2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Binary Bacterial Foraging Optimization for 0/1 Knapsack Problem

Ben Niu[1,2]
[1]College of Management, Shenzhen University,
Shenzhen, China;
drniuben@gmail.com

Ying Bi[1]
[2]The Hong Kong Polytechnic University,
Hong Kong, SAR
yingbi.szu@gmail.com

***Abstract*—Knapsack problem is famous NP-complete problem where one has to maximize the benefit of objects in a knapsack without exceeding its capacity. In this paper, a binary bacterial foraging optimization (BBFO) is proposed to find solutions of 0/1 knapsack problems. The original BFO chemotaxis equation is modified to operate in discrete space by using a mapping function, where some new variables and parameter, i.e., binary matrix *y*, logistic transformation *S*, and limiting transformation *L* is built to transform the bacterial position to a binary matrix. By using this schema, the proposed BBFO model can also be easily applied in other discrete problem solving. To further validate the efficiency of the BFO-based approach, an improved version BFO named BFO with linear decreasing chemotaxis step (BFO-LDC) is used to evaluate on six different instances. Comparisons with particle swarm optimization (PSO) and original BFO are presented and discussed.**

Keywords—Bacterial foraging; Binary; Knapsack problem

I. INTRODUCTION

Bacterial foraging optimization (BFO) is a population-based optimization technology proposed in 2002 by Passino [1]. Recently BFO has become an active area of swarm intelligence, and been studied from different angels, included parameters modification [2-3] and hybrid algorithms [4-5]. These variants of BFO were proposed based on the original Bacterial Foraging Algorithm, which used to solve continue optimization problems. However, these BFOs may have shortage of solving discrete combinatorial optimization problems, including 0/1 knapsack problem and scheduling problems. Developing a binary version of BFO to make BFO suitable for discrete domain is necessary.

The knapsack problem is a well-known NP-hard problem [6], which has very important application in the area of industry domain and financial, for example, budget controlling, investment decision-making, resource distribution, projects selection and items shipment [7]. Nowadays many intelligence methods have been proposed to solve the knapsack problem, such as Genetic Algorithm [8], Dynamic Programming [9], Simulated Annealing approach [10], Ant Colony Optimization [11], Artificial Bee Colony algorithm [12] and Particle Swarm Optimization [13] and so on.

To solve discrete combinatorial optimization problems, in this paper, we proposed a binary version of Bacterial Foraging Algorithm. The rest of this paper is organized as follows: Section II reviews the theory of BFO. Section III develops binary bacterial foraging algorithm. The knapsack problem model is described in Section IV. The step and process of BBFO solving knapsack problems is given in Section V. In Section VI, experiments are implemented, experimental results and discussions are presented. Finally, Section VII gives a conclusion.

II. BACTERIAL FORAGING OPTIMIZATION

In the process of bacterial evolution, excellent bacteria been preserved and imitated, and those with poor foraging abilities tend to be eliminated. Inspired by this evolutionary principle, BFO is proposed to simulate the foraging strategies of Escherichia coli bacteria. And the social foraging behaviors of bacteria are explored by Passino and Liu [14] form the angel of biology and physics. There are four main representative behaviors in the bacterial foraging process, namely chemotaxis, swimming, reproduction and disperse and elimination.

- Chemotaxis. A bacterium has two types of movements, swimming and tumbling. The pattern of bacteria moving is swimming along a straight line in one direction for a while, and then tumbling with a random direction. After a while, it may repeat previous moving pattern. Chemotaxis is the most important behavior of bacteria, in this process, bacteria move towards favorable chemicals as well as nutrients and away from the toxic ones. Suppose $x^{id}(j, k, l)$ represents the ith bacterium at jth chemotactic kth reproductive and lth elimination and dispersal step on dth dimension. The length of a unit swim (run) is $C(i)$. Δ is a vector represents a random direction at jth chemotactic step, and its elements value range between [-1,1]. Then the movement of the bacterium may be represented by:

$$x^{id}(j+1, k, l) = x^{id}(j, k, l) + c(id)\frac{\Delta(id)}{\sqrt{\Delta^T(id)\Delta(id)}} \quad (1)$$

