Open Journal of Optimization, 2015, 4, 21-30
Published Online June 2015 in SciRes. http://www.scirp.org/journal/ojop
http://dx.doi.org/10.4236/ojop.2015.42003

Quantum-Inspired Particle Swarm Optimization Algorithm Encoded by Probability Amplitudes of Multi-Qubits

Xin Li[1], Huangfu Xu[2], Xuezhong Guan[2]

[1]School of Computer and Information Technology, Northeast Petroleum University, Daqing, China
[2]School of Electrical and Information Engineering, Northeast Petroleum University, Daqing, China
Email: lixin_dq@163.com

Received 9 February 2015; accepted 12 May 2015; published 14 May 2015

Abstract

To enhance the optimization ability of particle swarm algorithm, a novel quantum-inspired particle swarm optimization algorithm is proposed. In this method, the particles are encoded by the probability amplitudes of the basic states of the multi-qubits system. The rotation angles of multi-qubits are determined based on the local optimum particle and the global optimal particle, and the multi-qubits rotation gates are employed to update the particles. At each of iteration, updating any qubit can lead to updating all probability amplitudes of the corresponding particle. The experimental results of some benchmark functions optimization show that, although its single step iteration consumes long time, the optimization ability of the proposed method is significantly higher than other similar algorithms.

Keywords

Quantum Computing, Particle Swarm Optimization, Multi-Qubits Probability Amplitudes Encoding, Algorithm Design

1. Introduction

In 1999, Dr. Eberhart and Dr. Kennedy proposed particle Swarm Optimization (particle swarm optimization, PSO) [1]. As a new optimization tool, it is now widely used in combinatorial optimization [2] and numerical optimization [3]. In PSO's performance improvement, some commonly used strategies are as follows: selecting the appropriate control parameters [4]; designing reasonable update rules of the particle velocity and position [5]; combining PSO with the other algorithms [6]; and employing quantum computation to design the update strate-

How to cite this paper: Li, X., Xu, H.F. and Guan, X.Z. (2015) Quantum-Inspired Particle Swarm Optimization Algorithm Encoded by Probability Amplitudes of Multi-Qubits. *Open Journal of Optimization*, **4**, 21-30.
http://dx.doi.org/10.4236/ojop.2015.42003

gy [7]. These approaches enhance the PSO performance in different degrees. Quantum computing is an emerging interdisciplinary, combining the information science and quantum mechanics, and its integration with intelligent optimization algorithms begun in the 1990s; there is quantum-behaved particle swarm optimization algorithm [8], quantum-inspired evolutionary algorithm [9], quantum-inspired harmony search algorithm [10], quantum-inspired immune algorithm [11], quantum-inspired genetic algorithm [12], and quantum-inspired derivative differential evolution algorithm [13]. In the algorithm mentioned above, Ref. [8] applied real-based code method; the other references employed single qubit probability amplitude to code individuals. In these kinds of coding, the adjustment of a qubit can only change one gene on the individual. However, in the multi-qubits probability amplitude-based code, with application of coherence quantum states, simply adjusting a qubit can change all probability amplitudes of the ground state in multi-bit quantum superposition states, and then update all genes on the individual. In this paper, we propose a new multi-qubits probability amplitude encoding-based quantum-inspired particle swarm optimization. Standard function extreme optimization experiments show the superiority of the proposed algorithm.

2. Basic PSO Model

There is M particles in the n-dimensional space. For the i^{th} particle, its position $\boldsymbol{X}_i$, velocity $\boldsymbol{V}_i$, self-optimum position $\boldsymbol{P}_i^L$, global optimum position $\boldsymbol{P}_g$, are written as: $\boldsymbol{X}_i = (x_{i1}, x_{i2}, \cdots, x_{in})$; $\boldsymbol{V}_i = (v_{i1}, v_{i2}, \cdots, v_{in})$; $\boldsymbol{P}_i^L = (p_{i1}, p_{i2}, \cdots, p_{in})$; $\boldsymbol{P}_g = (p_{g1}, p_{g2}, \cdots, p_{gn})$. The update strategy of particles can be described as follows.

$$\boldsymbol{V}_i(t+1) = w\boldsymbol{V}_i(t) + c_1 r_1 \left(\boldsymbol{P}_i^L - \boldsymbol{X}_i(t)\right) + c_2 r_2 \left(\boldsymbol{P}_g - \boldsymbol{X}_i(t)\right) \tag{1}$$

$$\boldsymbol{X}_i(t+1) = \boldsymbol{X}_i(t) + \boldsymbol{V}_i(t) \tag{2}$$

where $i = 1, \cdots, M$, w is the inertia factor, c_1 is itself factor, c_2 global factor, r_1, r_2 is an uniformly distributed random number in (0, 1).

For convenience of description, Equation (1) can be rewritten as follows.

$$\boldsymbol{V}_i(t+1) = w\boldsymbol{V}_i(t) + [\Phi]\left(\boldsymbol{P}_i - \boldsymbol{X}_i(t)\right) \tag{3}$$

where

$$\boldsymbol{P}_i = \operatorname{diag}\left(\frac{c_1 r_1^1}{c_1 r_1^1 + c_2 r_1^2}, \cdots, \frac{c_1 r_n^1}{c_1 r_n^1 + c_2 r_n^2}\right)\boldsymbol{P}_i^L + \operatorname{diag}\left(\frac{c_1 r_1^2}{c_1 r_1^1 + c_2 r_1^2}, \cdots, \frac{c_1 r_n^2}{c_1 r_n^1 + c_2 r_n^2}\right)\boldsymbol{P}_g \tag{4}$$

$$[\Phi] = \operatorname{diag}\left(c_1 r_1^1 + c_2 r_1^2, \cdots, c_1 r_n^1 + c_2 r_n^2\right) \tag{5}$$

To make the PSO convergence, all particles must approximation $\boldsymbol{P}_i$.

