*Journal of **AI** and **D**ata **M**ining*
Vol 6, No 1, 2018, 133-143

BQIABC: A new Quantum-Inspired Artificial Bee Colony Algorithm for Binary Optimization Problems

F. Barani[1*] and H. Nezamabadi-pour[2]

1. Department of Computer Engineering, Higher Education Complex of Bam, Bam, Iran.
2. Department of Electrical Engineering, Shahid Bahonar University of Kerman, Street, Bam, Iran.

Received 06 December 2016; Accepted 21 February 2017
**Corresponding author: f.barani@bam.ac.ir(F. Barani).*

Abstract
Artificial bee colony (ABC) algorithm is a swarm intelligence optimization algorithm inspired by the intelligent behavior of honey bees when searching for food sources. Various versions of the ABC algorithm have been widely used to solve continuous and discrete optimization problems in different fields. In this paper, a new binary version of the ABC algorithm inspired by quantum computing called binary quantum-inspired artificial bee colony algorithm (BQIABC) is proposed. BQIABC combines the main structure of ABC with the concepts and principles of quantum computing such as quantum bit, quantum superposition state, and rotation Q-gates strategy to make an algorithm with more exploration ability. Due to its higher exploration ability, the proposed algorithm can provide a robust tool to solve binary optimization problems. To evaluate the effectiveness of the proposed algorithm, several experiments are conducted on the 0/1 knapsack problem, Max-Ones, and Royal-Road functions. The results produced by BQIABC are compared with those of ten state-of-the-art binary optimization algorithms. Comparisons show that BQIABC presents better results than or similar to other algorithms. The proposed algorithm can be regarded as a promising algorithm to solve binary optimization problems.

Keywords: Artificial Bee Colony Algorithm, Quantum Computing, Rotation Q-gate, 0/1 Knapsack Problems, Benchmark Functions.

1. Introduction

The artificial bee colony (ABC) algorithm is a population-based optimization algorithm, which was developed by Karaboga in 2005 [1]. The ABC algorithm was motivated by the intelligence foraging behavior of real bees. This algorithm, due to easy implementation, low number of control parameters, and rapid convergence, has attracted the attention of many researchers to itself. In [2] and [3], the performance of the ABC algorithm has been compared with those of Genetic Algorithm (GA), Particle Swarm Algorithm (PSO), Differential Evolution (DE), Evolutionary Algorithm (EA), and Particle Swarm Inspired Evolutionary Algorithm (PS-EA) on some well-known test functions. The comparison results have indicated that ABC may be regarded as a promising algorithm to solve optimization problems.

The standard versions of evolutionary algorithms are basically used for solving continuous problems. H. Gökdağ et al. [4] have applied the PSO algorithm to detect the damaged structural elements of a Timoshenko beam. To solve the multi-pass turning optimization problems, optimization methods based on the ABC algorithm [5] and hybrid robust differential evolution (HRDE) [6] by A.R. Yildiz have been developed. Also to optimize cutting parameters in milling operations, a hybrid method based on the differential evolution algorithm and immune system has been presented [7], and a method based on the cuckoo search algorithm, reported in [8], has been introduced. I. Durgun et al. [9] and B. Yildiz et al. [10] have applied the cuckoo search algorithm and the gravitational search algorithm, respectively, to solve the problem of

the optimum design of a vehicle component. Z. Izakian et al. [11] applied the particle swarm optimization algorithm for clustering time series data.

However, most optimization problems in the engineering fields are set in binary space. Different binary and discrete versions of evolutionary algorithms have been presented to solve the binary encoded problems such as the 0/1 knapsack problem, feature selection, and benchmark binary functions. H. Shi [12] has adopted the ant colony optimization, A. Lui et al. [13] have improved the simulated annealing algorithm using two kinds of solution spaces, Z. Li et al. [14] have proposed a novel binary particle swarm optimization based on the QBPSO algorithm by applying a multi-mutation strategy including single mutation operator and full mutation operator, and H. Sajedi et al. [15] have proposed a discrete gravitational search algorithm (DGSA) based on a new method for discretely updating the position of the agents for solving the 0/1 knapsack problem. S. Sunder et al. [16] have introduced a hybrid artificial bee colony algorithm called ABC-MKP, and M. Kong et al. [17] have proposed a binary ant system (BAS) based on a pheromone-laying method for solving the 0/1 multi-dimensional knapsack problem. In [18], the quantum-inspired binary gravitational search algorithm (BQGSA), and in [19], a modified GSA have been applied to solve the feature subset selection problem in data classification.

The binary artificial bee colony (BABC) algorithms can be divided into two categories, the BABC algorithm based on the continuous space and the discrete space [20]. In the first category, algorithms are based on the standard ABC algorithm and apply its formulae and operation rules to solve binary encoded problems by mapping the discrete space to the continuous space. The BinABC algorithm, proposed by Y. Marinakis et al. [21], AMABC, and NormABC, proposed by G. Pampara et al. [22] are in this category. However, it should be noted that these maps have a high complexity. The algorithms in the second category modify the food source generating formula, and replace the bit operation with the traditional vector operation. The DisABC algorithm proposed by M.H. Kashani et al. [23] is in this category.

Quantum computing (QC) is the art of applying the laws of quantum mechanics to computer science [24]. R. Feynman [25] and D. Deutsch [26] have proposed the initial idea of QC in the early 1980s. In solving computational problems, QC is stronger than classical computing. Since the late 1990s, many studies have been done on the integration of evolutionary algorithms and quantum computing. The kind of algorithms can be classified into three categories, as follow [27]:

- Evolutionary-designed quantum algorithms, which aim to automate the combination of new quantum algorithms using EAs for quantum computers (e.g. [28, 29]).
- Quantum evolutionary algorithms (QEAs) concentrate on execution of EC algorithms in a quantum computation environment (e.g. [30-32])
- Quantum-inspired evolutionary algorithms (QIEAs) concentrate on execution of new EC algorithms using some concepts and principles of QC (e.g. [27, 33-35]).