- Swimming. Bacteria can achieve mutual cooperation among group and swarm together via communicate by specific forms such as quorum sensing. The description for swarming can be represented in mathematic show as follow:

$$J_{cc}(x, p(j,k,l)) = \sum_{i=1}^{n} J_{cc}^{i}(x, x^{i}(j,k,l)) + \sum_{i=1}^{n}[h_{repelent} exp(-w_{repelent} \sum_{m=1}^{p}(x_m - x_m^i)^2)] \quad (2)$$
$$= \sum_{i=1}^{s}[-d_{attract} exp(-w_{attract} \sum_{m=1}^{p}(x_m - x_m^i)^2]$$

where $j_{cc}(x, p(j, k, l))$ is the fitness value minimized to add to actual fitness function to show the change of fitness function at a time. The population of bacteria is s. p is the number of variables optimized. $d_{attract}, w_{attract}, h_{repelent}, w_{repelent}$ are different coefficients that are to be chosen properly [1,14].

- Reproduction: The bacterium with poor energy or healthy will die, and the bacterium obtains enough energy will split into two sub-bacteria at its current position. This strategy can make the population of bacteria colony remains stable and replace bacterium with poor performance by good ones.
- Elimination and Dispersal: The nutrients concentration of the current area contained may be consumed gradually by a population of bacteria or some changes occurred, such as a sudden drop in temperature. These variations may cause a number of bacteria in the area eliminate or disperse to a new area may with better nutrient concentrations. The chance of a bacterium perform dispersal and elimination can be judged by a certain probability, if the condition is satisfied, the bacteria will die and then randomly generates a new bacterium in any position in the solution space.

III. Binary Bacterial Foraging Optimization

The most important thing in the proposed BBFO model is how to express the meaning of operations such as chemotaxis in binary space. Simulation the movement of the E. coli bacterium, a bacterium has a state on each dimension. The solution to this problem is establishing the rule of restricting the position $x^{id}(j, k, l)$ to 0 and 1 on dth dimension. New variables and parameter: binary matrix y, logistic transformation $S(x^{id}(j, k, l))$, and limiting transformation L were introduced to build our BBFO model.

In BBFO model, $S(x^{id}(j, k, l))$ is logistic transformation function related to $x^{id}(j, k, l)$, L can be simply described as the probabilities of $x^{id}(j, k, l)$ changed to zero or one, $y^{id}(j, k, l)$ represents the value obtained after restrict $x^{id}(j, k, l)$. The rule described in mathematics is given as follow:

$$\begin{aligned} &if\ S(x^{id}(j,k,l)) \geq L, \quad y^{id}(j,k,l) = 1 \\ &else\ y^{id}(j,k,l) = 0 \end{aligned} \quad (3)$$

The movement trajectory of each bacterium doesn't change when the rule working, a binary matrix y is added to record results after transformation.

$S(x^{id}(j, k, l))$ is key in the rule, various versions of $S(x^{id}(j, k, l))$ can be designed to simplify computation. In our BBFO model, $S(x^{id}(j, k, l))$ inspired by the idea of discrete Particle Swarm Optimization in paper [15] is defined as follow:

$$S(x^{id}(j,k,l)) = \frac{1}{1 + exp^{-x^{id}(j,k,l)}}. \quad (4)$$

By the equation (3), the position of a bacterium is restricted within the range [0, 1], which will make the transformation probabilities easily to control.

Limiting transformation L is crucial for $x^{id}(j, k, l)$ change to 0 or 1. In other words, if L=0.5, the probabilities of $y^{id}(j, k, l)$ will become zero and one is 0.5 when $S(x^{id}(j, k, l))$ is unknown. If we set $L = rand()$ ($rand()$ is a stochastic number selected from a uniform distribution in [0, 1]), and then $S(x^{id}(j, k, l))$ =0.2, there is a twenty percent chance that $y^{id}(j, k, l)$ will be zero, and an eighty percent chance it will be one.