3. Multi-Bit Quantum System and the Multi-Bit Quantum Rotation Gate

3.1. Qubits and Single Qubit Rotation Gate

What is a qubit? Just as a classical bit has a state—either 0 or 1—a qubit also has a state. Two possible states for a qubit are the state $|0\rangle$ and $|1\rangle$, which as you might guess correspond to the states 0 and 1 for a classical bit.

Notation like $|\ \rangle$ is called the Dirac notation, and we will see it often in the following paragraphs, as it is the standard notation for states in quantum mechanics. The difference between bits and qubits is that a qubit can be in a state other than $|0\rangle$ or $|1\rangle$. It is also possible to form linear combinations of states, often called superposition.

$$|\phi\rangle = \cos\theta|0\rangle + \sin\theta|1\rangle = [\cos\theta \quad \sin\theta]^{\mathrm{T}} \tag{6}$$

where θ is the phase of $|\phi\rangle$, $\cos\theta$ and $\sin\theta$ denote the probability amplitude of $|\phi\rangle$.

In the quantum computation, the logic function can be realized by applying a series of unitary transform to the qubit states, which the effect of the unitary transform is equal to that of the logic gate. Therefore, the quantum services with the logic transformations in a certain interval are called the quantum gates, which are the basis of

performing the quantum computation. A single qubit rotation gate can be defined as

$$\boldsymbol{R}(\Delta\theta)=\begin{bmatrix}\cos\Delta\theta & -\sin\Delta\theta\\ \sin\Delta\theta & \cos\Delta\theta\end{bmatrix} \tag{7}$$

Let the quantum state $|\phi\rangle=\begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}$, and $|\phi\rangle$ can be transformed by $\boldsymbol{R}(\Delta\theta)=\begin{bmatrix}\cos(\theta+\Delta\theta)\\ \sin(\theta+\Delta\theta)\end{bmatrix}$. It is obvious that $\boldsymbol{R}(\Delta\theta)$ shifts the phase of $|\phi\rangle$.

3.2. The Tensor Product of Matrix

Let the matrix $\boldsymbol{A}$ has *m* low and *n* column, and the matrix $\boldsymbol{B}$ has *p* low and *q* column. The tensor product of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as.

$$\boldsymbol{A}\otimes\boldsymbol{B}=\begin{bmatrix}A_{11}\boldsymbol{B} & A_{12}\boldsymbol{B} & \cdots & A_{1n}\boldsymbol{B}\\ A_{21}\boldsymbol{B} & A_{22}\boldsymbol{B} & \cdots & A_{2n}\boldsymbol{B}\\ \vdots & \vdots & \vdots & \vdots\\ A_{m1}\boldsymbol{B} & A_{m2}\boldsymbol{B} & \cdots & A_{mn}\boldsymbol{B}\end{bmatrix} \tag{8}$$

where $A_{i,j}$ is the element of matrix $\boldsymbol{A}$.

3.3. Multi-Bit Quantum System and the Multi-Bit Quantum Rotation Gate

In general, for an *n*-qubits system, there are 2^n of the form $|x_1x_2\cdots x_n\rangle$ ground states, similar to the single-qubit system, *n*-qubits system can also be in the a linear superposition state of 2^n ground states, namely

$$|\phi_1\phi_2\cdots\phi_n\rangle=\sum_{x_1=0}^{1}\sum_{x_2=0}^{1}\cdots\sum_{x_n}^{1}a_{x_1x_2\cdots x_n}|x_1x_2\cdots x_n\rangle=\left[a_{00\cdots0}\ \ a_{00\cdots1}\ \ \cdots\ \ a_{11\cdots1}\right]^{\mathrm{T}} \tag{9}$$

where $a_{x_1x_2\cdots x_n}$ is called probability amplitude of the ground state $|x_1x_2\cdots x_n\rangle$, and to meet the following equation.

$$\sum_{x_1=0}^{1}\sum_{x_2=0}^{1}\cdots\sum_{x_n}^{1}\left|a_{x_1x_2\cdots x_n}\right|^2=1 \tag{10}$$

Let $|\phi_i\rangle=\cos\theta_i|0\rangle+\sin\theta_i|1\rangle$, according to the principles of quantum computing, the $|\phi_1\phi_2\cdots\phi_n\rangle$ can be written as

$$|\phi_1\phi_2\cdots\phi_n\rangle=|\phi_1\rangle\otimes|\phi_2\rangle\otimes\cdots|\phi_n\rangle=\begin{bmatrix}\cos\theta_1\\ \sin\theta_1\end{bmatrix}\otimes\cdots\otimes\begin{bmatrix}\cos\theta_n\\ \sin\theta_n\end{bmatrix}=\begin{bmatrix}\cos\theta_1 & \cos\theta_2 & \cdots & \cos\theta_n\\ \cos\theta_1 & \cos\theta_2 & \cdots & \sin\theta_n\\ \vdots & \vdots & \ddots & \vdots\\ \sin\theta_1 & \sin\theta_2 & \cdots & \sin\theta_n\end{bmatrix} \tag{11}$$

It is clear from the above equations that, in an *n*-qubits system, any one of the ground state probability amplitude is a function *of n*-qubits phase $(\theta_1,\theta_2,\cdots,\theta_n)$, in other words, the adjustment of any θ_i can update all 2^n probability amplitudes.

