International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 4, Issue 4 (April 2015), PP.41-51

Quantum ant colony optimization algorithm based onBloch spherical search

Jianping Li, [2]Siyuan Zhao, [3]Aiping Lu
[1]*School of Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China,*
[2]*School of Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China,*
[3]*School of Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China,*

Abstract:-In the existing quantum-behaved optimization algorithms, almost all of the individuals are encoded by qubits described on plane unit circle. As qubits contain only a variable parameter, quantum properties have not been fully embodied, which limits the optimization ability rise further. In order to solve this problem, this paper proposes a quantum ant colony optimization algorithm based on Bloch sphere search. In the proposed algorithm, the positions of ants are encoded by qubits described on Bloch sphere. First, the destination to move is determined according to the select probability constructed by the pheromone and heuristic information, then, the rotation axis is established with Pauli matrixes, and the evolution search is realized with the rotation of qubits on Bloch sphere. In order to avoid premature convergence, the mutation is performed with Hadamard gates. Finally, the pheromone and the heuristic information are updated in the new positions of ants. As the optimization process is performed in n-dimensional hypercube space $[-1, 1]^n$, which has nothing to do with the specific issues, hence, the proposed method has good adaptability for a variety of optimization problems. The simulation results show that the proposed algorithm is superior to other quantum-behaved optimization algorithms in both search ability and optimization efficiency.

Keywords:- ant colony optimization, quantum ant colony optimization, Bloch sphere,algorithm design

I. INTRODUCTION

At present, colony intelligence optimization algorithms have been widely studiedby many scholars, and have obtained the successful applications [1]. Quantum-behavedoptimization algorithm is an emerging interdisciplinary based on combinationof quantum computing and information science. In 1996, Ajit et al proposedquantum-inspired genetic algorithms [2], where concepts and principles of quantum mechanics were used to inform and inspire more efficient evolutionary computing methods. After this, quantum intelligent optimization quickly became a hot international research. Beginning of this century, Han et al proposed several quantum genetic algorithms [3-5]. Compared with the traditional evolutionary algorithm, the advantage of Han's algorithms is a better ability to maintain population diversity. In 2004, Hichem et al presented a new algorithm for solving the traveling salesman problem [6], which extended the standard genetic algorithm by combining them to some concepts and principles provided from quantum computing field such as qubits, states superposition and interference. Many quantum behaved optimization algorithms proposed later may be regarded as improvements of the above-mentioned algorithms [7-11]. In 2009, Balicki proposed an adaptive quantum-based multi-objective evolutionary algorithm where the crossover probability is decreased due to the number of new generations [12]. It is an advanced technique for finding Pareto-optimal task allocation problem with the maximization of the system reliability and distributed system performance. In addition, for the combination of quantum computation and neural networks, Perus presented an analogous quantum information processing system called a quantum associative network [13]. It is expected that successful quantum implementation of the model would yield many benefits. In 2010, we proposed an improved design for CNOT gated quantum neural networks model and presented a smart algorithm for it [14]. The experimental results shown that our model has more superior performance to the standard error back-propagation networks.

It is clear that, first, in the majority of quantum-behaved optimization algorithms, all individuals are encoded by qubits described on the plane unit circle. Because this description has only an adjustable parameter, quantum properties have not been fully reflected. Secondly, almost all of evolution and mutation used quantum rotation gates and quantum NOT gates. These operations only change a parameter of qubit, therefore, quantum properties are weakened. In 2008, we proposed a quantum-inspired evolutionary algorithm for continuous space optimization based on Bloch coordinates of qubits [15]. This algorithm adopted the qubits' Bloch spherical coordinate coding, had two adjustable parameters, and showed some good optimization performances. However, in this algorithm, the best matching of two adjustments was not achieved, which affected the optimization ability to further improve. Based on the above problems, we select ant colony optimization as a starting point,

and then propose a quantum ant colony optimization algorithm based on Bloch spherical search (BQACO). Unlike traditional ant colony optimization, in our algorithm, ants release pheromone not on the paths but on the current resident points. The individuals are directly encoded by qubits. This algorithm uses Pauli matrices to establish the rotation axis, uses qubit's pivoting to achieve ant's movement, and uses the Hadamard gates to achieve mutation. This search approach can simultaneously adjust two parameters of qubits, and can automatically achieve the best matching of two adjustments. In search process, the positive feedback is formed by sharing in pheromone. Along with the algorithm running, some pheromone trails can be seen leading to the global optimum solution in the optimization space. With the typical function extremum optimization and the fuzzy controller parameters optimization, and comparison with other algorithms, the experimental results verify the effectiveness of the BQACO.

II. THE BASIC PRINCIPLES OF BQACO

2.1 The spherical description of qubits

In quantum computing, a qubit is a two-level quantum system, described by a twodimensional complex Hilbert space. From the superposition principles, any state of the qubit may be written as follows

$$|\varphi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\cos\frac{\theta}{2}|1\rangle \quad (1)$$

where $0 \le \theta \le \pi$, $0 \le \phi \le 2\pi$, i∗ is the imaginary unit.

