JOURNAL OF CURRENT RESEARCH IN SCIENCE (ISSN 2322-5009)
CODEN (USA): JCRSDJ
2014, Vol. 2, No. 6, pp: 781-787
Available at www.jcrs010.com

ORIGINAL ARTICLE

A HYBRID QUANTUM-INSPIRED ARTIFICIAL BEE COLONY ALGORITHM FOR COMBINATORIAL OPTIMIZATION PROBLEM: 0-1 KNAPSACK

Kooroush Manochehri, Amir Alizadegan
Department of IT and Computer Engineering, Islamic Azad University (Parand Branch), Parand, Tehran, Iran.
Email: amiralizadegan@yahoo.com

ABSTRACT: This paper propose a new mixture method called Quantum Artificial Bee Colony (QABC) algorithm. QABC is based on some quantum computing concepts, such as qubits and superposition of states. In QABC these quantum concepts are applied on Artificial Bee Colony (ABC) algorithm. ABC is one of the recent algorithms in optimization area that has earned good popularity and some new works based on original ABC has done for optimizing various problems.For presenting the power and effectiveness of QABC, a new appropriated QABC algorithm for knapsack problem is proposed. The results of various experiments show the high performance of QABC on knapsack problem with respect to other co-classified optimization approaches. QABC is a general algorithm that can be altered for special real problems.
Key words: Optimization- Artificial Bee Colony Algorithm- Quantum Computing- Knapsack 0-1 problem

INTRODUCTION

Natural inspired algorithms that have been created for optimizing (minimizing or maximizing) problems, obtain their idea from theory of Darvean: "Existence of the stronger and better animals in the nature and so, extinction of weakers".

Evolutionary Algorithms (EAs) are a set of algorithms that uses such a biological evolution theory. The Genetic Algorithm (GA), Evolutionary Programming (EP), Evolution strategy (ES), and Genetic programming (GP) are well known and useful methods of EAs.

Later, some populated behavior of animals were concentrated by researchers and caused introducing a new set of optimization algorithms that are known as Swarm Intelligence methods. Ant Colony Optimization [Dorigo, 1992], Particle Swarm Optimization [Kennedy and Eberhart, 1995], Differential Evolution [Storn and Price, 1997], Artificial Bee Colony [Karaboga and Basturk, 2007, 2008, 2009], and Harmony Search [Lee and Geem, 2004] are the most renowned Swarm Intelligence algorithms. These algorithms based on population of solutions and by means of some techniques like stochastic and mathematic formulas, produce new solutions and maybe based on some conditions (or may not) can be replaced in place of a solution in population. Because these algorithms are robust, easy to use, and effective in producing good results, they have been used a lot in optimizing various real problems.

Quantum algorithms are known as those that must be implemented on real quantum computers and because of its difficulties, little quantum algorithms have been introduced.

Shor's quantum factoring algorithm [Shor, 1997], and Grover's database search algorithm [Grover, 1996], are famous quantum algorithms. But ordinary evolutionary and swarm intelligence algorithms can be inspired by quantum computing concepts and some merging and hybrid approaches have been proposed during last decades [Han and Kim, 2002], [Zhang et al., 2008], [Wang and Feng, 2007].

In this work, an Artificial Bee Colony (ABC) algorithm that is affected by some principles of quantum concepts, like qubits and superposition of states is presented and is called QABC is suggested. For presenting the affectivity, an adapted QABC for knapsack problem is suggested too, and some experiment is done on the knapsack problem. In the following, brief description of Artificial Bee Colony algorithm, and quantum concepts is come in sections 2 and 3, respectively. Section 4 present new Quantum Artificial Bee Colony algorithm and section 5 introduce knapsack problem and QABC algorithm for knapsack. Finally the results of various experiments are shown in sections 6.

ARTIFICIAL BEE COLONY

Artificial Bee Colony (ABC) algorithm was proposed by Karaboga for optimizing numerical problems [Karaboga and Basturk, 2007]. The algorithm simulates the intelligent foraging behavior of honeybee swarms. It is a very simple, robust and population-based stochastic optimization algorithm. Karaboga and Basturk have compared the performance of the ABC algorithm with those of other well-known modern heuristic algorithms such as Genetic Algorithm (GA), Differential Evolution (DE), Particle Swarm Optimization (PSO) on unconstrained problems [Karaboga and Basturk, 2008, 2009].