In our works, the *n*-qubits rotation gate is employed to update the probability amplitudes. According to the principles of quantum computing, the tensor product of *n* single-qubit rotation gate $\boldsymbol{R}(\Delta\theta_i)$ is *n*-qubits rotation gate. Namely

$$\boldsymbol{R}(\Delta\theta_i\Delta\theta_2\cdots\Delta\theta_n)=\boldsymbol{R}(\Delta\theta_i)\otimes\boldsymbol{R}(\Delta\theta_2)\otimes\cdots\otimes\boldsymbol{R}(\Delta\theta_n) \tag{12}$$

where $\boldsymbol{R}(\Delta\theta_i)=\begin{bmatrix}\cos\Delta\theta_i & -\sin\Delta\theta_i\\ \sin\Delta\theta_i & \cos\Delta\theta_i\end{bmatrix}$, $i=1,2,\cdots,n$.

Taking $n=2$ as an example, the $\boldsymbol{R}(\Delta\theta_1\Delta\theta_2)$ can be rewritten as follows.

$$\boldsymbol{R}(\Delta\theta_1\Delta\theta_2)=\begin{bmatrix}\cos\Delta\theta_1\cos\Delta\theta_2 & -\cos\Delta\theta_1\sin\Delta\theta_2 & -\sin\Delta\theta_1\cos\Delta\theta_2 & \sin\Delta\theta_1\sin\Delta\theta_2\\ \cos\Delta\theta_1\sin\Delta\theta_2 & \cos\Delta\theta_1\cos\Delta\theta_2 & -\sin\Delta\theta_1\sin\Delta\theta_2 & \sin\Delta\theta_1\cos\Delta\theta_2\\ \sin\Delta\theta_1\cos\Delta\theta_2 & -\sin\Delta\theta_1\sin\Delta\theta_2 & \cos\Delta\theta_1\cos\Delta\theta_2 & -\cos\Delta\theta_1\sin\Delta\theta_2\\ \sin\Delta\theta_1\sin\Delta\theta_2 & \sin\Delta\theta_1\cos\Delta\theta_2 & \cos\Delta\theta_1\sin\Delta\theta_2 & \cos\Delta\theta_1\cos\Delta\theta_2\end{bmatrix} \tag{13}$$

It is clear that

$$\boldsymbol{R}_n(\Delta\theta_1\Delta\theta_2\cdots\Delta\theta_n)\left|\phi_1\phi_2\cdots\phi_n\right\rangle=\left|\hat{\phi}_1\right\rangle\otimes\left|\hat{\phi}_2\right\rangle\otimes\cdots\otimes\left|\hat{\phi}_n\right\rangle \tag{14}$$

where $\left|\hat{\phi}_i\right\rangle=\cos(\theta_i+\Delta\theta_i)\left|0\right\rangle+\sin(\theta_i+\Delta\theta_i)\left|1\right\rangle$.

4. Particle Encoding Method Based on Multi-Bits Probability Amplitudes

In this paper, the particles are encoded by multi-qubits probability amplitudes. Let *N* denote the number of particles, D denote the dimension of optimization space. Multi-qubits probability amplitudes encoding method can be described as follows.

4.1. The Number of Qubits Needed to Code

For an *n*-bits quantum system, there are 2^n probability amplitudes, which can be used directly as a result of an individual encoding. In the *D*-dimensional optimization space, it is clear that $D\le 2^n$. Due to the constraint relation between each probability amplitude (see to Equation (10)), hence $D<2^n$. For the *D*-dimensional optimization problem, the required number of qubits can be calculated as follows.

$$n=\log(D)+1 \tag{15}$$

4.2. The Encoding Method Based on Multi-Qubits Probability Amplitudes

First, generating randomly *N* *n*-dimensional phase vector $\boldsymbol{\theta}_i$, $i=1,2,\cdots,N$, as follows

$$\boldsymbol{\theta}_i=[\theta_{i1},\theta_{i2},\cdots,\theta_{in}] \tag{16}$$

where $\theta_{ij}=2\pi\times\text{rand}$, rand is a random number uniformly distributed within the (0,1), $j=1,2,\cdots,n$.

Let $\left|\phi_{ij}\right\rangle=\cos\theta_{ij}\left|0\right\rangle+\sin\theta_{ij}\left|1\right\rangle$, using Equation (11), we can obtain following N *n*-qubits systems $\left|\phi_{11}\phi_{12}\cdots\phi_{1n}\right\rangle$, $\left|\phi_{21}\phi_{22}\cdots\phi_{2n}\right\rangle$, $\cdots$, $\left|\phi_{N1}\phi_{N2}\cdots\phi_{Nn}\right\rangle$. In each of the quantum system, the first *D* probability amplitudes can be regarded as *a* *D*-dimensional particle code.

5. The Update Method Based on Multi-Qubits Probability Amplitudes

In this paper, the multi-bit quantum rotation gates are employed to update particles. Let the phase vector of the global optimal particle be $\boldsymbol{\theta}_g=[\theta_{g1},\theta_{g2},\cdots,\theta_{gn}]$, the phase vectors of the i^{th} particle be $\boldsymbol{\theta}_i=[\theta_{i1},\theta_{i2},\cdots,\theta_{in}]$, and the itself optimum the phase vector be $\boldsymbol{\theta}_{bi}=[\theta^i_{b1},\theta^i_{b2},\cdots,\theta^i_{bn}]$.

From Equation (11), it is clear that, once θ_i has been updated, all its corresponding probability amplitudes will be updated. To improve the search capability, in an iteration, all phases θ_i are updated in turn, which allows all particles are updated *n* times. Let the $\Delta\theta_0$ denote the phase update step size, the specific update can be described as follows.