Therefore, unlike the classical bit, which can only be set equal to 0 or 1, the qubit resides in a vector space parametrized by the continuous variables θ and ϕ. Thus, a continum of states is allowed. The Bloch sphere representation is useful in thinking about qubits since it provides a geometric picture of the qubit and of the transformations that one can operate on the state of a qubit. Owing to the normalization condition, the qubit's state can be represented by a point on a sphere of unit radius, called the Bloch sphere. This sphere can be embedded in a three-dimensional space of Cartesian coordinates($x = \cos\phi\sin\theta$, $y = \sin\phi\sin\theta$, $z = \cos\theta$). Thus the state $|\varphi\rangle$ can be written as follows

$$|\varphi\rangle = \left[\sqrt{\frac{1+z}{2}}, \frac{x+i*y}{\sqrt{2(1+z)}}\right]^T \quad (2)$$

By definition, a Bloch vector is a vector whose components (x, y, z) single out a point on the Bloch shpere. We can also say that the angles θ and ϕ define a Bloch vector, as shown in Fig.1.

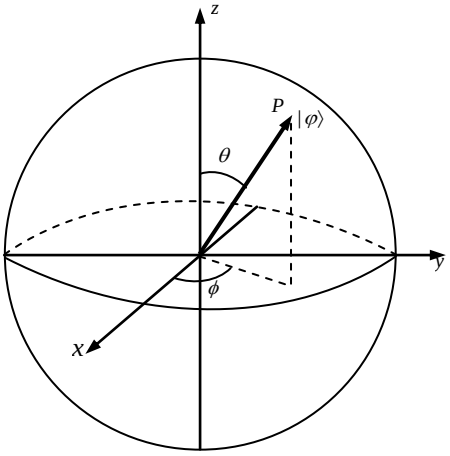

Fig.1 Bloch-sphere representation of a qubit.

2.2 The BQACO encoding method

In BQACO, all ants are encoded by qubits described on Bloch sphere. Set the colony size to m, and the space dimension to n. Then the ith ant is encoded as follows

$$\boldsymbol{p}_i = [|\varphi_{i1}\rangle, |\varphi_{i2}\rangle, \cdots, |\varphi_{in}\rangle] \quad (3)$$

where $|\varphi\rangle = [\cos\frac{\theta}{2}|0\rangle, e^{i\varphi}\cos\frac{\theta}{2}|1\rangle$ denotes the characteristic length of the *delta* potential well.

As the optimization process is performed in n-dimensional hypercube space$[-1, 1]^n$, which has nothing to do with the specific issues, hence, the proposed method has good adaptability for a variety of optimization problems.

2.3 Projective measurement of ant position

From the principles of quantum computing, the coordinates x, y, and z of a qubiton the Bloch sphere can be measured by using the Pauli operators written in thecomputational basis as follows

$$\boldsymbol{\sigma}_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \boldsymbol{\sigma}_y = \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix},\ \boldsymbol{\sigma}_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (4)$$

Let $|\boldsymbol{\varphi}_{ij}\rangle$ denote the j^{th} qubit on the i^{th} ant. The coordinates (x_{ij}, y_{ij}, z_{ij}) of $|\boldsymbol{\varphi}_{ij}\rangle$ can be obtained by the follows equations

$$x_{ij} = \langle \boldsymbol{\varphi}_{ij} | \boldsymbol{\sigma}_x | \boldsymbol{\varphi}_{ij} \rangle = \langle \boldsymbol{\varphi}_{ij} | \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} | \boldsymbol{\varphi}_{ij} \rangle \quad (5)$$

$$y_{ij} = \langle \boldsymbol{\varphi}_{ij} | \boldsymbol{\sigma}_y | \boldsymbol{\varphi}_{ij} \rangle = \langle \boldsymbol{\varphi}_{ij} | \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix} | \boldsymbol{\varphi}_{ij} \rangle \quad (6)$$

$$z_{ij} = \langle \boldsymbol{\varphi}_{ij} | \boldsymbol{\sigma}_z | \boldsymbol{\varphi}_{ij} \rangle = \langle \boldsymbol{\varphi}_{ij} | \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} | \boldsymbol{\varphi}_{ij} \rangle \quad (7)$$

2.4 Solution space transformation

In BQACO, each ant in colony contains 3n Bloch coordinates of n qubits that can be transformed from hypercube space [−1, 1]n to solution space of the continuous optimization problem. Each of Bloch coordinates corresponds to an optimization variable in solution space. Let the j^{th} variable of optimization problem $\mathrm{x}_j \in [A_j, B_j]$ and (x_{ij}, y_{ij}, z_{ij}) denote the coordinates of the j^{th} qubit on the i^{th} ant. Then the corresponding variables (x_{ij}, y_{ij}, z_{ij}) in solution space are respectively computed as follows

$$X_{ij} = \frac{1}{2}[A_j(1 - x_{ij}) + B_j(1 + x_{ij})] \quad (8)$$

$$Y_{ij} = \frac{1}{2}[A_j(1 - y_{ij}) + B_j(1 + y_{ij})] \quad (9)$$

$$Z_{ij} = \frac{1}{2}[A_j(1 - z_{ij}) + B_j(1 + z_{ij})] \quad (10)$$

where $i = 1, 2, \cdots, m$ ， $j = 1, 2, \cdots, n$。

2.5 Selecting target position of ant moving

Suppose $\tau(\boldsymbol{x}_r)$ denote the pheromone of point $\boldsymbol{x}_r$ held by the ant r^{th} which is a constant in the beginning of the optimization, and $\eta(\boldsymbol{x}_r)$ denote the heuristicinformation of point $\boldsymbol{x}_r$ which its meaning is similar to the pheromone of point $\boldsymbol{x}_r$ and it is also set to a constant in the beginning of the optimization. The rules ofthe r^{th} ant moving from $\boldsymbol{x}_r$ to $\boldsymbol{x}_s$ is described as follows