Y. Jeong et al. [31] have proposed a quantum-inspired binary particle swarm optimization, and have applied it to solve unit commitment problems for power systems. H.B. Duan et al. [36] have combined the artificial bee colony algorithm and the quantum evolutionary algorithm quantum, introducing a hybrid method, and also N.H. Abbas et al. [37] have presented a quantum ABC inspired by quantum physics concepts to solve the continuous optimization problems. K. Han et al. [38] have introduced a quantum-inspired evolutionary algorithm for a class of combinatorial optimization, and G. Zhang has presented a comprehensive survey of the recent work in the field of quantum-inspired algorithm. To enhance the performance of ABC algorithm, G. Li et al. [39] have integrated QC into ABC and used Q-bits described on the Bloch sphere to solve continuous problems. Also X. Yuan et al. [40] have integrated ABC with QC and the chaotic local search strategy to solve the optimal power flow problem. K. Manochehri et al. [41] have presented a quantum ABC, called KQABC, by an unclear relation to produce new food sources, and have adopted it to solve 0/1 knapsack problem. Also the updating process of Q-bits in the KQABC is not clearly defined. M. Soleimanpour et al. [32] have proposed a quantum-behaved gravitational search algorithm, and H. Nezamabadi-pour [27] has presented a binary quantum-inspired GSA to solve binary encoded problems.

The admirable results achieved from combination of the evolutionary algorithms and quantum computing persuaded us to develop a new quantum-inspired version of the ABC algorithm called binary quantum-inspired artificial bee colony algorithm (BQIABC) to effectively solve combinatorial problems in binary space by applying the some concepts of quantum

computing such as a quantum bit and superposition of states in the standard ABC algorithm. The use of quantum computing in ABC has enabled BQIABC to improve the exploration ability and convergence rate in obtaining optimum solutions.

The main contributions of this paper are summarized as the following:

- The artificial bee colony algorithm is combined with the quantum computing, and a binary quantum-inspired algorithm is presented to solve the binary-encoded problems.
- Binary quantum-inspired artificial bee colony algorithm (BQIABC) is proposed based on the concepts and principles of quantum computing such as a quantum bit, superposition of states, and a new rotation gate.
- In this work, a combination of ABC and QC is used to improve the convergence rate and exploration ability, and prevent trapping in local optima.
- In accordance with the ABC algorithm, a new rotation Q-gate is proposed, which determines rotation angle based on the current position and the best position so far.
- In BQIABC, a new behavior for scout bee is suggested to replace the abandoned food sources.
- A comparative study is carried out with ten other binary optimization algorithms to emphasize on the effectiveness of BQIABC algorithm.

The remainder of this paper is organized as follows. A brief review on the artificial bee colony algorithm and quantum computing that is used in this study is presented in Section 2. Section 3 presents the proposed algorithm. In Section 4, we present the experimental results and comparison with ten other algorithms on several cases of 0/1 knapsack problem, Max-Ones, and Royal-Road functions. Finally, a brief conclusion is offered in Section 5.

2. Background Knowledge

In this section, we review some background knowledge required for an easier understanding of our proposed algorithm, in brief, including artificial bee colony algorithm and quantum computing.

2.1. Artificial Bee Colony Algorithm

The artificial bee colony (ABC) algorithm [8] is a member of the class of swarm intelligence algorithms, proposed by Karaboga in 2005. The ABC algorithm is motivated by the natural behaviour of honey bees when searching for food sources. The colony of artificial bees consists of three different types of bees: employed, onlooker, and scout bees [42]. The number of employed and onlooker bees in the colony is the same, and is equal to the number of food sources around the hive. A potential solution in the optimization problem corresponds to the position of each food source, and the solution fitness is its nectar amount.

In this algorithm, the initial position of every food source is randomly generated by scout bee, and each employed bee is assigned to a food source.

$$x_{ij} = x_i^{\min} + r(x_i^{\max} - x_i^{\min}) \quad (1)$$

where, $x_i^{\min}$ and $x_i^{\max}$ are the minimum and maximum values of the *j*th dimension, respectively, and *r* is a random number in the interval [0,1].

At each iteration *t* of the algorithm, the employed bee *j* selects a food source x_w randomly and discovers a new food source v_j around its assigned food source x_j, as follows:

$$v_{jk}(t+1) = x_{jk}(t) + \varphi_{jk}(t)(x_{jk}(t) - x_{wk}(t)) \quad (2)$$

where, $\varphi_{jk}(t)$ is a random number in the interval [-1,1] $x_{jk}(t)$ indicates the position of the *j*th food source in the *k*th dimension, and v_{jk} presents the position of the new food source *j* in the dimension *k*. The nectar amount of new food source v_j is computed. If v_j has a further nectar amount, the current food source x_j will be replaced by it.

$$x_j(t+1) = \begin{cases} v_j(t+1) & if\ fit(x_j(t)) < fit(v_j(t+1)) \\ x_j(t) & if\ fit(x_j(t)) \geq fit(v_j(t+1)) \end{cases} \quad (3)$$

There is only one scout bee in the colony. The employed bee whose food source has been abandoned becomes a scout bee and carries out a random search to find a new food source. After finishing the search process by all the employed bees, they perform a waggle dance in the hive to share the information about the nectar amount and the position of food sources with onlooker bees. Each onlooker bee chooses a food source based on a probability value associated with it, and explores a new food source around the selected food

source. The probability value of food source *i* is calculated as follows:

$$p_i = \frac{fit_i}{\sum_{j=1}^{SN} fit_j} \quad (4)$$

where, fit_i and *SN* are the fitness value of food source *i* and the number of food sources or the colony size, respectively. The pseudo-code of the ABC algorithm is given in figure 1.

ABC algorithm

Initialize
Repeat
  Send the employed bees onto their food sources and determine their nectar amounts
  Send the onlooker bees onto the food sources based on their probability and determine their nectar amounts
  Send the scout bees for searching new food sources
  Memorize the best food source found so far
Until (a termination condition is met)

Figure 1. Pseudo-code of ABC algorithm.