Following is the detailed pseudo-code of the ABC algorithm:

```
1: Initialize the population of solutions $x_i$, $i = 1..SN$
2: Evaluate the population
3: cycle = 1
4: **repeat**
5:     Produce new solutions $v_i$ for the employed bees by using (2) and evaluate them
6:     Apply the greedy selection process
7:     Calculate the probability values $P_i$ for the solutions $x_i$ by (1)
8:     Produce the new solutions $v_i$ for the onlookers from the solutions $x_i$ selected depending on $P_i$ and evaluate them
9:     Apply the greedy selection process
10:    Determine the abandoned solution for the scout, if exists, and replace it with a new randomly produced solution $x_i$ by (3)
11:    Memorize the best solution achieved so far
12:    cycle = cycle+1
13: **until** cycle = MCN
```

In ABC algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population. At the first step, the ABC generates a randomly distributed initial population $P(G = 0)$ of SN solutions (food source positions), where SN denotes the size of the population. Each solution $x_i (i = 1, 2, ..., SN)$ is a D-dimensional vector. Here, D is the number of optimization parameters. After initialization, the population of the positions (solutions) is subjected to repeated cycles, $C = 1, 2, ..., MCN$, of the search processes of the employed bees, the onlooker bees and scout bees. An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar amount (fitness value) of the new source (new solution). Providing the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one. Otherwise, she keeps the position of the previous one in her memory. When all employed bees complete the search process, they share the nectar information of the food sources and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. As in the case of the employed bee, she produces a modification on the position in her memory and examines the nectar amount of the candidate source. Providing that its nectar is more than the previous one, the bee memorizes the new position and forgets the old one.

An artificial onlooker bee chooses a food source depending on the probability value associated with that food source, P_i, calculated by the following expression (1):

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \quad (1)$$

where fit_i is the fitness value of the solution i which is proportional to the nectar amount of the food source in the position i, and SN is the number of food sources which is equal to the number of employed bees.

In order to produce a candidate food position from the old one in memory, the ABC uses the following expression (2):

$$v_{ij} = x_{ij} + \Phi_{ij}\left(x_{ij} - x_{kj}\right) \quad (2)$$

where $k \in \{1, 2, ..., SN\}$ and $j \in \{1, 2, ..., D\}$ are randomly chosen indices. Although k is determined randomly, it has to be different from i. Φ_{ij} is a random number between [-1, 1]. It controls the production of neighbor food sources around x_{ij} and represents the visual comparison of two food positions by a bee. As can be seen from (2), as the difference between the parameters x_{ij} and x_{kj} decreases, the perturbation on the position x_{ij} decreases, too. Thus, as the search approaches to the optimum solution in the search space, the step length is reduced adaptively.

The food source of which the nectar is abandoned by the bees is replaced with a new food source by the scouts. In ABC, this is simulated by random production of a position and replacing it with the abandoned one. In ABC, providing that a position can`t be improved further through a predetermined number of cycles, then that food source is assumed to be abandoned. The value of predetermined number of cycles is an important control parameter of the ABC algorithm, which is called "limit" for abandonment. Assume that the abandoned source is x_i and $j \in \{1, 2, \ldots, D\}$, then the scout discovers a new food source to be replaced with x_i. This operation can be defined as (3):

$$x_i^j = x_{\min}^j + rand(0,1)\left(x_{\max}^j - x_{\min}^j\right)$$

(3)

When each candidate source position v_{ij} is produced and evaluated by the artificial bee, its performance is compared with that of its old one. If the new food source has equal or better nectar than the old source, it is replaced with the old one in the memory. Otherwise, the old one is retained in the memory. In other words, a greedy selection mechanism is employed as the selection operation between the old and the candidate one. There are three control parameters in the ABC: The number of food sources which equals to the number of employed or onlooker bees (SN), the value of limit, and the maximum cycle number (MCN).

In a robust search process, exploration and exploitation processes must be carried out together. In the ABC algorithm, while onlookers and employed bees carry out the exploitation process in the search space, the scouts control the exploration process.