$$\boldsymbol{x}_s = \begin{cases} \arg\max\limits_{\boldsymbol{x}_s \in X} \{[\tau(\boldsymbol{x}_s)]^{\alpha}[\eta(\boldsymbol{x}_s)]^{\beta}\}, q \le q_0 \\ \bar{\boldsymbol{x}}_s \end{cases} \quad (11)$$

$$p(\boldsymbol{x}_s) = \frac{[\tau(\boldsymbol{x}_s)]^{\alpha}[\eta(\boldsymbol{x}_s)]^{\beta}}{\sum\limits_{\boldsymbol{x}_s, \boldsymbol{x}_u \in X} [\tau(\boldsymbol{x}_u)]^{\alpha}[\eta(\boldsymbol{x}_u)]^{\beta}} \quad (12)$$

where α and β are parameters controlling the relative importance between $\tau(x_i)$ and $\eta(x_i)$, q is a random number uniformly distributed in (0,1), q_0 is a pre-specified parameter ($0 < q_0 < 1$) used to switch two select ways, x is the set of points held by ant colony in solution space $\prod_{j=1}^{n}[A_j, B_j]$.

2.6 The ant movement to target position

In BQACO, we realize research on Bloch sphere. Namely, make qubit on the Bloch sphere rotate to the target around a fixed axis. This approach can simultaneously adjust two parameters θ and ϕ, and can automatically achieve the best matching of two adjustments, which can enhance optimization efficiency. Let $\boldsymbol{x}_r$ denote a ant's current position, and $\boldsymbol{x}_s$ denote its target position.

$$\boldsymbol{x}_r = [|\boldsymbol{\varphi}_{r1}\rangle, |\boldsymbol{\varphi}_{r2}\rangle, \cdots, |\boldsymbol{\varphi}_{rn}\rangle] \quad (13)$$

$$\boldsymbol{x}_s = [|\boldsymbol{\varphi}_{s1}\rangle, |\boldsymbol{\varphi}_{s2}\rangle, \cdots, |\boldsymbol{\varphi}_{sn}\rangle] \quad (14)$$

To make the ant move from $\boldsymbol{x}_r$ to $\boldsymbol{x}_s$, The determination of rotation axis and rotation angle is crucial, which can directly impact on convergence speed and efficiency of algorithm. For rotation axis, we propose the following method for determining.

Theorem 1 Let the vectors $\boldsymbol{P} = [p_x, p_y, p_z]$ and $\boldsymbol{Q} = [q_x, q_y, q_z]$ denote respectively qubits P and Q on Bloch sphere, then, the axis of rotating qubit from P and Q can be written as follows

$$\boldsymbol{R}_{axis} = \boldsymbol{P} \times \boldsymbol{Q} \quad (15)$$

Proof In the sphere, the shortest distance between two points is defined as the length of minor arc on the great circle through these two points. To make P approximate to Q after rotating, we should make P rotate along with the minor arc on the great circle. From $\boldsymbol{R}_{axis} = \boldsymbol{P} \times \boldsymbol{Q}$, we know that the direction of $\boldsymbol{R}_{axis}$ is perpendicular to the plane consisted of the vector P and Q, and the direction of these three vectors meet to the right-hand rule. Namely, right hand four fingers grip from point P to point Q with angle less than π, at this time, the direction of thumb is defined as the direction of $\boldsymbol{R}_{axis}$. the relation of these three vectors is shown in Fig.2. Therefore, if let P rotate around axis $\boldsymbol{R}_{axis}$, then its path will be the minor arc on great circle through points P and Q. Hence, the rotation axis is $\boldsymbol{R}_{axis} = \boldsymbol{P} \times \boldsymbol{Q}$.

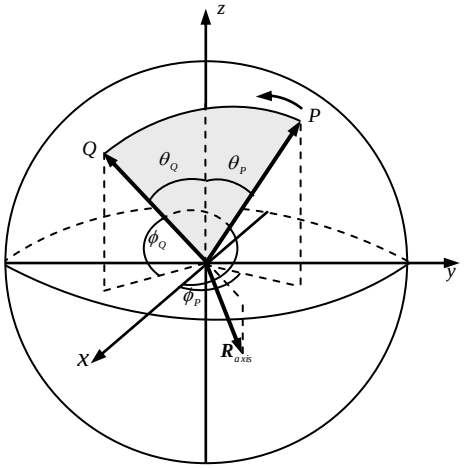

Fig.2 The rotation axis of qubit on Bloch sphere

Let O denote the centre of Bloch sphere, and points P and Q on Bloch sphere denote respectively qubits $|\boldsymbol{\varphi}_{rj}\rangle$ and $|\boldsymbol{\varphi}_{sj}\rangle$. According to the above theorem, the rotation axis of rotating $|\boldsymbol{\varphi}_{rj}\rangle$ to $|\boldsymbol{\varphi}_{sj}\rangle$ can be written as follows

$$\boldsymbol{R}_{axis} = \frac{\boldsymbol{OP} \times \boldsymbol{OQ}}{\| \boldsymbol{OP} \times \boldsymbol{OQ} \|} \quad (16)$$