Step 1. Set $j=1$, $\boldsymbol{P}_i(\theta)=[\cos\theta_{i1}\ \ \cos\theta_{i2}\ \ \cdots\ \ \cos\theta_{in}\ \ \cdots\ \ \sin\theta_{i1}\ \ \sin\theta_{i2}\ \ \cdots\ \ \sin\theta_{in}]^{\mathrm{T}}$.

Step 2. Set $\Delta\theta_{i1}=\Delta\theta_{i2}=\cdots=\Delta\theta_{in}=0$.

Step 3. Determine the value of the rotation angle, where the sgn donates the symbolic function.

If $\left|\theta^i_{bj}-\theta_{ij}\right|\le\pi$, then $\Delta\theta^b_{ij}=\text{sgn}(\theta^b_{ij}-\theta_{ij})\Delta\theta_0$.

If $\left|\theta^i_{bj}-\theta_{ij}\right|\le\pi$, then $\Delta\theta^b_{ij}=-\text{sgn}(\theta^i_{bj}-\theta_{ij})\Delta\theta_0$.

If $\left|\theta_{gj}-\theta_{ij}\right|>\pi$, then $\Delta\theta^b_{ij}=\text{sgn}(\theta_{gj}-\theta_{ij})\Delta\theta_0$.

If $\left|\theta_{gj}-\theta_{ij}\right|>\pi$, then $\Delta\theta_{ij}^{g}=-\operatorname{sgn}\left(\theta_{gj}-\theta_{ij}\right)\Delta\theta_{0}$.

Step 4. Compute the rotation angles, and update all particles according to the following equation, $\Delta\theta_{ij}=r1\times\Delta\theta_{ij}^{b}+r2\times\Delta\theta_{ij}^{g}$, $P_i\left(\theta\right)=R_n\left(\Delta\theta_{i1},\Delta\theta_{i2},\cdots,\Delta\theta_{in}\right)P_i\left(\theta\right)$. where $r1$ and $r2$ denote random numbers between the interval (0, 1).

Step 5. If $j<n$, then $j=j+1$, back to step 2.

6. Quantum-Inspired Particle Swarm Optimization Algorithm Encoded by Probability Amplitudes of Multi-Qubits

Suppose that, N denote the number of particles, D denote the number of optimization space dimension. For multi-qubits probability amplitudes encoding quantum-inspired particle swarm optimization, called MQPAP-SO, the optimization process can be described as follows.

1) Initialize the particles swarm

According to Equation (15) to determine the number of qubits n, according to Equation (16) initialize phase of each particle, according to Equation (11) to calculate the probability amplitude of 2^n each particle, where the first D probability amplitudes are the coding of the particles. Set the j^{th} probability amplitude of the i^{th} particle be x_{ij}, coding result can be expressed as the following equation.

$$\begin{cases}P_1=\left[x_{11},x_{12},\cdots,x_{1D}\right]^{\mathrm{T}}\\P_1=\left[x_{21},x_{22},\cdots,x_{2D}\right]^{\mathrm{T}}\\\vdots\\P_n=\left[x_{N1},x_{N2},\cdots,x_{ND}\right]^{\mathrm{T}}\end{cases}\tag{17}$$

Initialization phase update step $\Delta\theta_0$, the limited number of iteration G. Set the current iteration step $t=1$.

2) Calculation of the objective function value

Set the j-dimensional variable range be $\left[\operatorname{Min}X_j,\operatorname{Max}X_j\right]$, because of the probability amplitude x_{ij} values in the interval [0, 1], it is need to make the solution space transformation. The transformation equation is below.

$$X_{ij}=\frac{1}{2}\left[\operatorname{Max}X_j\left(1+x_{ij}\right)+\operatorname{Min}X_j\left(1-x_{ij}\right)\right]\tag{18}$$

where $i=1,2,\cdots,N$, $j=1,2,\cdots,D$.

Calculate the objective function values of all particles. Let the i^{th} particle phase be $\boldsymbol{\theta}_i=\left[\theta_{i1},\theta_{i2},\cdots,\theta_{in}\right]$, the objective function value is f_i, global optimal particle phase be $\hat{\boldsymbol{\theta}}_g=\left[\hat{\theta}_{g1},\hat{\theta}_{g2},\cdots,\hat{\theta}_{gn}\right]$, global optimal objective function value be $\hat{f}_g$, the i^{th} particle itself optimal phase is $\hat{\boldsymbol{\theta}}_i=\boldsymbol{\theta}_i$, Its optimal objective function value is $\hat{f}_i=f_i$.

3) Update the particle position

For each particle P_i, accordance to step 1 - step 5 in Section 5, update repeatedly n times. Using the Equation (11) to calculate the probability amplitude, using Equation (18) to implement the solution space transformation and calculate the value of the objective function. Let the objective function value of the i^{th} particle be f_i. If $f_i<\hat{f}_i$, then $\hat{f}_i=f_i$, $\theta_g=\hat{\theta}_g$.

4) Update the global optimal solution

Let the optimal particle phase be $\boldsymbol{\theta}_g=\left[\theta_{g1},\theta_{g2},\cdots,\theta_{gn}\right]$, the corresponding objective function value be f_g. If $f_g<\hat{f}_g$, then $\hat{f}_g=f_g$, $\hat{\theta}_g=\theta_g$, otherwise $f_g=\hat{f}_g$, $\theta_g=\hat{\theta}_g$.