IV. 0/1 Knapsack Problem

Knapsack problem is a well-known NP-hard problem [6]. The simple version is studied in this paper. In this problem, supposed that there are n objects, the target is selecting a certain number of objects to maximize the total profit, while the whole capacity of objects selected is less than a fixed capacity S. So 0/1 knapsack problem [16] is expressed in mathematics as follow:

$$\begin{aligned} &Maximise \quad f = \sum_{i=1}^{n} \alpha_i y_i, \qquad i = 1,2,3,\cdots,n \\ &subject\ to\ s(y) = \sum_{i=1}^{n} c_i y_i \leq S \qquad i = 1,2,3,\cdots,n \\ &y_i = 0 \quad or\ 1 \quad i = 1,2,3,\cdots,n \end{aligned} \quad (5)$$

α_i is profit of object i, and c_i is capacity. $y_i = 0$ indicates that object i isn't selected, if $y_i = 1$, select object i.

V. BBFO Solving 0/1 Knapsack Problem

In dealing with the knapsack problem, each bacterium represents a feasible solution, the dimensions of bacterial equal to the number of objects n. Before calculate the value of fitness, the position of bacterium should be restricted in a binary matrix.

A penalty function is used to deal with the fixed capacity.

For bacterium i, its position is $x^i(j, k, l)$ and corresponding binary sequence is $y^i(j, k, l)$, calculate fitness value is $f(y^i(j, k, l))$, the capacity of selected objects is $s(y^i(j, k, l))$, if $s(y^i(j, k, l)) > S$, set $f(y^i(j, k, l)) = 0$. The steps of solving 0/1 knapsack problem with BBFO is given as follow:

- Step1: Initialize parameter and bacterial, each bacterium has a random position; record y.
- Step2: Evaluation the fitness function. The function of evaluate fitness is the objective function in the knapsack problem.
- Step3: Chemotaxis operation. Bacterial update its position, record y. Calculate the current fitness value.
- Step4: Swimming operation. Compare the current fitness value with the before, if the current fitness value is better, bacterium swimming along the original direction, then record y.
- Step5: Calculate $f(y)$ and $s(y)$. If $s(y) > S$, set $f(y) = 0$.
- Step6: Reproduction operation. Record the best value of fitness, and find the worst and the best bacterium, reproduce the best position for the worst bacterium.
- Step7: Migration operation.
- Step8: If the termination condition is satisfied, output the result.

VI. EXPERIMENTAL RESULT

A. Knapsack problems

For knapsack problems, the greater the number of objects is, the more difficult the problem to be solved is. However it will be more easily to calculate if the profit and capacity of objects are integer rather than decimal when the number of objects is same. Taking into account all of these cases, six different instances of knapsack problems are selected as benchmark functions to test the effectiveness and performance of the proposed BBFO and BBFO-LDC (binary BFO with linear decreasing chemotaxis step length in [3]). All the information including profit, capacity, fixed capacity and the number of objects of six functions is presented in Table I as follow.

B. Parameter setting

As the most important parameter L in the proposed BBFO, the value of L determines the probability of $x^{id}(j, k, l)$ restrict to one or zero. If a solution of a knapsack problem is known and many items are selected, then L should be set small to make the algorithm convergence to optimal fast. However, in most cases, we don't know the optimal solution, so experiments are conducted to verify the effects on the performance of BBFO of different value of L.