From the principles of quantum computing, on Bloch sphere, the rotation matrix through an angle δ about unit vector $\boldsymbol{n} = [n_x, n_y, n_z]$ is defined as follows

$$\boldsymbol{R}_{\boldsymbol{n}}(\delta)=\cos\frac{\delta}{2}\boldsymbol{I}-\mathrm{i}\sin\frac{\delta}{2}(\boldsymbol{n}\times\boldsymbol{\sigma}) \quad (17)$$

where $\boldsymbol{\sigma}=[\boldsymbol{\sigma}_x,\boldsymbol{\sigma}_y,\boldsymbol{\sigma}_z]$

Hence, on Bloch sphere, the rotation matrix through an angle δ about the axis $\boldsymbol{R}_{axis}$ of rotating the current qubit $|\boldsymbol{\varphi}_{rj}\rangle$ to the target qubit $|\boldsymbol{\varphi}_{sj}\rangle$ can be written as follows

$$\boldsymbol{M}_{\boldsymbol{R}_{axis}}(\delta)=\cos\frac{\delta}{2}\boldsymbol{I}-\mathrm{i}\sin\frac{\delta}{2}(\boldsymbol{R}_{axis}\times\boldsymbol{\sigma}) \quad (18)$$

where the rotation angle $\delta\le 0.05\pi$.

The rotation operation of rotating the current qubit $|\boldsymbol{\varphi}_{rj}\rangle$ to the target qubit $|\boldsymbol{\varphi}_{sj}\rangle$ can be written as follows

$$|\boldsymbol{\varphi}_{rj}(k+1)\rangle=\boldsymbol{M}_{\boldsymbol{R}_{axis}}(\delta)|\boldsymbol{\varphi}_{rj}(k)\rangle \quad (19)$$

where $r=1,2,\cdots,m$, $j=1,2,\cdots,n$, k is the iterative step.

For each ant in colony, the move method describes as follows. For each qubit in the current position, determine the rotation axis according to the corresponding qubit in target position, perform rotation around axis so that achieve the movement of this ant. By the above method, all the ants' positions are in turn updated so that complete the entire colony update.

2.7 The mutation of ant position

In order to increase colony diversity and prevent premature convergence, a variety of evolutionary algorithms introduce mutation. Most of the current quantum evolutionary algorithm use the quantum NOT gate (namely, Pauli metrix $\boldsymbol{\sigma}_x$) to perform the mutation, in which two probability amplitudes are exchanged, and only one parameter is changed. In BQACO, we propose a new mutation based on Hadamard gate whose a possible description is defined as follows

$$\boldsymbol{H}=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \quad (20)$$

This gate is an important unitary operator in quantum computing, it can be written as the linear combinations of two Pauli matrices, and has the following property.

$$-\mathrm{i}\boldsymbol{H}=\cos\frac{\pi}{2}\boldsymbol{I}-\mathrm{i}\sin\frac{\pi}{2}\left(\frac{\boldsymbol{\sigma}_x}{\sqrt{2}}+0\boldsymbol{\sigma}_y+\frac{\boldsymbol{\sigma}_z}{\sqrt{2}}\right) \quad (21)$$

From this equation it is clear that the Hadamard gate is a rotation through an angle δ = π about the axis $\boldsymbol{n}=\left(\frac{1}{\sqrt{2}},0,\frac{1}{\sqrt{2}}\right)$.

For each ant in colony, first generate a random number, if the number is less than the mutation probability, then randomly selecte a qubit of this ant, and apply Hadamard gate to perform its mutation.

2.8 The update rules of pheromone and heuristic information

For the optimization problem, we always hope to find such a point not only with a high fitness but also with a faster rate of fitness change, which can stride in the search process instead of roaming in the flat places. Based on this concept, in BQACO, the idea of pheromone updating is to add the fitness value of the current ant position to pheromone, which makes the better position hold the greater pheromone. The idea of heuristic information updating is to add the fitness change to heuristic information, which makes the position with the faster fitness change hold the greater heuristic information. In fact, in our approach, the fitness change is regarded as heuristic information. After all ants finish a step search, we compute the fitness and its change for ants' new position, and update the pheromone and heuristic information according to the following equations. Let $\boldsymbol{x}_q$ denote the ant's previous position, $\boldsymbol{x}_r$ denote the ant's current position, and $\boldsymbol{x}_s$ denote the ant's target position. Local update rules as follows

$$\tau(\boldsymbol{x}_s)=(1-\rho)\tau(\boldsymbol{x}_s)+\rho\times fit^{\alpha} \quad (22)$$

$$\eta(\boldsymbol{x}_s)=(1-\rho)\eta(\boldsymbol{x}_s)+\rho\times|\Delta fit|^{\beta} \quad (23)$$

$$\Delta fit = fit(\boldsymbol{x}_s) - fit(\boldsymbol{x}_r) \quad (24)$$

where $0 < \alpha < 1$ and $0 < \beta < 1$ denote the update index of pheromone and heuristic information, respectively, $0 < \rho < 1$ denotes the update coefficient of pheromone and heuristic information, $1-\rho$ denotes the evaporation coefficient, and t denotes the current iteration steps.

The global update of pheromone and heuristic information is performed after all ants finish a step search according to the following equation.

$$\tau(\boldsymbol{x}_u) = \begin{cases} (1-\rho)\tau(\boldsymbol{x}_u) + \rho fit(\boldsymbol{x}_u), \boldsymbol{x}_u = \bar{\boldsymbol{x}} \\ (1-\rho)\tau(\boldsymbol{x}_u) \quad , \boldsymbol{x}_u \neq \bar{\boldsymbol{x}} \end{cases} \quad (25)$$

$$\eta(\boldsymbol{x}_u) = \begin{cases} (1-\rho)\eta(\boldsymbol{x}_u) + \rho fit(\boldsymbol{x}_u), \boldsymbol{x}_u = \bar{\boldsymbol{x}} \\ (1-\rho)\eta(\boldsymbol{x}_u) \quad , \boldsymbol{x}_u \neq \bar{\boldsymbol{x}} \end{cases} \quad (26)$$

where $\bar{\boldsymbol{x}}$ denotes the optimal solution in current colony and *t* denotes the current iteration steps.