5) Examine termination conditions

If $t<G$, $t=t+1$ back to (3), otherwise, save $\hat{\theta}_g$ and $\hat{f}_g$, end.

7. Comparative Experiment

In this study, the 20 standard test functions are employed to verify the optimization ability of MQPAPSO, and compare with the general particle swarm optimization (PSO) [14], quantum delta potential-well particle swarm optimization, QDPSO [15], shuffled frog leaping algorithm, SFLA [16]. All functions belong to minimum optimization, where D is the number of independent variables, Ω is the solution space, $\boldsymbol{X}^*$ is the exact minimum

point, $f(\boldsymbol{X}^*)$ is the corresponding minimum.

7.1. Test Function

(1) $f_1(\boldsymbol{X})=\sum_{i=1}^{D}x_i^2$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(2) $f_2(\boldsymbol{X})=\sum_{i=1}^{D}|x_i|+\prod_{i=1}^{D}|x_i|$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(3) $f_3(\boldsymbol{X})=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}x_j\right)^2$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(4) $f_4(\boldsymbol{X})=\max_{1\le i\le D}(|x_i|)$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(5) $f_5(\boldsymbol{X})=\sum_{i=1}^{D-1}\left(100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right)$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(6) $f_6(\boldsymbol{X})=\sum_{i=1}^{D}ix_i^4\left(1+\text{random}(0,1)\right)$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(7) $f_7(\boldsymbol{X})=\left[x_i^2-10\cos(2\pi x_i)+10\right]$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(8) $f_8(\boldsymbol{X})=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)+20+\mathrm{e}$; $\boldsymbol{\Omega}=[-100,100]^D$;
$\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(9) $f_9(\boldsymbol{X})=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(10) $f_{10}(\boldsymbol{X})=\frac{1}{D}\sum_{i=1}^{D}\left(x_i^4-16x_i^2+5x_i\right)+78.3323314$; $\boldsymbol{\Omega}=[-100,100]^D$; $x_i^*=-2.903534$; $f(\boldsymbol{X}^*)=0$.

(11) $f_{11}(\boldsymbol{X})=\frac{D(D+4)(D-1)}{6}+\sum_{i=1}^{D}(x_i-1)-\sum_{i=2}^{D}x_ix_{i-1}$; $\boldsymbol{\Omega}=\left[-D^2,D^2\right]^D$; $x_i^*=-2.903534$; $f(\boldsymbol{X}^*)=0$.

(12) $f_{12}(\boldsymbol{X})=\frac{\pi}{D}\left[10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2\left(1+10\sin^2(\pi y_{i+1})\right)+(y_D-1)^2\right]+\sum_{i=1}^{D}u(x_i,10,100.4)$;

$$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a;\\ 0, & -a\le x_i\le a;\\ k(-x_i-a)^m, & x_i<-a.\end{cases}\quad y_i=1+\frac{1}{4}(x_i+1);\ \boldsymbol{\Omega}=[-100,100]^D;\ \boldsymbol{X}^*=[-1,-1,\cdots,-1],$$

$f(\boldsymbol{X}^*)=0$.

(13) $f_{13}(\boldsymbol{X})=\sum_{i=1}^{D-1}\left(x_i^2+2x_{i+1}^2-0.3\cos(3\pi x_i)\cos(4\pi x_{i+1})+0.3\right)$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$;
$f(\boldsymbol{X}^*)=0$.

(14) $f_{14}(\boldsymbol{X})=\sum_{i=1}^{D/4}\left[(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}+10x_{4i})^4\right]$;
$\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(15) $f_{15}(\boldsymbol{X})=\sum_{i=1}^{D-1}g(x_i,x_{i+1})+g(x_D,x_1)\,g(x,y)=\left(x^2+y^2\right)^{0.25}\times\left[\sin^2\left(50\left(x^2+y^2\right)^{0.1}\right)+1\right]$;
$\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(16) $f_{16}(\boldsymbol{X})=10D+\sum_{i=1}^{D}\left(y_i^2-10\cos(2\pi y_i)\right)$; $y_i=\begin{cases}x_i, & |x_i|<1/2;\\ \text{round}(2x_i)/2 & |x_i|\ge 1/2.\end{cases}$; $\boldsymbol{\Omega}=[-100,100]^D$;
$\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0$.

(17) $f_{17}(\boldsymbol{X})=\sum_{i=1}^{D}\left\{\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]\right\}-D\sum_{k=0}^{k_{\max}}\left(a^k\cos\left(\pi b^k\right)\right)$; $a=0.5$; $b=0.3$; $k_{\max}=30$;

$\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0.$

(18) $f_{18}(\boldsymbol{X})=\sum_{i=1}^{D}x_i^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^4$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0.$

(19) $f_{19}(X)=\sum_{i=1}^{D-1}\left(0.5+\frac{\sin^2\left(\sqrt{100x_i^2+x_{i+1}^2}\right)-0.5}{1+0.001\left(x_i^2-2x_ix_{i+1}+x_{i+1}^2\right)^2}\right)$; $\boldsymbol{\Omega}=[-100,100]^D$; $\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0.$

(20) $f_{20}(\boldsymbol{X})=-\sum_{i=1}^{D-1}\left[\exp\left(\frac{-\left(x_i^2+x_{i+1}^2+0.5x_ix_{i+1}\right)}{8}\right)\times\cos\left(4\sqrt{x_i^2+x_{i+1}^2+0.5x_ix_{i+1}}\right)\right]+D-1$; $\boldsymbol{\Omega}=[-100,100]^D$;

$\boldsymbol{X}^*=[0,0,\cdots,0]$; $f(\boldsymbol{X}^*)=0.$

7.2. The Experimental Scheme and Parameter Design

The dimension of all test functions is set to $D=50$ (f_{14} for $D=52$) and $D=100$. Population size of these four algorithms is set to $N=50$. For PSO, QPSO and SFLA, the limited iteration number is set to $G=100$ and $G=1000$, respectively, and for MQPAPSO, set to $G=100$.