2.2. Quantum computing

The smallest unit of information in digital computers is a bit representing either 0 or 1 at a certain time, whereas Q-bit or quantum bit is the smallest unit of information in quantum computing. Each Q-bit is able to be in states "0", "1" or a combination of both states at the same time. This is known by superposition. A Q-bit is indicated as a pair of numbers (α, β), where the values for $|\alpha|^2$ and $|\beta|^2$ denote the probability of discovering the Q-bit in the states "0" and "1", respectively. The state of a Q-bit is presented as follows:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (5)$$

Each Q-bit should satisfy the following normalization equation:

$$|\alpha|^2 + |\beta|^2 = 1 \quad (6)$$

In a quantum computer, an individual *q* is denoted by a sequence of *n* Q-bits, as follows [43]:

$$q = [q_1, q_2, ..., q_n] = \begin{bmatrix} \alpha_1 & \alpha_2 & ... & \alpha_n \\ \beta_1 & \beta_2 & ... & \beta_n \end{bmatrix} \quad (7)$$

In the act of observing a quantum state, it collapses to a single state. The observation process of Q-bit *i* is performed as follows:

$$\begin{aligned} &if && rand(0,1) < (\alpha_i)^2 \\ &then && f_i = 0 \\ &else && f_i = 1 \end{aligned}$$

Figure 2: Observation process in a quantum system.

Quantum computers apply a sequence of quantum operations to update the values for the Q-bits in each individual such that the updated Q-bits should satisfy Eq. (7). Q-gate is one of the quantum operations to update Q-bits. There are various Q-gates such as NOT gate, controlled NOT gate, rotation gate, Hadamard gate, X-gate, Y-gate, and Z-gate [24]. In most studies, the rotation Q-gate is employed more than other Q-gates. The rotation Q-gate $U(\Delta\theta_i)$ is defined as follows [38]:

$$U(\Delta\theta_i) = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix} \quad (8)$$

where $\Delta\theta_i$ is the rotation angle of Q-bit *i* toward either 0 or 1 state. The state of Q-bit *i* at time *t* is updated as follows:

$$\begin{bmatrix} \alpha_i(t+1) \\ \beta_i(t+1) \end{bmatrix} = U(\Delta\theta_i) \begin{bmatrix} \alpha_i(t) \\ \beta_i(t) \end{bmatrix} \quad (9)$$

3. Proposed Algorithm

The BQIABC algorithm was introduced by applying some concepts of quantum computing such as quantum bits, quantum gates, and superposition of states in the main structure of the ABC algorithm. BQIABC preserves the basic structure of the standard ABC algorithm and its main ideas, and replaces the concept of position of food sources and their updating process with new concepts. The position of each food source is defined as a Q-bit vector of length *n*. Each element in a food source takes a value of 0 or 1 by the probability of $|\alpha|^2$ or $|\beta|^2$. In other words, by observing a quantum state, it collapses to a single state [27]. The observation process is given by figure 2. The steps of the BQIABC algorithm are as follows:

i. ***Initialization***: The set $FB(t) = \{B_1(t), ..., B_{SN}(t)\}$ is an archive set to hold the best binary solution achieved by the onlooker and employed bees through iterations of BQIABC. At iteration $t = 0$, the set $FB(0) = \{\}$. In this step, the set $Q(t)$ of *SN* quantum food sources in a *n*-dimensional search space is randomly generated such that the normalization equation (6) is satisfied.

$$Q(t) = \{q_1(t), q_2(t), ..., q_{SN}(t)\} \quad (10)$$

where, $q_i(t)$ is the quantum food source *i* that is defined as follows:

$$q_i(t) = \begin{bmatrix} \alpha_i^1(t) & \alpha_i^2(t) & ... & \alpha_i^n(t) \\ \beta_i^1(t) & \beta_i^2(t) & ... & \beta_i^n(t) \end{bmatrix} \quad (11)$$

The value $\alpha_i^d \ \ (d = 1,...,n)$ is initially set to $\frac{1}{\sqrt{2}}$ and the value β_i^d is calculated by equation $\left|\beta_i^d\right|^2 = 1 - \left|\alpha_i^d\right|^2$.

ii. ***Observation***: the set $FW(t) = \{F_1(t), F_2(t), ..., F_{SN}(t)\}$ contains the *SN* binary food sources (current solutions) that are made by observing on each Q-bit $q_i(t)$ in the set $Q(t)$. The binary food source i is presented as $F_i(t) = \left[f_i^{\,1}, f_i^{\,2}, ..., f_i^{\,n}(t)\right]$ where $f_i^{\,d} \in \{0,1\}$. The observation process on the *i*th quantum food source q_i is carried out, as shown in figure 2.

iii. ***Fitness evolution***: in this step, the fitness value of each binary food source $F_i(t)$ in the set $FW(t)$ is evaluated using function fit.

iv. ***Updating*** $FB(t)$: in this step, *N* binary food sources with the highest fitness value are selected from all food sources in the set $FW(t)$ and $FB(t)$ and are replaced by the previous food sources in $FB(t)$. At $t = 0$, all binary food sources in $FW(0)$ are transferred into $FB(t)$.

v. ***Employed bees***: in this step, each employed bee i selects randomly a food source $F_w(t) \in FB(t)$ and then the rotation angle $\Delta\theta_i$ is calculated by Eq. (12). The employed bee i updates the position of quantum food source $q_i(t)$ using the rotation Q-gate in Eqs. (8, 9) and explores a new quantum food source q_i in the neighborhood of $q_i(t)$. The amount of movement towards 0 or 1 is denoted by $\Delta\theta_i$.

$$\Delta\theta_i^j = \theta(f_i^{\,j} - b_w^j) \tag{12}$$

where, $f_i^{\,j}$ and b_w^j are the value of *j*th dimension of the food sources F_i and B_w, respectively, and θ denotes the magnitude of the rotation angle. To enhance the convergence of the proposed algorithm, we employed a dynamic rotation angle approach used in [43] to calculate θ.

$$\theta = \theta_{\max} - (\theta_{\max} - \theta_{\min}) \times \frac{t}{iter_{\max}} \tag{13}$$

where, *t* and $iter_{\max}$ are the current iteration number and the maximum iteration number, respectively. Based on the above equation, the value for θ changes monotonously from $\theta_{\max}$ to $\theta_{\min}$. In general, the value from 0.05π to 0.001π is considered for θ. The considered values for θ are dependent on problem [35].