The design of fitness function usually depends on the specific optimization problem. For maximum optimization of the object function F, the fitness function may be defined as $fit = \exp(F)$, and for minimum optimization of the object function F, the fitness function may be defined as $fit = \exp(-F)$.

2.9 Population assessment and the optimal solution update

Substituting three solutions X_{ij}, Y_{ij}, Z_{ij} described by the ith ant into the fitness function, respectively, we may compute its fitness. Let $gfit_{best}$ denote the best fitness so far, and gp_{best} denote the corresponding best ant, $fit(p_i) = \max(fit(X_i), fit(Y_i)\, fit(Z_i))$, $fit_{best} = \max_{1 \le i \le m}(fit(p_i))$, if $gfit_{best} < fit_{best}$ then $gfit_{best} = fit_{best}$, $gp_{best} = p_{best}$.

III. THE IMPLEMENTATION SCHEME OF BQACO

Step1 Ant colony initialization. Include: colony size m, space dimension n, rotation angle δ, mutation probability p_m, iterative steps G, pheromone update index α, heuristic information update index β, evaporation coefficient $1-\rho$. According to equation (3), generate the initial colony, and initialize each ant's pheromone and heuristic information to a constant. Set the current iterative step $t = 0$.

Step2 For each ant, select the target position according to Eqs.(11), compute the rotation axis according to Eq.(14), compute the rotation matrix according to Eq.(17), rotate qubit according to Eq.(18) to achieve ant move. Mutate ant by Hadamard gates according to mutation probability.

Step3 Derive the ants' positions by means of projective measurements Eqs.(5-7), perform solution space transformation according to Eqs.(8-10), compute the fitness of each ant and its change.

Step4 Perform the local update and the global update of pheromone and heuristic information according to Eqs.(21-25).

Step5 Perform the global maximum solution update, set $t = t+1$, if $t > G$ then save the optimization results and stop, else go back to **step2**.

IV. COMPARATIVE EXPERIMENT

To verify the effectiveness of the BQACO, we design the two experiments. In these experiments, we implemented and evaluated the proposed method in Matlab (Version 7.1.0.246) on a Windows PC with 2.19 GHz CPU and 1.00 GB RAM, and we also compared BQACO with double chains quantum genetic algorithm(DCQGA) in Ref.[11] and Bloch quantum-inspired evolutionary algorithm (BQEA) in Ref.[15].

4.1 Function extremum optimization

For function extremum optimization, there already exists a lot of benchmark test functions. In this section, we select the following four typical 2-dimension functions as simulation objects. These four functions are complex and there exist a number of local extreme points in their domain of definition, which the performance of different algorithms can be investigate by these functions.

(1) Shaffers F5 function

$$f(x_i)=\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6} \quad (27)$$

where $x_i \in (-65.536, 65.536)$

$$(a_{ij}^k)=\begin{pmatrix} -32 & -16 & 0 & 16 & 32 \\ -32+16k & -32+16k & -32+16k & -32+16k & -32+16k \end{pmatrix},$$

$(a_{ij})=(a_{ij}^0,\ a_{ij}^1,\ a_{ij}^2,\ a_{ij}^3,\ a_{ij}^4)$,),i= 1, 2, j = 1, 2, · · · , 25, k = 0, 1, 2, 3, 4.

This function has multiple local maximum points, and the global maximum point is (-32,-32), the global maximum is 1.002. When the optimization result is greater than 1.000, the algorithm is considered convergence.

(2) Shubert function

$$f(x,y)=\{\sum_{i=1}^{5} i\cos[(i+1)x+i]\}\times \{\sum_{i=1}^{5} i\cos[(i+1)y+i]\} \quad (28)$$

where $x, y \in [-10, 10]$. This function has 760 local minimum points, and the global minimum is -186.73090882259. This function can easily fall into local minimum -186.34. When the optimization result is less than -186.34, the algorithm is considered convergence.

(3) Branin function

$$f(x,y)=(x-\frac{5.1}{4\pi^2}y^2+\frac{5}{\pi}y-6)^2 +10(1-\frac{1}{8\pi})\cos y+10 \quad (29)$$

where $x \in [0,15]$, $y \in [-5,10]$.This function has a global minimum 0.3979, and a local minimum 0.4004 that is very close to the global minimum. When the optimization result is less than 0.4000, the algorithm is considered convergence.

(4) Camel function

$$f(x,y)=(4-4x^2)x^2-xy-(4-2.1y^2+\frac{y^4}{3})y^2 \quad (30)$$

where x, y ∈ [−10, 10]. This function has a global maximum 1.031628. When the optimization result is greater than 1.000, the algorithm is considered convergence.

For the above four functions, we use respectively BQACO, DCQGA, BQEA to perform optimization. In order to reflect the fairness of comparing results, three algorithms use the same colony size, mutation probability, rotation angle, and iterative steps. These parameters are set as: colony size m = 20, space dimension n = *2*, iterative steps G = *100*, pre-specified parameter $q_0 = 0.8$, evaporation coefficient *1* − ρ = 0.5, pheromone update index α = 0.5, heuristic information update Index β = 0.5, rotation angle δ = 0.01π, mutation probability $p_m = 0.001$.