For SFLA, according to Ref. [16], the biggest jump step is set to $D_{\max}=5$. Because of the sub-group number of SFLA is related to the specific problem, we consider some different a variety of groupings, and the best results are used to compare with other algorithm. Specifically, we take the following six cases:

$$N=1\times50=2\times25=5\times10=10\times5=25\times2=50\times1,$$

where the first number denotes the number of sub-group and the second number denotes the number of frog in sub-group. For each of combination, the SFLA is independent run 30 times, and the average optimization result over 30 runs and the average time of a single iteration are recorded. In these six groups, the best optimization results and the corresponding average time of a single iteration are regarded as a comparison index.

For PSO, according to Ref. [14], $w=0.7298$, $c_1=c_2=1.49618$. For QDPSO, according to Ref. [15], the control parameters is set to $\lambda=1.2$. For MQPAPSO, phase update step take $\Delta\theta_0=0.05\pi$. Each function is optimized independently 30 times by these three algorithms, and the average optimization results and the average time of a single iteration are taken as a comparison index.

7.3. Comparative Experiment Results

Experiments conducted using Matlab R2009a. Taking $G=100$ as an example, the average time of a single iteration, the results of such comparison are shown in **Table 1**, the average optimization results for $D=50$ and $D=100$, are shown in **Table 2** and **Table 3**.

For the function f_i, let the average time of four algorithms for a single iteration be T_i^M, T_i^Q, T_i^P, T_i^S, respectively, and the average optimization results be Q_i^M, Q_i^Q, Q_i^P, Q_i^S, respectively. To facilitate comparison, taking MQPAPSO and QDPSO as an example, the ratio of the average time of a single iteration and the ratio of the average optimal results are defined as follows.

$$\frac{T^M}{T^Q}=\frac{\sum_{i=1}^{20}\frac{T_i^M}{T_i^Q}}{20},\quad \frac{Q^M}{Q^Q}=\frac{\sum_{i=1}^{20}\frac{Q_i^M}{Q_i^Q}}{20} \tag{19}$$

For four algorithms, the ratios of the average time of a single iteration are shown in **Table 4**, and the ratios of the average optimization results are shown in **Table 5**.

From **Table 1** and **Table 4**, for single iteration mean time, MQPAPSO is nearly 10 times longer than QDPSO, PSO, and SFLA. To make the comparison fair, we must further investigate the optimization results under the same running time. This is the fundamental reason why the iteration steps of for QDPSO, PSO, SFLA are set to $G=100$ and $G=1000$. From **Table 2** and **Table 3**, the average results of MQPAPSO are far less than the other three algorithms in both $D=100$ and $D=1000$, where shows that the use of multi-bit probability

Table 1. The average time contrast of single iteration for the four algorithms (unit: seconds).

f_i	MQPAPSO		QDPSO		PSO		SFLA	
	$D = 50$	$D = 100$	$D = 50$	$D = 100$	$D = 50$	$D = 100$	$D = 50$	$D = 100$
f_1	0.0186	0.0290	0.0011	0.0019	0.0009	0.0016	0.0014	0.0020
f_2	0.0187	0.0292	0.0012	0.0020	0.0012	0.0016	0.0017	0.0025
f_3	0.0248	0.0428	0.0064	0.0127	0.0099	0.0227	0.0068	0.0168
f_4	0.0188	0.0296	0.0011	0.0019	0.0009	0.0016	0.0014	0.0024
f_5	0.0230	0.0397	0.0049	0.0095	0.0016	0.0025	0.0022	0.0043
f_6	0.0235	0.0387	0.0018	0.0031	0.0028	0.0048	0.0019	0.0032
f_7	0.0187	0.0291	0.0013	0.0021	0.0016	0.0022	0.0014	0.0024
f_8	0.0191	0.0295	0.0015	0.0023	0.0019	0.0028	0.0024	0.0036
f_9	0.0234	0.0382	0.0016	0.0024	0.0019	0.0028	0.0017	0.0027
f_{10}	0.0193	0.0301	0.0019	0.0031	0.0028	0.0048	0.0017	0.0028
f_{11}	0.0234	0.0383	0.0016	0.0024	0.0016	0.0025	0.0017	0.0027
f_{12}	0.0262	0.0441	0.0096	0.0173	0.0054	0.0089	0.0033	0.0070
f_{13}	0.0193	0.0298	0.0020	0.0030	0.0025	0.0041	0.0017	0.0030
f_{14}	0.0233	0.0372	0.0048	0.0089	0.0028	0.0044	0.0024	0.0033
f_{15}	0.0256	0.0449	0.0031	0.0057	0.0051	0.0096	0.0038	0.0060
f_{16}	0.0248	0.0418	0.0065	0.0124	0.0028	0.0048	0.0027	0.0043
f_{17}	0.0378	0.0706	0.0246	0.0486	0.1116	0.2192	0.0292	0.0651
f_{18}	0.0234	0.0382	0.0017	0.0024	0.0019	0.0025	0.0022	0.0028
f_{19}	0.0206	0.0314	0.0028	0.0037	0.0038	0.0054	0.0033	0.0041
f_{20}	0.0212	0.0320	0.0028	0.0039	0.0041	0.0057	0.0043	0.0052

Table 2. The average optimization results contrast for four algorithms ($D = 50$).