Then the observation process is applied to the quantum food source q_i and is makes the binary food source F_i. If F_i has a more nectar amount, the previous food source F_i will be replaced by F_i.

$$\begin{bmatrix} \alpha_i^j(t+1) \\ \beta_i^j(t+1) \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta_i^j(t)) & -\sin(\Delta\theta_i^j(t)) \\ \sin(\Delta\theta_i^j(t)) & \cos(\Delta\theta_i^j(t)) \end{bmatrix} \begin{bmatrix} \alpha_i^j(t) \\ \beta_i^j(t) \end{bmatrix} \tag{14}$$

vi. ***Onlooker bees***: in this step, the selection probability of each food source is calculated by Eq. (4). Then the onlooker bee *i* chooses a food source based on the probability values associated with food sources, and explores a new food source around the selected food source similar to the employed bees.

vii. ***Scout bee***: if the number of sequential unsuccessful attempts to improve the fitness value of a food source is higher than the given value *limit*, it is considered as an abandoned food source. In this situation, the scout bee replaces the worst food source in the current iteration with the best food source in that iteration.

viii. ***Repeat***: the steps (ii)-(vii) are repeated until the stopping criterion is met.

The pseudo-code of the proposed algorithm is presented in figure 3.

4. Experimental results

In this section, vast experiments are carried out to assess the performance of our proposed algorithm in order to solve the binary encoded optimization problems. In this study, the Max-Ones, Royal-Road functions, and the 0-1 knapsack problem are considered as well-known benchmark binary problems. The BQIABC algorithm will be compared with ten binary-valued algorithms, which have been applied on the 0/1 knapsack

problems, the Max-Ones, and Royal-Road functions to validate the superiority of the proposed algorithm. The experimental results are summarized in tables 2-5. In the following sections, the benchmark binary problems will be clarified in detail. It is noteworthy that all experiments have been implemented on the Matlab environment on a system with 2.40 GHz CPU and 4 GB of RAM.

BQIABC algorithm

$t = 0, FB(t) = \{\}, FW(t) = \{\}$

Initialize $Q(t)$ (Eq. 10 and 11)

Repeat

Observe $Q(t)$ and make $FW(t)$ (Fig. 2)

Calculate fitness values of $F_i(t) \in FW(t)$

Update $FB(t)$

for each employed bee i do

Generate a new quantum food source q_i in the neighborhood of q_i using Equations (12,13,14)

Observe q_i and make F_i by (Fig. 2)

Calculate fitness value of f_i

if $fit(B_i) > fit(B_i)$ then

$q_i = q_i$

$B_i = B_i$

end if

end for

for each onlooker bee i do

Calculate the probability of food sources using Eq. 4

Select a quantum food source q_j based on probability values

Generate a new quantum food source q_j in the neighborhood of q_j using Equations (12,13, 14)

Observe q_j and make B_j by (Fig. 2)

Calculate fitness value of f_j

if $fit(B_j) > fit(B_j)$ then

$q_j = q_j$

$B_j = B_j$

end if

end for

Determine abandoned food source and replace it with a new quantum food source for the scout bee

Memorize the best food source found so far

$t = t + 1$

Until (a termination condition is met)

Figure 3. Pseudo code of BQIABC algorithm.

4.1. Knapsack problem

The 0/1 knapsack problem is one of well-known binary encoded optimization problems. Given a set of *N* objects, where each object i having a weight w_i and a profit p_i and a knapsack with limited weight capacity *C*. The aim of problem is filling the knapsack with a subset of objects in such a way that the sum of weight of selected objects does not exceed the specified capacity of knapsack, whereas their profit is maximized. The 0/1 knapsack problem can be explained as follows:

$$Maximize: \sum_{i=1}^{n} p_i x_i \tag{15}$$

Subject to the constraint:

$$\sum_{i=1}^{n} w_i x_i \le C, x_i \in \{0,1\} \tag{16}$$

where, x_i is 0 or 1. If x_i takes the value of "1", the object i is selected, otherwise the object is not selected for knapsack.

In the recent years, researchers have proposed several exact methods based on branch and bound, dynamic programming, and heuristic methods to deal with the knapsack problems. In this study, five test cases of the 0/1 knapsack problem with 50, 200, 400, 600, and 1000 items are employed to assess the BQIABC algorithm. All test cases are created by strongly correlated sets of data [38]. The unsorted data considered by Zhang [43] on the knapsack problems with 50, 200, and 400 items and the unsorted data has been considered by Nezamabadi-pour [27] on the knapsack problems with 600 and 1000 items. Also some of experiments reported on them are used for comparison between the proposed algorithm and other algorithms.

4.2. Binary benchmark functions

Max-ones and Royal-Road are maximization benchmark functions in binary space (see Table 1). In this study, the Max-Ones function with dimensions $n = 40, 80, 160, 320, 640$ and the Royal-Road function with dimensions $n = 40, 80, 160, 320$ were used to assess the performance of the BQIABC algorithm.

4.3. Comparative algorithms

To confirm the superiority of the proposed algorithm, it was compared with ten binary encoded heuristic algorithms. The proposed algorithm was applied on the Max-Ones, Real-Road functions, and the 0/1 knapsack problem. For comparison, we implemented KQABC [41] and used the results reported by Nezamabadi-pour [27] for the binary quantum-inspired particle swarm optimization (BQIPSO) [44], binary particle swarm optimization (BPSO) [45], modified binary particle swarm optimization (MBPSO) [46], binary gravitational search algorithm (BGSA) [47], and novel binary

differential evolution NBDE [48] and the results given by Zhang [43] for the original binary quantum-inspired evolutionary algorithm (BQIEAo) [38], modified BQIEA by incorporating cross-over and mutation operators (BQIEAcm) [49], modified BQIEA by introducing new rotation Q-gate strategy (BQIEAn) [50] and conventional genetic algorithm (CGA).

Table 1. Binary benchmark functions.

Name	Function	S
Max-Ones	$f(X_i)=\sum_{d=1}^{n} X_i^d$	$\left[\{0,1\}\right]^n$
Royal-Road	$f(X_i)=\sum_{d=1}^{\frac{n}{8}}\left(\prod_{j=8(d-1)+1}^{8d} X_i^d\right)$	$\left[\{0,1\}\right]^n$

4.4. Comparative study

The comparison between binary algorithm is performed based on three criteria, the best, mean, and the worst solution found. The comparison results are reported as the mean of 30 independent runs. In BQIABC, the maximum number of iterations and the parameter value of *limit* are set to 1000 and 200, respectively. The colony size is considered to be 20.