In order to manifest the objectivity of the comparison results, for the above four functions, we use each algorithm to optimize 1000 times, and record convergence times, average iterative steps, average results, and variance of results. For the case of convergence, we also record the times that converges to *X* solutions, *Y* solution, and *Z* solutions, respectively. The experimental results are shown in Tables 1-4. For the Shaffers F5 function, after it is optimized by BQACO, the pheromone distribution in solution space is shown in Fig.3.

TABLE I. THE OPTIMIZATION RESULTS OF SHAFFER'S F5

Alg.	Con. times	Avg. steps	Avg. Results	Variance
BQACO	784	66.1220	0.8970	0.0502
BQEA	723	68.7330	0.9044	0.0473
DCQGA	328	89.2410	0.7534	0.1024

TABLE II. THE OPTIMIZATION RESULTS OF SHUBERT

Alg.	Con. times	Avg. steps	Avg. Results	Variance
BQACO	944	61.6470	-186.4801	0.1860
BQEA	543	76.5250	-186.0720	0.7106
DCQGA	334	87.1210	-184.8086	56.1972

TABLE III. THE OPTIMIZATION RESULTS OF BRANIN

Alg.	Con. times	Avg. steps	Avg. Results	Variance
BQACO	996	49.8420	0.3990	8,6e-007
BQEA	826	58.9750	0.3995	2.5e-006
DCQGA	489	80.0120	0.4018	7.8e-005

TABLE IV. THE OPTIMIZATION RESULTS OF CAMEL

Alg.	Con. times	Avg. steps	Avg. Results	Variance
BQACO	999	36.5520	1.0143	7.2e-004
BQEA	947	44.6050	1.0109	2.1e-003
DCQGA	658	75.8260	-8.5867	1.2e-004

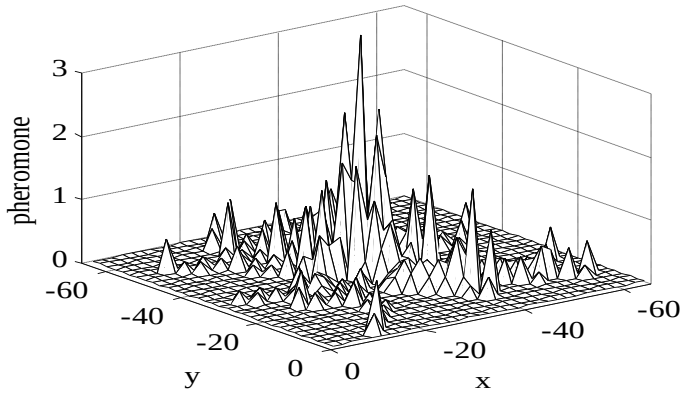

Fig.3 The pheromone distribution of Shaffers F5

It can be seen from tables 1-4 that, for the above four functions, three algorithms have the same sort in the optimization ability, from high to low in turn for BQACO, BQEA, DCQGA. These results can be analyzed as follows.

For DCQGA, individual's coding is based on the qubit described in unit circle. Because of only one adjustable parameter, quantum behavior can not be fully reflected. Hence, in the three algorithms, its optimization ability is the lowest.

For BQEA, because individual's coding is directly based on the Bloch coordinates of qubits, which transform the description of qubit from the unit circle to the Bloch sphere and fully reflect the quantum behavior, the search ability is effectively improved, which result in the optimization efficiency is obviously superior to that of DCQGA. In BQEA, because the two parameters θ and φ of a qubit are respectively adjusted, the best matching of two adjustment need to be considered. However, in BQEA, this best matching is ignored. In other words, when the current qubit moves toward the target qubit, the path is not the shortest.

For BQACO, ants' coding directly based on qubits described on the Bloch sphere, by means of projection measurement, the Bloch coordinates of qubits can be easily obtained. Therefore, BQACO have all advantages of BQEA. In addition, particularly noteworthy that, in BQACO, two parameters θ and φ of a qubit can be simultaneously adjusted by means of rotating the current qubit through an angle δ about the rotation axis. This rotation method can automatically achieve the best matching of two adjustments. In other words, when the current qubit moves toward the target qubit, the path is the minor arc on the great circle, which is clearly the shortest. Obviously, this rotation with the best matching of two adjustments has a higher optimization efficiency. So BQACO is more efficient than BQEA, and it also is the most efficient in three algorithms. The above analysis is consistent with the experimental results.

4.2 Fuzzy controller parameters optimization

In the fuzzy control system, the performance of the controller has a significant impact on the performance of system. Fuzzy controller performance to a large extent depends on the fuzzy control rules and its scalability. Therefore, we can introduce an adjustable parameter to adjust the control rules, so that the controlled object can obtain satisfactory control performance. This is the design problem of the fuzzy controller with a group of adjustable fuzzy rules. In this kind of fuzzy controller design, the control action depends on the error and error change. To adapt to different requirements of the controlled object, by introducing an adjustment factor, we may obtain a kind of fuzzy control rule with analytical description as follows

$$u = -\langle \alpha E + (1-\alpha) EC \rangle, \quad \alpha \in (0,1) \tag{31}$$