f_i	MQPAPSO	QDPSO		PSO		SFLA	
	$G = 100$	$G = 100$	$G = 1000$	$G = 100$	$G = 1000$	$G = 100$	$G = 1000$
f_1	1.9E−08	1.5E+03	3.4E−05	3.4E+03	6.0E−05	8.5E+02	0.00108
f_2	1.3E−04	9.4E+10	33.1953	3.8E+15	1.3E+02	2.8E+02	2.6E+02
f_3	3.7E−09	3.9E+04	1.1E+04	6.7E+04	1.6E+04	6.3E+03	2.5E+03
f_4	0.00101	36.9364	10.2029	61.9675	54.6625	12.1258	9.71406
f_5	73.2154	1.2E+08	1.3E+02	2.7E+08	2.0E+02	1.1E+07	4.8E+02
f_6	4.1E−11	7.2E+07	2.1E+02	3.7E+08	1.5E+05	1.2E+04	2.3E−09
f_7	7.9E−06	1.9E+03	2.9E+02	3.3E+03	3.7E+02	1.4E+03	1.0E+03
f_8	3.3E−05	21.1629	20.5964	21.2778	21.1744	17.0524	15.7169
f_9	4.9E−10	11.1857	0.00352	23.6757	0.03275	2.00564	0.01209
f_{10}	18.5824	2.7E+04	10.5743	6.5E+04	23.4260	1.8E+03	12.7798
f_{11}	2.8E+04	1.6E+06	8.4E+04	5.1E+06	4.2E+05	9.7E+05	2.4E+04
f_{12}	0.18150	2.9E+07	0.28276	6.5E+07	1.65872	1.1E+04	17.3770
f_{13}	7.6E−07	4.2E+03	0.00231	1.0E+04	4.00830	3.2E+03	13.9008
f_{14}	3.9E−11	2.9E+06	7.3E+04	3.9E+07	4.5E+07	8.1E+05	3.4E+04
f_{15}	0.25413	2.3E+02	26.5175	3.0E+02	1.7E+02	1.7E+02	1.5E+02
f_{16}	2.2E−06	1.9E+03	3.5E+02	3.5E+03	3.9E+02	1.3E+03	1.0E+03
f_{17}	0.51201	67.3310	47.5805	78.8131	75.8892	48.0274	33.5393
f_{18}	1.1E−05	8.0E+04	4.1E+04	1.2E+05	9.8E+04	8.0E+03	5.4E+03
f_{19}	1.5E−04	0.49997	0.49959	0.49999	0.49998	0.49469	0.49168
f_{20}	1.1E−06	46.2759	34.4948	47.2014	45.7578	44.0433	43.4175

Table 3. The average optimization results contrast for four algorithms (D = 100).

f_i	MQPAPSO	QDPSO		PSO		SFLA	
	G = 100	G = 100	G = 1000	G = 100	G = 1000	G = 100	G = 1000
f_1	5.7E−08	2.3E+04	1.2E+02	3.9E+04	2.7E+02	3.3E+03	5.48043
f_2	6.3E−04	1.0E+20	6.0E+02	5.4E+25	1.2E+15	5.9E+02	5.7E+02
f_3	2.2E−08	1.9E+05	1.1E+05	3.0E+05	2.4E+05	2.4E+04	1.4E+04
f_4	0.00133	67.7123	44.9334	85.5049	85.4186	15.2185	13.0471
f_5	1.3E+02	4.1E+09	5.3E+05	9.4E+09	6.5E+07	3.0E+07	1.0E+05
f_6	1.6E−09	4.0E+09	2.2E+08	1.5E+10	5.9E+08	2.4E+06	28.1635
f_7	2.5E−05	2.1E+04	1.4E+03	4.3E+04	2.5E+03	4.4E+03	3.7E+03
f_8	5.4E−05	21.2627	21.0887	21.4234	21.3745	18.4078	17.1200
f_9	1.5E−08	1.4E+02	1.42371	2.4E+02	2.74191	18.7604	0.22863
f_{10}	21.6660	4.7E+05	1.0E+02	9.4E+05	6.7E+03	2.6E+03	16.4156
f_{11}	2.4E+05	2.0E+08	2.1E+07	4.9E+08	1.1E+08	3.0E+08	1.8E+06
f_{12}	0.25614	1.8E+09	2.7E+03	4.2E+09	1.3E+07	7.5E+04	26.5760
f_{13}	1.6E-06	6.9E+04	7.8E+02	1.1E+05	1.0E+03	1.0E+04	58.7252
f_{14}	9.6E−11	9.3E+07	4.4E+06	1.0E+09	1.5E+09	3.8E+06	3.3E+05
f_{15}	0.67362	6.7E+02	3.5E+02	8.1E+02	5.5E+02	3.8E+02	3.4E+02
f_{16}	1.0E−05	2.2E+04	1.4E+03	4.3E+04	3.2E+03	3.9E+03	3.7E+03
f_{17}	1.06924	1.4E+02	1.1E+02	1.7E+02	1.6E+02	1.2E+02	1.0E+02
f_{18}	1.3E−05	1.9E+05	1.4E+05	3.0E+05	2.4E+05	2.1E+04	1.9E+04
f_{19}	0.00127	0.49999	0.49998	0.49999	0.49999	0.49774	0.49830
f_{20}	0.00222	96.3208	83.6293	97.0198	95.6162	92.8530	90.5664

Table 4. The ratio of single iteration average time for four algorithms.

D	T^M/T^Q	T^M/T^P	T^M/T^S
50	9.863512	9.800094	9.620418
100	9.785713	10.03969	9.752004
AVG	9.824613	9.919894	9.686211

Table 5. The ratio of average optimization results for four algorithms.