The θ_{max} and θ_{min} value are set to 0.05π and 0.001π, respectively. Table 2 presents the comparison results of the performance of BQIABC with those of CGA, BPSO, MBPSO, NBDE, BGSA, BQIEAo, BQIEAcm, BQIEAn and BQIPSO on solving the knapsack problem with 50, 200, and 400 items.

Rows captioned by "BS", "MBS", and "WS" report the best of the best profits, mean best profits, and worst of the best profits over 30 runs.

Table 2. Comparison between BQIABC and other algorithms using the knapsack problems with 50, 200, and 400 items.

Items	Criteria	CGA	BPSO	MBPSO	NBDE	BGSA	BQIEAo	BQIEAcm	BQIEAn	BQIPSO	BQIABC
	BS	296.45	307.05	302.23	301.98	312.16	312.17	312.13	307.25	312.22	**312.23**
50	MBS	287.29	303.22	297.57	298.41	307.32	307.40	306.86	304.24	307.67	311.09
	WS	282.00	297.21	295.25	296.03	306.74	307.21	302.24	299.23	302.23	310.33
	BS	1047.98	1107.93	1078.27	1082.35	1147.92	1178.22	1173.18	1102.08	1193.31	**1198.22**
200	MBS	1027.13	1089.85	1069.09	1070.09	1120.18	1166.67	1156.22	1090.64	1184.91	1188.79
	WS	1017.15	1078.04	1058.55	1060.72	1078.29	1153.27	1143.20	1077.45	1178.23	1185.69
	BS	2120.54	2210.46	2175.19	2181.15	2255.59	2341.36	2336.41	2211.12	2396.42	**2410.94**
400	MBS	2100.85	2190.18	2156.64	2164.15	2219.28	2322.47	2315.92	2190.67	2380.10	2407.08
	WS	2086.29	2175.00	2140.68	2146.38	2191.14	2300.49	2291.25	2165.62	2366.02	2395.89

The results summarized in table 2 denote that BQIABC can present the better solutions than the other binary algorithms for all test cases. The largest difference in the performance between BQIGSA and other binary algorithms occurs in the test case with 400 items. It implies that BQIABC in more complex problems, due to its higher exploration ability, provides better results than the other binary algorithms.

Also we carried out experiments on the higher-dimensional cases of the 0/1 knapsack problem with 600 and 1000 items. The comparison results of BQIABC with BPSO, BGSA, and BQIPSO are tabulated in table 3.

From table 3, it can be observed that BQIABC finds the solutions with higher profits in comparison with the other comparative binary algorithms. As it can be seen, the distinction between the results found by BQIABC and the other comparative algorithms is very noticeable.

In [41], a basic version of quantum ABC, called KQABC, is presented to solve the 0/1 knapsack problem. Authors used a raw relation to produce and update food sources.

To confirm the superiority of BQIABC to the KQABC algorithm in solving the 0/1 knapsack problem, the KQABC algorithm was implemented based on the description provided in that paper, and the results obtained were compared with the proposed algorithm. These algorithms were implemented in the same conditions. The results obtained are tabulated in table 4.

Table 3. Comparison between BQIABC and some binary algorithms using knapsack problems with 600 and 1000 items.

Items	Criteria	BPSO	BGSA	BQIPSO	BQIABC
600	BS	3265.45	3415.23	3564.97	**3585.23**
	MBS	3242.03	3379.19	3545.32	3577.02
	WS	3219.77	3318.24	3519.56	3573.43
1000	BS	5349.12	5439.99	5866.55	**5929.91**
	MBS	5328.96	5403.17	5832.95	5911.22
	WS	5306.90	5318.24	5803.74	5887.89

Table 4. Comparison between BQIABC and KQABC.

	item / criteria	50	200	400	600	1000
KQABC	BS	279.26	1028.22	2087.63	3000.12	5229.7
	MBS	276.02	1022.94	2087.59	3100.05	5205.51
	WS	272.77	1017.74	2087.57	3100.03	5181.29
BQIABC	BS	**312.23**	**1198.22**	**2410.94**	**3585.23**	**5929.91**
	MBS	311.09	1188.79	2407.08	3577.02	5911.22
	WS	310.33	1185.69	2395.89	3573.43	5887.89

Table 4 denotes that the results found by BQIABC in comparison with KQABC have very large distinctions, and our proposed algorithm can provide better solutions for the 0/1 knapsack problem. The performance of BQIABC, BPSO, BGSA, and BQIPSO in solving two well-known binary benchmark functions with several sizes was compared in table 5. The binary functions are given in table 1. The Max-Ones function with different sizes $n = 40, 80, 160, 320, 640$ and the Royal-Road function with different sizes $n = 40, 80, 160, 320$ are considered to assess the performance of the proposed algorithm. For small size cases on solving the Max-Ones, BGSA, and BQIPSO can be found optimal solutions but by increasing the size of problem, their performance is reduced. This reduction is very sensible for BGSA. In Max-Ones with sizes of 320 and 640, there was a considerable difference between BQIABC and other comparative algorithms. The results obtained for BGSA and BQIABC are almost the same on solving the Royal-Road with size of 40 and 80. BQIABC is able to outperform BGSA, BQIPSO, and BPSO in the sizes of 160 and 320. In all sizes of Royal-Road, there was a significant distinction between BQIABC and BPSO, and between BQIGSA and BQIPSO.

Table 5. Comparison between BQIABC and some binary algorithms using Max-Ones and Royal-Road with different sizes.