By adjusting the size of α, we can achieve varying degrees of weighting error and error change. When the error is large, the main control task is to eliminate error, at this time, we should increase the error's weighting. On the contrary, when the error is small, the main control task is to reduce the overshoot in order to stabilize the system as soon as possible. Therefore, in the different error levels, need to introduce different weighting factors in order to achieve self-adaptive adjusting of control rules. Taking the following second-order system for the controlled object, and using the step signal as input, we investigate the optimization ability of BQACO.

$$F(s) = \frac{20}{(2s+1)(4s+1)} \tag{32}$$

The domain of error, error change, and control size is selected as $\{E\} = \{EC\} = \{u\} = \{-3, -2, -1, 0, 1, 2, 3\}$. Taking into account the fuzzy control system in different system states should have different requirements for the control parameter α, in this experiment, the α is divided into three levels.

$$u = \begin{cases} -\langle \alpha_1 E + (1-\alpha_1) EC \rangle & E = 0, \pm 1 \\ -\langle \alpha_2 E + (1-\alpha_2) EC \rangle & E = \pm 2 \\ -\langle \alpha_3 E + (1-\alpha_3) EC \rangle & E = \pm 3 \end{cases} \tag{33}$$

Therefore, this study need to optimize the six fuzzy controller parameters such as quantization factor ke, kc, scale factor ku, adjustment factor $\alpha_1, \alpha_2, \alpha_3$. With help of the ITAE integral performance index, the evaluation function is designed as follows

$$f = \frac{1}{a + J(\mathrm{ITAE})} \tag{34}$$

where J(ITAE) = $\int_0^\infty t \,|\, e(t) \,|\, dt$, a denotes a small positive number so that the denominator is not zero.

According to experience, the initial scopes of six controller parameters are given by $\alpha_1 \in (0, 0.4)$, $\alpha_2 \in (0.4, 0.8)$, $\alpha_3 \in (0.8, 1.0)$, $ke, kc, ku \in (0, 10)$. These six parameters are respectively optimized by three algorithms. The colony size m = 15, space dimension n = 6, iterative steps G = 50, the other setting is the same as the previous experiment. The optimization results of three algorithms are shown in Table 5, the ITAE performance comparisons are shown in Fig.4, and the system response for step signal input under the control action of three fuzzy controllers optimized by three algorithms are shown in Fig.5.

TABLE V. OPTIMIZATION RESULTS CONTRAST OF FUZZY CONTROLLER PARAMETERS

Alg.	Ke	Kec	Ku	α_1	α_1	α_1	*J*(ITAE)
BQACO	4.5036	2.7263	9.9703	0.2468	0.4769	0.9194	3.9503
BQEA	4.2562	3.7794	4.1931	0.3712	0.5059	0.9967	5.4465
DCQGA	6.1607	5.6757	6.2983	0.0751	0.4728	0.8917	4.8329

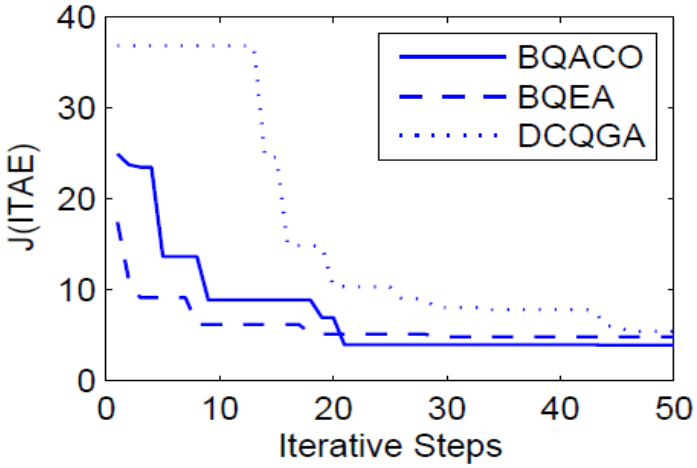

Fig.4 ITAE integral index comparison.

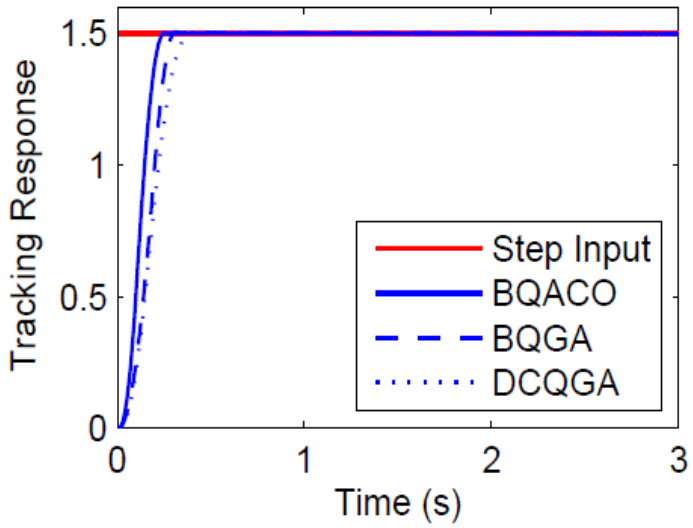

Fig.5 Tracking response curve comparisons of Fuzzy controllers.

Table 5 shows that, in the same colony size and iterative steps, the BQACO's J(ITAE) is the smallest, followed by BQEA and DCQGA. From Fig.4, it is clear that the BQACO has obtained the minimum 3.9503 after about 20 iterative steps, while, the BQEA and DCQGA have respectively obtained 4.8329 and 5.4465 after 50 iterative steps. Fig.5 shows that, the controller optimized by BQACO has a faster tracking speed and a less tracking time than that optimized by BQEA and DCQGA, which shows that BQACO obtains more excellent controller parameters combination than BQEA and DCQGA, and makes the fuzzy controller have the excellent control performance. The experimental results show that the optimization ability of BQACO indeed better than that of BQEA and DCQGA. These results can be explained as follows.