D	$O^M_{G=100}/O^Q$		$O^M_{G=100}/O^P$		$O^M_{G=100}/O^S$	
	$G=10^2$	$G=10^3$	$G=10^2$	$G=10^3$	$G=10^2$	$G=10^3$
50	0.001361	0.165868	0.000671	0.067214	0.002587	0.140945
100	0.000623	0.012133	0.000510	0.000795	0.001124	0.073973
AVG	9.92E-04	0.089001	5.91E-04	0.034004	0.001856	0.107459

amplitude coding and evolutionary mechanisms can indeed improve the optimization capability. From **Table 5**, in the same iteration steps, the optimization result of MQPAPSO is only one thousandth of that of QDPSO. On the other hand, in the same running time, the optimization result of MQPAPSO is only nine percent of QDPSO. Experimental results show that multi-bit probability amplitude coding method can indeed significantly improve the optimization ability of the traditional PSO algorithm and other similar algorithms.

8. Conclusion

In this paper, a quantum-inspired particle swarm optimization algorithm is presented encoded by probability amplitudes of multi-qubits. Function extreme optimization results show that under the same running time, the optimization ability of proposed algorithm has greatly superior to the traditional methods, revealing that the multi-qubits probability amplitude encoding method indeed greatly enhances the ability of traditional particle swarm optimization performance.

Funding

This work was supported by the Youth Foundation of Northeast Petroleum University (Grant No. 2013NQ119) and the National Natural Science Foundation of China (Grant No. 61170132).

References

[1] Kennedy, J. and Eberhart, R.C. (1995) Particle Swarms Optimization. *Proceedings of IEEE International Conference on Neural Networks*, New York, November/December 1995, 1942-1948. http://dx.doi.org/10.1109/icnn.1995.488968

[2] Guo, W.Z., Chen, G.L. and Peng, S.J. (2011) Hybrid Particle Swarm Optimization Algorithm for VLSI Circuit Partitioning. *Journal of Software*, **22**, 833-842. http://dx.doi.org/10.3724/SP.J.1001.2011.03980

[3] Qin, H., Wan, Y.F., Zhang, W.Y. and Song, Y.S. (2012) Aberration Correction of Single Aspheric Lens with Particle Swarm Algorithm. *Chinese Journal of Computational Physics*, **29**, 426-432.

[4] Cai, X.J., Cui, Z.H. and Zeng, J.C. (2008) Dispersed Particle Swarm Optimization. *Information Processing Letters*, **105**, 231-235. http://dx.doi.org/10.1016/j.ipl.2007.09.001

[5] Liu, Y., Qin, Z. and Shi, Z.W. (2007) Center Particle Swarm Optimization. *Neurocomputing*, **70**, 672-679. http://dx.doi.org/10.1016/j.neucom.2006.10.002

[6] Zhang, Y.J. and Shao, S.F. (2011) Cloud Mutation Particle Swarm Optimization Algorithm Based on Cloud Model. *Pattern Recognition and Artificial Intelligence*, **24**, 90-96.

[7] Fang, W., Sun, J., Xie, Z.P. and Xu, W.B. (2010) Convergence Analysis of Quantum-Behaved Particle Swarm Optimization Algorithm and Study on Its Control Parameter. *Acta Physica Sinica*, **59**, 3686-3694.

[8] Sun, J., Wu, X.J., Fang, W., Lai, C.H. and Xu, W.B. (2012) Conver Genceanalysis and Improvements of Quantum-Behaved Particle Swarm Optimization. *Information Sciences*, **193**, 81-103. http://dx.doi.org/10.1016/j.ins.2012.01.005

[9] Lu, T.C. and Yu, G.R. (2013) An Adaptive Population Multi-Objective Quantum Inspired Evolutionary Algorithm for Multi-Objective 0/1 Knapsack Problems. *Information Sciences*, **243**, 39-56. http://dx.doi.org/10.1016/j.ins.2013.04.018

[10] Abdesslem, L. (2013) A Hybrid Quantum Inspired Harmony Search Algorithm for 0-1 Optimization Problems. *Journal of Computational and Applied Mathematics*, **253**, 14-25. http://dx.doi.org/10.1016/j.cam.2013.04.004

[11] Gao, J.Q. (2011) A Hybrid Quantum Inspired Immune Algorithmfor Multi Objective Optimization. *Applied Mathematics and Computation*, **217**, 4754-4770. http://dx.doi.org/10.1016/j.amc.2010.11.030

[12] Han, K.H. and Kim, J.H. (2002) Quantum-Inspired Evolutionary Algorithm for a Class of Combinatorial Optimization. *IEEE Transactions on Evolutionary Computation*, **6**, 580-593. http://dx.doi.org/10.1109/TEVC.2002.804320

[13] Liu, X.D., Li, P.C. and Yang, S.Y. (2014) Design and Implementation of Quantum-Inspired Differential Evolution Algorithm. *Journal of Signal Processing*, **30**, 623-633.

[14] Eberhart, R.C. and Shi, Y. (2000) Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. *Proceedings of IEEE Congress on Evolutionary Computation*, New York, 84-88.

[15] Li, P.C., Wang, H.Y. and Song, K.P. (2012) Research on Improvement of Quantum Potential Well-Based Particle Swarm Optimization Algorithm. *Acta Physica Sinica*, **61**, Article ID: 060302.

[16] Zhao, Y.X. and Liu, L.Q. (2013) Emerging Heuristic Optimization Algorithm. Science Press, Beijing.