Function	Criteria	BPSO [43]	BGSA [47]	BQIPSO [44]	BQIABC
Max-Ones(40)	BS	**40 (1)**	**40 (1)**	**40 (1)**	**40 (1)**
	WS	38	40	40	40
	MBS	39.251	40	40	40
Max-Ones(80)	BS	74 (2)	**80 (1)**	**80 (1)**	**80 (1)**
	WS	69	79	80	80
	MBS	71.15	79.65	80	80
Max-Ones(160)	BS	129 (2)	**160 (1)**	**160 (1)**	**160 (1)**
	WS	123	153	160	157
	MBS	125.35	157.32	160	159
Max-Ones(320)	BS	237 (4)	302 (3)	319 (2)	**320 (1)**
	WS	219	291	314	318
	MBS	227.05	308.60	316.95	319
Max-Ones(640)	BS	422 (4)	553 (3)	606 (2)	**632 (1)**
	WS	408	513	586	627
	MBS	413.60	529.85	596.15	630
Royal-Road(40)	BS	4 (2)	**5 (1)**	4 (2)	**5 (1)**
	WS	2	5	1	5
	MBS	2.95	5	2.7	5
Royal-Road(80)	BS	5 (3)	**10 (1)**	7 (2)	**10 (1)**
	WS	3	8	2	9
	MBS	3.80	9.25	3.60	9.5
Royal-Road(160)	BS	5 (4)	15 (2)	10 (3)	**16 (1)**
	WS	4	10	3	15
	MBS	4.2	11.55	6.50	15
Royal-Road(320)	BS	6 (4)	15 (3)	18 (2)	**24 (1)**
	WS	4	9	8	21
	MBS	5.00	11.19	12.00	22.5

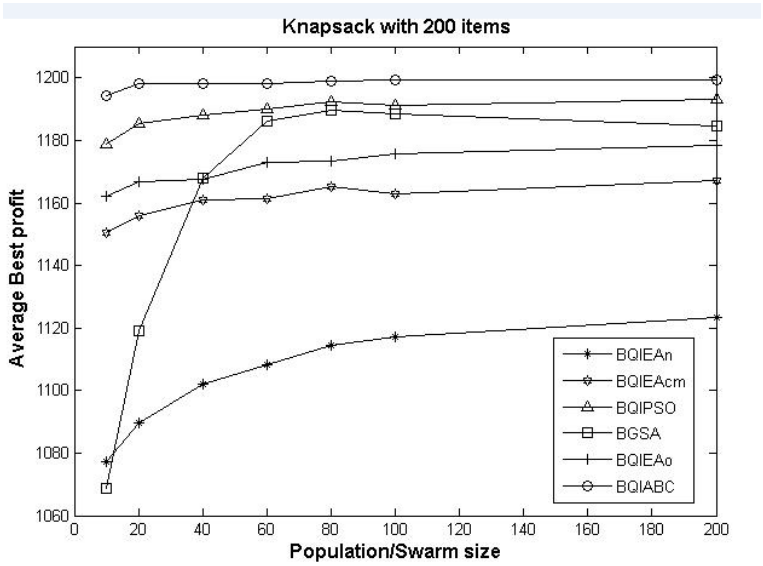

Figure 4. Effect of population size on average best profits obtained by BQIEAn, BQIEAcm, BQIEAo, BQIPSO, BGSA, and the proposed algorithm. The different swarm sizes are 10, 40, 60, 80, 100, and 200.

It can be observed in table 4 that BQIABC is able to provide better solutions in comparison with the other binary algorithms. The ability of the proposed algorithm to solve the binary encoded problems with larger sizes is mostly intuitive. In the table, rows captioned by "BS", "WS", and "MBS" report the best solution, worst solution, and mean best solutions over 30 runs, respectively. In Max-Ones and Royal-Road, the maximum number of iterations and the colony size were selected as 1000 and 20, respectively.

Figure 4 illustrates the relationship between the population size and the average best profits on the knapsack problem with 200 items. The results observed in this figure were achieved by the proposed algorithm and five binary comparative algorithms over 30 independent runs. These comparative algorithms include BQIEAn, BQIEAcm, BQIEAo, BQIPSO, and BGSA. The results for BQIPSO, BGSA and BQIEAn, BQIEAcm, BQIEAo have already been reported by H. Nezamabadi-pour [27] and G. Zhang [43], respectively. The information presented in figure 4 indicates that the increasing population size has the most influence on the optimal solutions obtained by BGSA and BQIEAn. Especially, BGSA has a significant increase from population size of 10 to 60. The BQIABC algorithm with population size of 20 could find the best solution with a profit of 1198.22. With increasing population size, the algorithms are able to find better solutions in the search space but to the contrary, the running time of algorithms will increase. In comparison with the other algorithms, BQIABC provides a better solution in all cases. In this study, the population size is considered to be 20 for the knapsack problem in all experiments.

5. Conclusions and future works

In the recent years, various versions of optimization algorithms have been widely used to solve binary problems. The artificial bee colony (ABC) is an evolutionary optimization algorithm motivated by the intelligence foraging behavior of real bees.

In this paper, we proposed a new quantum-inspired version of the ABC algorithm, called binary quantum-inspired artificial bee colony algorithm (BQIABC), to effectively solve combinatorial problems in binary space by applying some concepts of quantum computing such as a quantum bit and superposition of states in the standard ABC algorithm. BQIABC, due to its higher exploration ability, can provide a robust tool to solve binary optimization problems. BQIABC preserved the initial structure of the standard ABC algorithm. However, the concept of position of food sources and their updating process were replaced with new concepts. In this study, the 0/1 knapsack problem, Max-Ones, and Royal-Road functions were employed as binary optimization problems. To emphasize the effectiveness of BQIABC algorithm, a comparative study was done with ten other binary optimization algorithms. The comparison results illustrate that BQIABC can overcome other comparative algorithms. Also it seems that BQIABC has the ability to solve other binary optimization problems. We can consider this extension as one of the future works of this study.

References

[1] Hardy, G., Lucet, C. B. & Limnios, N. (2007). K-Terminal Network Reliability Measures with Binary

Decision Diagrams. IEEE Transactions on Reliability, vol. 56, no. 3, pp. 506-515.

[1] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.

[2] Karaboga, D. & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization. vol. 39, no. 3, pp. 459-471.

[3] Karaboga, D. & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing, vol. 8, no. 1, pp. 687-697.

[4] Gökdağ, H. & Yildiz, A. R. (2012). Structural damage detection using modal parameters and particle swarm optimization, Materials Testing, vol. 54, no. 6, pp. 416-420.