QUANTUM COMPUTERS

The lack of abilities of classic computers for dealing with quantum concepts in physics, caused some extensions in presenting quantum concepts and so, subsequent activities in quantum world. Here some important notions about quantum computers that are important for our work are explained briefly.

The smallest data unit in quantum computer is called quantum bit or qubit. Against classical bit that only can possess either ‘0’or ‘1’, any qubit can be in ‘0’, ‘1’, or any superposition of the two.

The state of a qubit can be represented as

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

(4)

Where α and β are complex numbers that specify the probability amplitudes of the corresponding states. $|\alpha|^2$ Gives the probability that the qubit will be found in ‘0’ state and $|\beta|^2$ gives the probability that qubit will be found in the ‘1’ state. Normalization of the state to unity guarantees

$$|\alpha|^2 + |\beta|^2 = 1$$

(5)

One qubit is defined with a pair of complex numbers, (α, β), as

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

Which is characterized by (4) and (5). An m-qubit representation is defined as:

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{bmatrix}$$

(6)

Where $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1, 2, \ldots, m$ .This representation has the advantage that it is able to represent any superposition of states. If there is, for instance, a three-qubits system with three pairs of amplitudes such as

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{2} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

(7)

Then the states of the system can be represented as

$$\frac{1}{4}|000\rangle-\frac{1}{4}|001\rangle+\frac{\sqrt{3}}{4}|010\rangle-\frac{\sqrt{3}}{4}|011\rangle+\frac{1}{4}|100\rangle-\frac{1}{4}|101\rangle+\frac{\sqrt{3}}{4}|110\rangle-\frac{\sqrt{3}}{4}|111\rangle$$
(8)

This result means that the probabilities to represent the states $|000\rangle,|001\rangle,|010\rangle,|011\rangle,|100\rangle,|101\rangle,|110\rangle,|111\rangle$ are $1/16, 1/16, 3/16, 3/16, 1/16, 1/16, 3/16, 3/16$ respectively. By consequence, the three qubits system of (7) contains the information of eight states.

Swarm intelligence and evolutionary computing with qubit representation has a better characteristic of population diversity than classical approaches, since it can represent superposition of states. Only one qubit system such as (7) is enough to represent eight states, but in binary representation at least eight string, (000), (001), (010), (011), (100), (101), (110), (111) are needed.

QUANTUM ARTIFICIAL BEE COLONY (QABC) ALGORITHM

The QABC algorithm is a general method that can be appropriated for various continuous and discrete problems. In the next section, we will show the proper QABC for knapsack problem. The pseudo code of the general QABC algorithm is shown below:

```
begin
$t = 0$
Initialize
produce $Foods(t)$ by observing $Q(t)$ states
evaluate $Foods(t)$
store the Best solution among $Foods(t)$
while (not termination-condition) do
begin
    $t = t + 1$
    For each solution in $Foods$
    begin
        Produce a new solution by observing $Foods(t)$ and $Q(t-1)$ states by Employed bees
        Evaluate new solution
        if new solution is better than the current solution in $Foods$
            then replace new solution
    end
    Calculate probability for each solution`s fitness
    For each Onlooker bee in QABC
    begin
        Choose a solution based on its fitness probability
        Produce a new solution by observing $Foods(t)$ and $Q(t-1)$ states by Onlooker bees
        evaluate new solution
        if new solution is better than the current solution in $Foods$
            then replace new solution
    end
    update $Q(t)$ using quantum gates $U(t)$
    Store the Best solution among $Foods(t)$
end
Until (t<maxcycle)
end
```

In *initialize* step, the problem to be optimized, and also some control parameters of QABC is defined and initiated. The most important part of this step is defining and launching a new structure, we call Q in this algorithm.