TABLE I. INFORMATION OF SIX CASES

Function	n	Profit α_i, Capacity c_i, Fixed capacity S
f_1	10	$\boldsymbol{\alpha}$ ={55, 10, 47, 5, 4, 50, 8, 61, 85, 87}; $\boldsymbol{c}$ ={95, 4, 60, 32, 23, 72, 80, 62, 65, 46}; $\boldsymbol{S}$ =269
f_2	20	$\boldsymbol{\alpha}$ ={92, 4, 43, 83, 84, 68, 92, 82, 6, 44, 32, 18, 56, 83, 25, 96, 70, 48, 14, 58}; $\boldsymbol{c}$ ={44, 46, 90, 72, 91, 40, 75, 35, 8, 54, 78 40, 77, 15, 61, 17, 75, 29, 75,63}; $\boldsymbol{S}$ =878
f_3	15	$\boldsymbol{\alpha}$ ={0.125126, 19.330424, 58.500931,35.029145, 82.284005, 17.410810, 71.050142, 30.399487, 9.140294, 14.731285, 98.852504, 11.908322, 0.891140, 53.166295, 60.176397}; $\boldsymbol{c}$ ={56.358531 , 80.874050, 47.987304, 89.596240, 74.660482, 85.894345, 51.353496, 1.498459, 36.445204, 16.589862, 44.569231, 0.466933, 37.788018, 57.118442, 60.716575}; $\boldsymbol{S}$ =375
f_4	23	$\boldsymbol{\alpha}$ = {981, 980, 979, 978, 977,976, 487, 974, 970, 485, 485, 970, 970,484, 484, 976, 974, 482, 962, 961, 959,958, 857}; $\boldsymbol{c}$ ={983, 982, 981, 980, 979, 978, 488,976, 972, 486, 486, 972, 972, 485, 485,969, 966, 483, 964, 963, 961, 958, 959}; $\boldsymbol{S}$ =10000
f_5	50	$\boldsymbol{\alpha}$ ={220, 208, 198, 192, 180, 180, 165, 162, 160, 158, 155, 130, 125, 122, 120, 118, 115, 110, 105, 101, 100, 100, 98, 96, 95, 90, 88, 82, 80, 77, 75, 73, 72, 70, 69, 66, 65, 63, 60, 58, 56, 50, 30, 20, 15, 10, 8, 5, 3, 1}; $\boldsymbol{c}$ ={80, 82, 85, 70, 72, 70, 66, 50, 55, 25, 50, 55, 40, 48, 50, 32, 22, 60, 30, 32, 40, 38, 35, 32, 25, 28, 3, 22, 50, 30, 45, 30, 60, 50, 20, 65, 20, 25, 30, 10, 20, 25, 15, 10, 10, 10, 4, 4, 2, 1}; $\boldsymbol{S}$ =1000
f_6	100	$\boldsymbol{\alpha}$ ={297, 295, 293, 292, 291, 289, 284, 284, 283, 283, 281, 280, 279, 277, 276, 275, 273, 264, 260, 257, 250, 236, 236, 235, 235, 233, 232, 232, 228, 218, 217, 214, 211, 208, 205, 204, 203, 201, 196, 194, 193, 193, 192, 191, 190, 187, 187, 184, 184, 184, 181, 179, 176, 173, 172, 171, 160, 128, 123, 114, 113, 107, 105, 101, 100, 100, 99, 98, 97, 94, 94, 93, 91, 80, 74, 73, 72, 63, 63, 62, 61, 60, 56, 53, 52, 50, 48, 46, 40, 40, 35, 28, 22, 22, 18, 15, 12, 11, 6, 5}; $\boldsymbol{c}$ ={54, 95, 36, 18, 4, 71, 83, 16, 27, 84, 88, 45, 94, 64, 14, 80, 4,23, 75, 36, 90, 20, 77, 32, 58, 6, 14, 86, 84, 59, 71, 21, 30, 22, 96, 49, 81, 48, 37, 28, 6, 84, 19, 55, 88, 38, 51, 52, 79, 55, 70, 53, 64, 99, 61, 86, 1, 64, 32, 60, 42, 45, 34, 22,49, 37, 33, 1, 78, 43, 85, 24, 96, 32, 99, 57, 23, 8, 10, 74, 59, 89, 95, 40, 46, 65, 6, 89, 84, 83, 6, 19, 45, 59, 26, 13, 8, 26, 5, 9}; $\boldsymbol{S}$ =3820