[5] Yildiz, A. R. (2013). Optimization of cutting parameters in multi-pass turning using artificial bee colony-based approach, Information Sciences, vol. 220, pp. 399-407.

[6] Yildiz, A. R. (2013). Hybrid Taguchi-differential evolution algorithm for optimization of multi-pass turning operations, Applied Soft Computing, vol. 13, no. 3, pp. 1433-1439.

[7] Yildiz, A. R. (2013). A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations, Applied Soft Computing, vol. 13, no. 3, pp. 1561–1566.

[8] Yildiz, A. R. (2013). Cuckoo search algorithm for the selection of optimal machining parameters in milling operations, International Journal of Advanced Manufacturing Technology, vol. 64, no. 1-4, pp. 55-61.

[9] Durgun, I. & Yildiz, A.R. (2012). Structural design optimization of vehicle components using Cuckoo search algorithm, Materials Testing, vol. 54, no. 3, pp. 185-188.

[10] Yildiz, B. S., Lekesiz, H. & Yildiz, A. R. (2016). Structural design of vehicle components using gravitational search and charged system search algorithms, Materials Testing, vol. 58, no. 1, pp. 79-81.

[11] Izakian, Z. & Mesgari, M. S. (2015). Fuzzy clustering of time series data: A particle swarm optimization approach. Journal of AI and Data Mining, vol. 3, no 1, pp. 39-46.

[12] Shi, H. (2006). Solution to 0-1 knapsack problem based on improved ant colony algorithm, In proceeding of IEEE International Conference on Information Acquisition, pp. 1062-1066, 2006.

[13] Liu, A., Wang, J., Han, G., Wang, S. & Wen, J. (2006). Improved simulated annealing algorithm solving for 0-1 knapsack problem, In proceedings of IEEE the 6th international conference in intelligent systems design and application, pp. 1-6, 2006.

[14] Li, ZK. & Li, N. (2009). A novel multi-mutation binary particle swarm optimization for 0-1 knapsack problem. In proceedings of Control and Decision conference, pp. 3042-3047, 2009.

[15] Sajedi, H. & Razavi, S. F. (2016). DGSA: discrete gravitational search algorithm for solving knapsack problem, Oper Res Int J, pp. 1-29.

[16] Sundar, S., Singh, A. & Rossi, A. (2010). An artificial bee colony algorithm for the 0-1 multi-dimensional knapsack problem, Commun. Comput. Inf. Sci., vol. 94, pp. 141–151.

[17] Kong, M., Tian, P. & Kao, Y. (2008). A new ant colony optimization algorithm for the multi-dimensional knapsack problem, Comput. Oper. Res., vol. 35, no. 8, pp. 2672–2683.

[18] Han, X. H., Quan, L., Xiong, X. Y. & Wu, B. (2013). Facing the classification of binary problems with a hybrid system based on quantum- inspired binary gravitational search algorithm and K-NN method, Eng. Appl. Artif. Intell., vol. 26, pp. 580-593.

[19] Han, X. H., Chang, X. M., Quan, L., Xiong, X. Y., Li, J. X., Zhang, Zh. X. & Liu Y. (2014). Feature subset selection by gravitational search algorithm optimization, Information Sciences, vol. 281, pp. 128-146.

[20] Liu, T., Zhang, L. & Zhang, J. (2013). Study of Binary Artificial Bee Colony Algorithm Based on Particle Swarm Optimization. Journal of Computational Information Systems, vol. 9, no. 16, pp. 6459-6466.

[21] Marinakis, Y, Marinaki, M, Matsatsinis, N. (2009). A hybrid discrete artificial bee colony-GRASP algorithm for clustering. International Conference on Computers Industrial Engineering. Troyes, France, pp. 548-553 2009.

[22] Pampara, G. & Engelbrecht, A. P. (2011). Binary artificial bee colony optimization. IEEE Symposium on Swarm Intelligence, IEEE, Perth, pp. 1-8, 2011.

[23] Kashan, M. H., Nahavandi, N. & Kashan, A. H. (2012). DisABC: A new artificial bee colony algorithm for binary optimization. App. Soft Com., vol. 12, no. 1, pp. 342-352.

[24] Hey, T. (1999). Quantum computing: An introduction. Com. and Cont. Eng. Journal, vol. 10, no. 3, pp. 105-112.

[25] Feynman, R. P. (1982). Simulating physics with computers. International Journal of Theoretical Physics, vol. 21, nos. 6/7, pp. 467-488.

[26] Deutsch, D. (1985). Quantum theory, the church-Turing principle and the universal quantum computer. In Proceedings of the Royal Society of London, pp. 97-117.

[27] Nezamabadi-pour, H. (2015). A quantum-inspired gravitational search algorithm for binary encoded

optimization problems. Engineering Applications of Artificial Intelligence. vol. 40, pp. 62-75.

[28] Spector, L., Barnum, H., Bernstein, H. J. & Swamy, N., (1999). Finding a better-than-classical quantum AND/OR algorithm using genetic programming. In Proceedings of Congr. Evolutionary Computation, Piscataway, NJ, vol. 3, pp. 2239–2246, 1999.

[29] Sahin, M., Atav, U. & Tomak, M. (2005). Quantum genetic algorithm method in self-consistent electronic structure calculations of a quantum dot with many electrons. International Journal of Modern Physics, vol. 16, no. 9, pp. 1379–1393.

[30] Sun, J., Feng, B. & Xu, W. (2004). Particle swarm optimization with particles having quantum behavior. In Proceedings of congress on evolutionary computation, Portland, Oregon, USA, pp. 325–331, 2004.

[31] Sun, J., Xu, W. & Feng, B. (2004). A global search strategy of quantum-behaved particle swarm optimization. In Proceedings of IEEE Conference on Cybernetics and Intelligent Systems, pp. 111–116, 2004.

[32] Soleimanpour-moghadam M., Nezamabadi-pour, H. & Farsangi, M.M. (2012). A quantum behaved gravitational search algorithm. Intel. Info. Manag., vol. 4, no. 6, pp. 390–395.

[33] Moore, M. & Narayanan A. (1995). Quantum-Inspired Computing. Dept. Comput. Sci., Univ. Exeter. Exeter, U.K.