At iteration t, we have $Q(t)$ like as below:

$$Q(t)=\begin{bmatrix} \langle a^t_{1,1}|b^t_{1,1}\rangle & \langle a^t_{1,2}|b^t_{1,2}\rangle & \cdots & \langle a^t_{1,D}|b^t_{1,D}\rangle \\ \langle a^t_{2,1}|b^t_{2,1}\rangle & \langle a^t_{2,2}|b^t_{2,2}\rangle & \cdots & \langle a^t_{2,D}|b^t_{2,D}\rangle \\ \vdots & \vdots & \vdots & \vdots \\ \langle a^t_{n,1}|b^t_{n,1}\rangle & \langle a^t_{n,2}|b^t_{n,2}\rangle & \cdots & \langle a^t_{n,D}|b^t_{n,D}\rangle \end{bmatrix}$$
(9)

Each pair of $\langle a^t_{i,j}|b^t_{i,j}\rangle$; $1\le i\le n$, $1\le j\le D$ and $1\le t\le$ *maxcycle*, are characterized by (4) and (5). We suppose that the target problem is D-dimensional and the value of n is equal to number of solutions. All of $a_{i,j}$s and $b_{i,j}$s are initialized with $1/\sqrt{2}$. This means that each pair of $a_{i,j}$ and $b_{i,j}$ have a same probability to be chosen.

In the next step, *produce*, we create a new solution based on the characteristic of optimization problem. To make *Foods*, we use *Q*, and fill it with legal values depending on defined optimization problem. After evaluating solutions, the best one is memorized.

Then in a loop, same as ABC, each of employed and onlooker bees is responsible to produce a new solution for a corresponding solution in *Foods*. If the new solution is better than the current corresponding solution, the new solution will be replaced with the current solution.

Because of creating various new solutions in each iteration that are affected by the values of a s and b s in *Q*, we make some modifications in *Q*. This is done in update step by means of quantum gates, *G(T)* in each repetition. More details of different steps in QABC will be shown in the next section.

5. QABC FOR THE KNAPSACK PROBLEM

5.1 KNAPSACK PROBLEM

Suppose that we have m items, each has its own weight and profit, w_i and p_i, $1 \le i \le m$, respectively, and there exists a knapsack with the capacity C. A selection of items can be shown with string $x = (x_1, x_2, \ldots, x_m)$, each x_i is 1 or 0, that means item x_i is selected or not, respectively.

The aim of solving knapsack problem is to select some items from {item$_1$,item$_2$, …,item$_m$} so that:

$$f(x) = \sum_{i=1}^{m} p_i x_i$$

be maximized, subject to

$$\sum_{i=1}^{m} w_i x_i \le C.$$

KQABC

As mentioned before, QABC algorithm can be adapted for various real continuous and discrete problems. In this work, for showing the potency of new proposed QABC algorithm, we have altered QABC a little for knapsack problem and introduced new KQABC algorithm.

In the KQABC, the number of dimension`s change is subject. The approach of producing new solutions by both of employed and onlooker bees is shown below:

```
if (rand(0,1) < BP)
        choose dimension(s) j from new solution, sol
        change the value of $sol_j$ without interfering of $Q(t)$
else
        do {
        choose dimension(s) j from new solution, sol
        if ((rand(0,1) < $Q^2_{i,j}$ & $sol_j$ == 1) || (rand(0,1) > $Q^2_{i,j}$ & $sol_j$ == 0))
                change the value of $sol_j$
        } while (1)
```

and the dimension`s value in current solution is equal to 0, the dimension`s value will be changed. Else, its value will be unchanged and this process should be repeated again for another random dimension. Four states are supposed in this paper for KQABC:

1. Change One dimension`s value: in this case a dimension is selected randomly and its value is changed (KQABC#1).
2. Change two dimensions` values: in this case two dimensions are selected randomly; their values should be opposite (KQABC#2).

3. Change four dimensions` values: in this case four dimensions are selected randomly; the first two dimension`s values should be opposite to the second two dimension`s value (KQABC#3).
4. Change all dimensions` values: in this case all of the dimensions` values are changed (KQABC#4).

EXPERIMENTS, SETTINGS AND RESULTS

Any item in knapsack problem should have two quantities: weight and profit, which are selected as below:
W_i = random [1,10],
$P_i = W_i + 5$.
Also, the capacity of knapsack is set as:

$$C = \frac{1}{2}\sum_{i=1}^{m} W_i$$

Experiments are implemented for 100, 250, and 500 items on 25 independent runs with different random seed. The results of KQABC are shown in Table1 for best (b), mean (m), and worst (w) among 25 results. Also, the amount of control parameter BP is set to 0.15.