For DCQGA, qubit contains only one adjustable parameter while qubit contains two adjustable parameters for BQEA and BQACO. Hence, quantum behavior of DCQGA can not be fully reflected, and its optimization ability is inferior to that of BQEA and BQACO.

For BQEA, the Bloch coordinates of qubits are directly used for individual's coding which the quantum behavior is fully reflected, and the search ability is effectively improved. Hence, its optimization efficiency is obviously superior to that of DCQGA. However, the two parameters θ and φ of a qubit are respectively adjusted and the best matching of two adjustment is ignored, which limits the optimization capabilities to further improve.

For BQACO, its qubits are described on the Bloch sphere and are directly used for coding individuals. Using the Pauli matrices, it is easy to obtain the Bloch coordinates of qubits. Hence, BQACO have all advantages of BQEA. On the other hand, two parameters θ and φ of a qubit can be simultaneously adjusted by means of rotation matrix, and this rotation can automatically achieve the best matching of two adjustments, which make BQACO more efficient than BQEA.

V. CONCLUSIONS

This paper presents a quantum ant colony optimization algorithm based on Blochsphere search. The experimental results reveal that the coding method based onqubits described on Bloch sphere can better simulate the quantum behavior, the search method based on qubits' rotating around axises can improve search efficiency, and the integration of the following three aspects of ant colony optimization, coding method based on qubits described on Bloch sphere, and qubits rotating around axises on Bloch sphere, can really improve the optimization ability of the ant colony optimization algorithms. The advantages and disadvantages of the Bloch spherical search and the integration of this search method with other intelligent optimization algorithms are two issues which we need to study in future.

REFERENCES

[1]. Coskun O, Celal O, Filiz S: Quantum-inspired genetic algorithms, The Artificial Bee Colony algorithm in training Artificial Neural Network for oil spill detection, Neural Network World, 2011,21(6), pp: 473-492.

[2]. Ajit N, Mark M.: Quantum-inspired genetic algorithms, Proceedings of IEEE International Conference on Evolutionary Computation, Nagoya: IEEE Press, 1996, pp. 61-66.

[3]. Han K H, Kim J H.: Genetic quantum algorithm and its application to combinational optimization problem, Proceedings of IEEE International Conference on Evolutionary Computation,La Jolla: IEEE Press, 2000, pp. 1354-1360.

[4]. Han K H, Park K H, Lee C H.: Parallel quantum-inspired genetic algorithm for combinatorial optimization problem. Proceedings of IEEE Intternational Conference on Evolutionary Computation, Seoul: IEEE Press, 2001, pp. 1422-1429.

[5]. Han K H, Kim J H.: Quantum-inspired evolutionary algorithm for a class of combinational optimization, IEEE Transactions on Evolutionary Computing, 2002, 6(6), pp. 580-593.

[6]. Talbi H, Draa A, Batouche M.: A new quantum-inspired genetic algorithm for solving the travelling salesman problem. Proceedings of IEEE International Conference on Industrial Technology, Constantine: IEEE Press, 2004, pp. 1192-1197.

[7]. Wang L, Tang F, Wu H.: Hybrid genetic algorithm based on quantum computing for numerical optimization and parameter estimation, Applied Mathematics and Computation, 2005, 171(2), pp. 1141-1156.

[8]. Zhang G X, Jin W D, Hu L Z.: A novel parallel quantum genetic algorithm, Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies, Chengdu: IEEE Press, 2003, pp. 693-697.

[9]. Chen H, Zhang J H, Zhang C.: Chaos updating rotated gates quantum-inspired genetic algorithm, Proceedings of the International Conference on Communications, Cuircuits and Systems, Chengdu: IEEE Press, 2004, pp. 1108-1112.

[10]. Yang J A, Li B, Zhuang Z Q.: Multi-universe parallel quantum genetic algorithm and its application to blind-source separation, Proceedings of the International Conference on Neural Networks and Signal Processing, Nanjing: IEEE Press, 2003, pp. 393-398.

[11]. Li P C, Song K P, Shang F H.: Double chains quantum genetic algorithm with application to neuro-fuzzy controller design, Advances in Engineering Software, 2011, 42, pp. 875-886.

[12]. Balicki J.: An adaptive quantum-based multiobjective evolutionary algorithm for efficient task assignment in distributed systems, Proceedings of 13th WSEAS International Conference on Computers, Greece: WSEAS Press, 2009, pp. 417-422.

[13]. Perus M: Neural networks as a basis for quantum associative networks, Neural Network World, 2000,10(4), pp: 1001-1013.

[14]. Panchi L, Kaoping S, Erlong Y.: Model and algorithm of neural networks with quantum gated nodes, Neural Network World, 2010,20(2), pp: 189-206.

[15]. Li P C, Li S Y.: Quantum-inspired evolutionary algorithm for continuous spaces optimization based on Bloch coordinates of qubits, Neurocomputing, 2008, 72, pp. 581-591.

[16]. Giuliano B, Giulio C, Giuliano S.: Principles of quantum computation and information (Volume I: Basic concepts), Singapore: World Scientifi, 2004, pp:108-111.