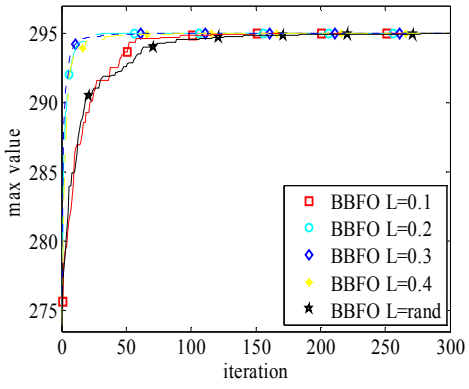

Fig.1. Performance of the BBFO under different L of f_1

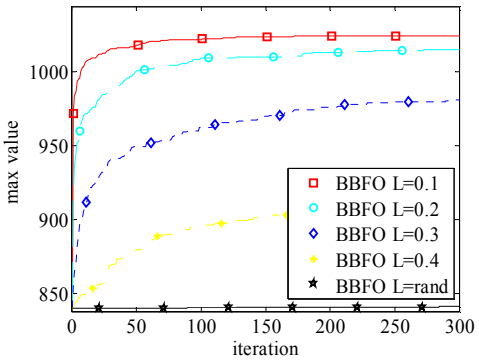

Fig.2. Performance of the BBFO under different L of f_2

Form Fig.1 and Fig.2, we found that the best value of L should be set is 0.2 in the first case, the best value of L is 0.1 in the second, and the different value of L have greater impact on the performance of BBFO in the second instance than the first one. With comprehensive comparison of the experimental results, we finally set L=0.

The performance of the proposed BBFO and BBFO-LDC is evaluated on the above mentioned 6 sets of benchmark instances, and compared with PSO.

In PSO, set the learning rate parameters $C_1 = C_2 = 2$, inertia weight were set to the value w=1 according to paper [17]. According to preliminary experiments, set chemotaxis step length $C_z = 2$ in BBFO, minimum chemotaxis step length $C_{min} = 1$, maximum chemotaxis step length $C_{max} = 4$ in BBFO-LDC. Other parameters of the two versions of BFO were set according to experiences. In order to reduce statistical errors, each experiment runs 30 times.

As mentioned in the above, dimensions=n. The max iteration of f_1-f_3 is set to 300 and the f_4-f_6 is 500 to ensure algorithms convergence to the optimal. The swarm size is set to 100 in all algorithms.

C. Experimental Results and Discussions

Results (best, worst, mean, standard deviation) of three algorithms on all tested cases for 30 runs are listed in Table II. Convergence curves are shown in Fig.3 to Fig.8.

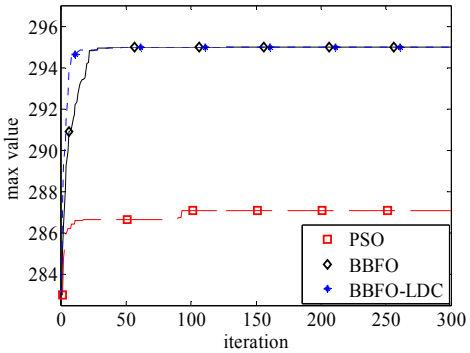

Fig.3. Convergence curve of three algorithms of f_6

TABLE II. RESULTS COMPARISON OF THREE ALGORITHMS ON ALL TESTED CASES FOR 30 RUNS

Function	Algorithm	Best	Worst	Mean	Std.
f_1	**PSO**	295	283	287.067	5.5766
	BBFO	295	295	295	0
	BBFO-LDC	295	295	295	0
f_2	**PSO**	1042	964	1014.5	47.0793
	BBFO	1042	1031	1039.5	2.9564
	BBFO-LDC	1042	1027	1037.6	5.1104
f_3	**PSO**	481.069	401.022	416.659	31.8305
	BBFO	481.069	437.934	471.959	11.6188
	BBFO-LDC	481.069	430.924	475.068	12.0541
f_4	**PSO**	9767	9730	9743.7	17.1186
	BBFO	9760	9740	9747.8	4.3739
	BBFO-LDC	9761	9745	9753.9	3.9386
f_5	**PSO**	2945	2781	2786.5	29.9422
	BBFO	2924	2793	2852.8	30.2913
	BBFO-LDC	2929	2847	2885.9	24.1460
f_6	**PSO**	14057	13392	11567	1609.3
	BBFO	13984	13597	13818	113.2849
	BBFO-LDC	14132	13615	13822	114.9347

All the results for the six instances indicate that the two BFOs have an obviously remarkable efficiency, especially for BBFO-LDC. From the perspective of the stability of algorithm, it is very clearly that BBFO and BBFO-LDC have greater superiority than PSO.