Table 1: Results of four KQABC algorithms

		KQABC#1	**KQABC#2**	**KQABC#3**	**KQABC#4**
100	**b**	623.5	612.9	622.2	611.0
	m	616.7	609.9	621.6	605.3
	w	613.5	607.8	617.2	601.2
250	**b**	1503.3	1508.7	1535.9	1469.8
	m	1493.4	1500.9	1522.2	1453.6
	w	1488.3	1493.7	1510.9	1443.7
500	**b**	2975.4	2976.5	2994.5	2860.0
	m	2960.8	2963.8	2981.4	2839.4
	w	2945.5	2951.3	2964.6	2808.5

From Table1 it is obvious that KQABC#3, changing 4 dimensions` value, has better performance with respect to other three changes. Also, KQABC#4 produces the worst results. Moreover, for number of items 100, KQABC#1 acts better than KQABC#2. But for 250 and 500, KQABC#2, changing two dimensions` value, acts better than KQABC#1, changing one dimension`s value.
In Table2 the results of KQABC#3 and GQA are compared. GQA [Han and Kim, 2000] is another optimization algorithm that inspire Genetic Algorithm with quantum concepts.

Table 2: Results of KQABC#3 and GQA

		KQABC#3	**GQA**
100	**b**	622.2	612.5
	m	621.6	603.9
	w	617.2	592.7
250	**b**	1535.9	1480.3
	m	1522.2	1467.1
	w	1510.9	1443.8
500	**b**	2994.5	2860.0
	m	2981.4	2841.3
	w	2964.6	2812.5

The results on Table2 show that KQABC#3 produces better results with respect to GQA for 100, 250, and 500 items.

CONCLUSION AND FUTURE WORKS

The new optimizing algorithm that is called QABC, was proposed in this work. The quantum properties of QABC, makes it an effective approach for dealing with various discrete and continuous problems. QABC is a general algorithm. For showing the performance and abilities of QABC, an adapted version for knapsack problem was introduced. The results on this practical benchmark show the high performance of QABC. Also, the number of changes in a new solution is important. In this work, four dimensions` change was suitable for Knapsack problem. The results of experiments approved that changing four dimensions` value produce better results among four mentioned states. For future works, more number of items choosing, and also other ways of inspiring quantum concepts are scopes of study.

ACKNOWLEDGMENT

The authors would like to appreciate Parand Azad University for their funded and scientific supports of this study.

REFERENCES

Dorigo M. Optimization, learning and natural algorithms (in Italian), PhD Thesis, Politecnico di Milano, Italy. 1992.

Grover LK. A fast quantum mechanical algorithm for database search, in: Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, Philadelphia, 1996: pp. 212–219.

Han KH, Kim JH. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Transaction on Evolutionary Computation 2002: 6: 580–593.

Han KH, Kim JH. Genetic quantum algorithm and its application to combinatorial optimization problem, Evolutionary Computation, Proceedings of the 2000: Congress. pp: 1354–1360.

Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm, J GLOBAL OPTIM 2007: 39: 459–471.

Karaboga D, Basturk B. On the performance of Artificial Bee Colony (ABC) algorithm, APPL SOFT COMPUT 2008: 8: 687–697.

Karaboga D, Basturk B. A comparative study of Artificial Bee Colony algorithm, APPL MATH COMPUT 2009: 214: 108-132.

Kennedy J, Eberhart RC, Particle swarm optimization, in: Proc. IEEE Int. Conf. on Neural Networks, WA, Australia, 1995: pp. 1942–1948.

Layeb A. A hybrid quantum inspired harmony search algorithm for 0–1 optimization problems. Journal of Computational and Applied Mathematics 2012: 253: 14–25.

Lee KS, Geem ZW. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Computer Methods in Applied Mechanics and Engineering 2004:194: 3902-3933.

Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Journal on Computing 1997:26: 1484–1509.

Storn R, Price K. Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. J GLOBAL OPTIM 1997: 11: 341–359.

Wang Y, Feng XY. A novel quantum swarm evolutionary algorithm and its applications. Neurocomputing 2007: 70: 633-640.

Zhang G, Gheorghe XM, Wu CZ. A Quantum-Inspired Evolutionary Algorithm Based on P systems for Knapsack Problem. Fundamenta Informaticae 2008: 87: 1-24.