012, pp. 389-394.

[21] R. Jovanovic and M. Tuba, "An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem," Applied Soft Computing, vol. 11, pp. 5360-5366, 2011.

[22] R. Jovanovic and M. Tuba, "Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem," Computer Science and Information Systems, vol. 10, pp. 133-149, 2013.

[23] B. Akay and D. Karaboga, "Solving integer programming problems by using artificial bee colony algorithm," in AI* IA 2009: Emergent Perspectives in Artificial Intelligence, ed: Springer, 2009, pp. 355-364.

[24] M. G. Omran, A. Engelbrecht, and A. Salman, "Barebones particle swarm for integer programming problems," in Swarm Intelligence Symposium, 2007. SIS 2007. IEEE, 2007, pp. 170-175.

[25] G. Rudolph, "An evolutionary algorithm for integer programming," in Parallel Problem Solving from Nature—PPSN III, ed: Springer, 1994, pp. 139-148.

[26] M. G. Omran and A. P. Engelbrecht, "Differential evolution for integer programming problems," in Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, 2007, pp. 2237-2242.

[27] F. Glover, "Tabu search—part II," ORSA Journal on computing, vol. 2, pp. 4-32, 1990.

[28] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in Nature inspired cooperative strategies for optimization (NICSO 2010), ed: Springer, 2010, pp. 65-74.

[29] X.-S. Yang, "Bat algorithm for multi-objective optimisation," International Journal of Bio-Inspired Computation, vol. 3, pp. 267-274, 2011.

[30] X.-S. Yang and A. H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," Engineering Computations, vol. 29, pp. 464-483, 2012.

[31] A. H. Gandomi, X.-S. Yang, A. H. Alavi, and S. Talatahari, "Bat algorithm for constrained optimization tasks," Neural Computing and Applications, pp. 1-17, 2013.
[32] X.-S. Yang, Engineering optimization: an introduction with metaheuristic applications: Wiley. com, 2010.
[33] T. C. Bora, L. d. S. Coelho, and L. Lebensztajn, "Bat-Inspired optimization approach for the brushless DC wheel motor problem," Magnetics, IEEE Transactions on, vol. 48, pp. 947-950, 2012.
[34] L. M. Pecora and T. L. Carroll, "Synchronization in chaotic systems," Physical review letters, vol. 64, pp. 821-824, 1990.
[35] D. Yang, G. Li, and G. Cheng, "On the efficiency of chaos optimization algorithms for global optimization," Chaos, Solitons & Fractals, vol. 34, pp. 1366-1375, 2007.
[36] A. H. Gandomi, G. J. Yun, X.-S. Yang, and S. Talatahari, "Chaos-enhanced accelerated particle swarm optimization," Communications in Nonlinear Science and Numerical Simulation, 2012.
[37] O. Abdel-Raouf, I. El-henawy and M. Abdel-Baset "chaotic Harmony Search Algorithm with Different Chaotic Maps for Solving Assignment Problems "International Journal of Computational Engineering & Management, Vol. 17, pp. 10-15 ,2014.
[38] O. Abdel-Raouf, I. El-henawy and M. Abdel-Baset." A Novel Hybrid Flower Pollination Algorithm with Chaotic Harmony Search for Solving Sudoku Puzzles", IJMECS, vol.6, no.3, pp.38-44, 2014.
[39] O. Abdel-Raouf, I. El-henawy and M. Abdel-Baset. "Chaotic Firefly Algorithm for Solving Definite Integral", IJITCS, vol.6, no.6, pp.19-24, 2014.
[40] O. Abdel-Raouf, I. El-henawy and M. Abdel-Baset "Improved Harmony Search with Chaos for Solving Linear Assignment Problems", IJISA, vol.6, no.5, pp.55 61, 2014.
[41] J. Mingjun and T. Huanwen, "Application of chaos in simulated annealing," Chaos, Solitons & Fractals, vol. 21, pp. 933-941, 2004.
[42] L. d. S. Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," Expert Systems with Applications, vol. 34, pp. 1905-1913, 2008.
[43] M. S. Tavazoei and M. Haeri, "Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms," Applied Mathematics and Computation, vol. 187, pp. 1076-1085, 2007.
[44] R. Hilborn, "Chaos and nonlinear dynamics, 1994," ed: Oxford University Press, New York.
[45] D. He, C. He, L.-G. Jiang, H.-W. Zhu, and G.-R. Hu, "Chaotic characteristics of a one-dimensional iterative map with infinite collapses," Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on, vol. 48, pp. 900-906, 2001.
[46] A. Erramilli, R. Singh, and P. Pruthi, Modeling packet traffic with chaotic maps: Citeseer, 1994.
[47] R. M. May, "Simple mathematical models with very complicated dynamics," Nature, vol. 261, pp. 459-467, 1976.
[48] A. Wolf, "Quantifying chaos with Lyapunov exponents," Chaos, pp. 273-290, 1986.
[49] R. L. Devaney, "An introduction to chaotic dynamical systems," 2003.
[50] R. Barton, "Chaos and fractals," The Mathematics Teacher, vol. 83, pp. 524-529, 1990.
[51] E. Ott, Chaos in dynamical systems: Cambridge university press, 2002.
[52] C. Letellier, Chaos in nature vol. 81: World Scientific Publishing Company, 2013.
[53] Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: Harmony search. Simulation vol. 76, pp. 60–68, 2001.
[54] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search 995 algorithm for solving optimization problems, Applied Mathematics and Computation 188 (2) ,pp. 1567–1579,2007.
[55] M. Mahdavi "Global-best harmony search", Applied Math. Computation vol.198, pp. 643–656, 2008.

Authors' Profiles

Ibrahim El-henawy received the M.S. and Ph.D. degrees in computer science from State University of New York, USA in 1980 and 1983, respectively. Currently, he is a professor in computer science and mathematics department, Zagazig University. ibrahim is a member of IEEE and AAEE (Australasian Association for Engineering Education) ,He is also a reviewer in different international journals and conferences. He has published more than 200 refereed articles in National and international journals, edited books, and conference proceedings. His current research interests are mathematics, operations research, statistics, networks, optimization, Intelligent Computing, Computer Theory, digital image processing, and security

Osama Abdel-Raouf received the M.S. and Ph.D. degrees in operations research and decision support systems from Monofia University. Currently, he is an associate professor in operations research department, Monofia University. His current research interests are evolutionary algorithms, artificial intelligence, and decision support systems.

Mohamed Abd El-Baset received the B.Sc. degree in information system and technology from Zagazig University in 2006, and he obtained his M.S. degree in operations research and decision support systems from Zagazig University, in 2011. Currently, he is a teaching assistant in operations research department, Zagazig University. His current research interests are characterization of probability distribution, optimization, Intelligent Computing, Evolutionary Computation and decision support systems.