Comparing the experimental results and the convergence graph, in several cases, we found that the convergence speed of PSO is faster than the two versions of BFO, however, PSO easily trap into a local optimum. For function f_4- f_6, the best position PSO located is better than BBFO and BBFO-LDC, but the probability of success is relatively small. Unlike with PSO, the two BFOs maintained a good global search capability, as well as stability.

The performance of BBFO-LDC is better than BFO since a linear decreasing chemotaxis step length is used to ensure the bacterial convergence to global optimum quickly at the beginning and search global optimum accurately in the end.

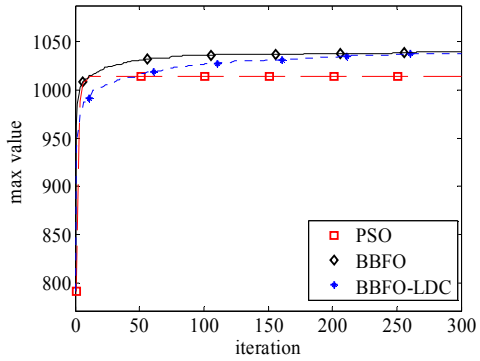

Fig.4. Convergence curve of three algorithms of f_2

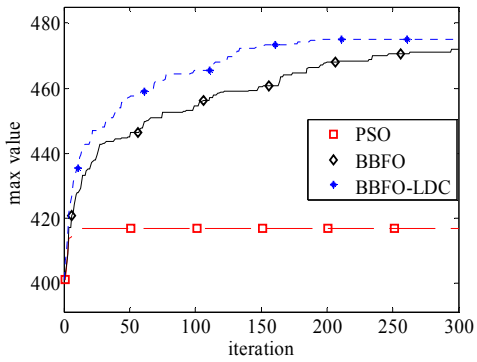

Fig.5. Convergence curve of three algorithms of f_3

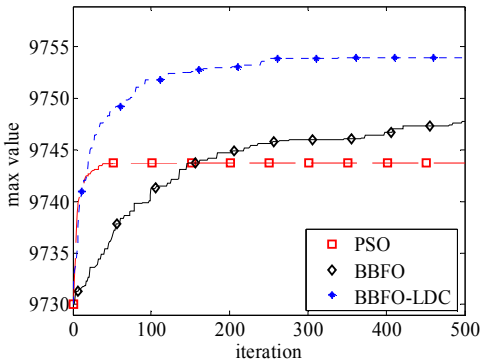

Fig.6. Convergence curve of three algorithms of f_4

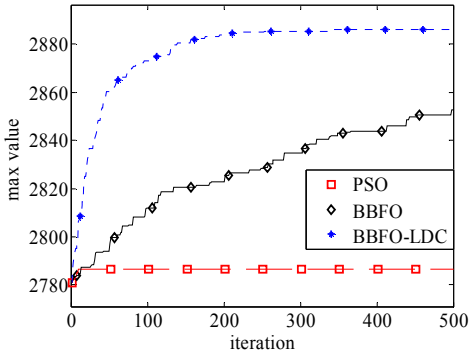

Fig.7. Convergence curve of three algorithms of f_5

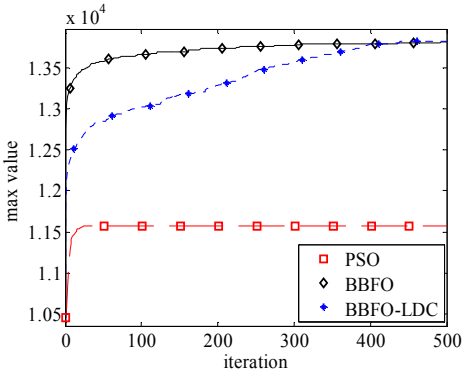

Fig.8. Convergence curve of three algorithms of f_6

VII. CONCLUSION

A novel variant of BFO called BBFO has been developed in this paper. Some new parameters and variables are designed to build BBFO model to make the position of bacterial convert to binary matrix. In order to further validate the efficiency of the BFO-based approach, an improved version of BBFO named BBFO-LDC is used to evaluate on six different instances. The final experimental results demonstrated that BBFO and BBFO-LDC be able to solve knapsack problems effectively and proved its superiority with the comparison of PSO.