[34] Narayanan, A. & Moore, M. (1996). Quantum-inspired genetic algorithms. In Proceedings of IEEE Int. Conf. Evolutionary Computation, Japan, pp. 61–66, 1996.

[35] Han, K. H. & Kim, J. H. (2002). Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Trans. Evol. Comput., vol. 6, no. 6, pp. 580–593.

[36] Duan, H. B., Xu, C. F. & Xing, Z. H. (2010). A hybrid artificial bee colony optimization and quantum evolutionary algorithm for continuous optimization problems. International Journal of Neural Systems, vol. 20, no. 1, pp. 39-50.

[37] Abbas, N. H. & Aftan, H. S. (2014). Quantum artificial bee colony algorithm for numerical function optimization. International Journal of Computer Applications, vol. 93, no. 9, pp. 0975-8887.

[38] Han, K. & Kim, J. (2002). Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Trans. Evol. Comput., vol. 6, no. 6, pp. 580–593.

[39] Li, G., Sun, M. & Li, P. (2015). Quantum-inspired bee colony algorithm. Journal of Optimization, vol. 4, pp. 51-60.

[40] Yuan, X., Wang, P., Yuan, Y., Huang, Y. & Zhang, X. (2015). A new quantum inspired chaotic artificial bee colony algorithm for optimal power flow problem. Energy Conversion and Management, vol. 100, pp. 1-9.

[41] Manochehri, K. & Alizadegan, A. (2015). Designing and Comparing Classic versus Quantum Artificial Bee Colony Algorithm. Journal of mathematics and computer science, vol. 14, pp. 183-192.

[42] Karaboga D (2010). Artificial Bee Colony Algorithm, www.scholarpedia.org/article/Artificial_bee_colony_algorithm, Scholarpedia. vol. 5, no. 3.

[43] Zhang, G. (2011). Quantum-inspired evolutionary algorithms: a survey and empirical study. Journal of Heuristics, vol. 7, no. 3, pp. 303-351.

[44] Jeong Y., Park J., Jang S. & Lee K. Y. (2010). A New Quantum-Inspired Binary PSO: Application to Unit Commitment Problems for Power Systems. IEEE Transactions on Power Systems, vol. 25, no. 3, pp. 1486 – 1495.

[45] Kennedy J. & Eberhart R. C. (1997). A discrete binary version of the particle swarm algorithm. In Proceedings of IEEE Int. Conf. Systems, Man, and Cybernetics, vol. 5, pp. 4104–4108, 1997.

[46] Lee, S., Soak, S., Oh, S., Pedrycz, W. & Joen, M., (2008). Modified binary particle swarm optimization. Natural Science, vol. 18, no. 9, pp. 1161-1166.

[47] Rashedi, E., Nezamabadi-pour, H. & Saryazdi, S. (2010). BGSA: Binary Gravitational Search Algorithm, Journal of Nat Compute, vol. 9, pp. 727–745.

[48] Deng, C., Zhao, B., Yang, Y. & Deng, A. (2010). Novel binary differential evolution without scale factor. In proceedings of the third international workshop on advanced computer intelligence. Auzhou, Jiangsu, China, pp. 250-253, 2010.

[49] Li, N., Du, P. & Zhao, H. J. (2005). Independent component analysis based on improved quantum genetic algorithm: Application in hyperspectral images. In Proceedings of IGARSS, pp. 4323–4326, 2005.

[50] Zhang, G. X., Li, N., Jin, W. D. & Hu, L. Z. (2006). Novel quantum genetic algorithm and its applications. Front. Electr. Electron. Eng. China, vol. 1, no. 1, pp. 31–36.

نشریه هوش مصنوعی و داده کاوی

ارائه یک الگوریتم کوانتومی کلونی زنبورهای مصنوعی باینری برای حل مسائل بهینه‌سازی باینری

فاطمه بارانی[1*] و حسین نظام‌آبادی‌پور[2]

[1] گروه مهندسی کامپیوتر، مجتمع آموزش عالی بم،، ایران.

[2] گروه مهندسی برق الکترونیک، دانشگاه شهید باهنر کرمان، ایران.

ارسال ۲۰۱۶/۱۲/۰۶؛ پذیرش ۲۰۱۷/۰۲/۲۱

چکیده:

الگوریتم کلونی زنبورهای مصنوعی (ABC) یکی از الگوریتم بهینه‌سازی هوش جمعی است که از رفتار هوشمندانه زنبورهای عسل در هنگام جستجوی منابع غذایی الهام گرفته شده است. نسخه‌های مختلف الگوریتم ABC به طور گسترده برای حل مسائل بهینه‌سازی پیوسته و گسسته در زمینه‌های مختلف استفاده شده‌اند. در این مقاله، نسخه باینری جدیدی از الگوریتم ABC، به نام الگوریتم کوانتومی کلونی زنبورهای مصنوعی باینری (BQIABC)، با الهام از محاسبات کوانتومی ارائه شده است. در BQIABC برای تعریف یک الگوریتم جدید با توانایی جستجوی بیشتر، ساختار اصلی الگوریتم ABC و برخی از مفاهیم موجود در محاسبات کوانتومی از جمله، بیت کوانتومی، وضعیت سوپرپوزیشن کوانتومی و استراتژی Q-گیت‌های چرخشی، ترکیب شده‌اند. الگوریتم پیشنهادی به دلیل توانایی جستجوی بالا قادر است ابزار قدرتمندی را برای حل مسائل بهینه‌سازی باینری فراهم نماید. برای ارزیابی کارایی الگوریتم پیشنهادی آزمایش‌های مختلفی بر روی مساله کوله‌پشتی ۰/۱، توابع معیار حداکثر Royal-Road و Max-Ones انجام شده است. نتایج تولید شده توسط BQIABC با ده الگوریتم بهینه‌سازی باینری دیگر مورد مقایسه قرار گرفت. مقایسه‌ها نشان دادند که BQIABC نتایج بهتر یا مشابه الگوریتم‌های دیگر ارائه می‌دهد.

کلمات کلیدی: الگوریتم کلونی زنبورهای مصنوعی، محاسبات کوانتومی، Q-گیت چرخشی، مساله کوله‌پشتی ۰/۱، توابع معیار حداکثر.