However, the gap between the results the two BFOs obtained on each instance and the real optimal value is unknown, methods should be improved to get better results in the future. And then a number of other discrete optimization problems may be solved by our proposed algorithms.

ACKNOWLEDGMENT

This work is partially supported by The National Natural Science Foundation of China (Grants nos. 71001072, 71271140, 60905039), The Hong Kong Scholars Program 2012 (Grant no. G-YZ24), China Postdoctoral Science Foundation (Grant nos. 20100480705), Special Financial Grant from the China Postdoctoral Science Foundation (Grant nos. 2012T50584, 2012T50639) and the Natural Science Foundation of Guangdong Province (Grant nos. S2012010008668, S2012040007098, 9451806001002294). The authors also would like to thank The Hong Kong Polytechnic University Research Committee for financial and technical support.

REFERENCES

[1] K. M. Passino. "Biomimicry of bacterial foraging for distributed optimization and control".IEEE Control Systems Magazine, vol. 3, pp. 52-67, 2002.

[2] L. J. Tan, et al. "An adaptive comprehensive learning bacterial foraging optimization for function optimization". Communications in Computer and Information Science,vol.335, pp. 194-199, 2013.

[3] B. Niu, Y. Fan, P. Zhao, et al. "A novel bacterial foraging optimizer with linear decreasing chemotaxis step". Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on. IEEE, pp. 1-4, 2010.

[4] D. H. Kim, A. Abraham, J. H. Cho. "A hybrid genetic algorithm and bacterial foraging approach for global optimization". Information Sciences, vol. 18, pp. 3918-3937, 2007.

[5] S. M. Abd-Elazim, E. S. Ali. "A hybrid particle swarm optimization and bacterial foraging for optimal power system stabilizers design". International Journal of Electrical Power & Energy Systems, vol. 46, pp. 334-341, 2013.

[6] D. Pisinger. "Where are the hard knapsack problems"? Computers & Operations Research, vol. 9, pp. 2271-2284, 2005.

[7] W. Liu, B. Niu, H. Chen. "Binary artificial bee colony algorithm for solving 0-1 knapsack problem". 2012.

[8] P. C. Chu, J. E. Beasley. "A genetic algorithm for the multidimensional knapsack problem". Journal of heuristics, vol. 1, pp. 63-86, 1998.

[9] S. Martello, D. Pisinger, P. Toth. "Dynamic programming and strong bounds for the 0-1 knapsack problem". Management Science, vol. 3, pp. 414-424, 1999.

[10] A. Drexl. "A simulated annealing approach to the multiconstraint zero-one knapsack problem". Computing, vol. 1, pp. 1-8, 1988.

[11] M. Kong, P. Tian, Y. Kao. "A new ant colony optimization algorithm for the multidimensional Knapsack Problem". Computers & Operations Research, vol. 8, pp. 2672-2683, 2008.

[12] F. A. N. Xiao-mao, M. A. Liang. "Artificial bee colony algorithm for 0-1 knapsack problem". Mathematics in Practice and Theory, vol. 6, pp. 025, 2010.

[13] F. Hembecker, H. S. Lopes, Jr. W. Godoy. "Particle swarm optimization for the multidimensional knapsack problem". Adaptive and Natural Computing Algorithms. Springer Berlin Heidelberg, pp. 358-365, 2007.

[14] Y. Liu, K. M. Passino. "Biomimicry of social foraging bacteria for distributed optimization: models, principles, and emergent behaviors". Journal of Optimization Theory and Applications, vol. 3, pp. 603-628, 2002.

[15] C. J. Liao, C. T. Tseng, P. Luarn. "A discrete version of particle swarm optimization for flowshop scheduling problems". Computers & Operations Research, vol. 34(10), pp. 3099-3111,2007.

[16] S. Martello, P. Toth." Knapsack Problems". New York: Wiley, 1990.

[17] H. Chen, Y. Zhu, K. Hu. "Discrete and continuous optimization based on multi-swarm coevolution". Natural Computing. vol. 3, pp. 659-682, 2010.