Adaptive Quantum-Behaved Particle Swarm Optimization Algorithm

Puyong YU
College Of Information Technology
TaiShan University,TSU
Tai'an, China

Xueming BAI
College Of Information Technology
TaiShan University,TSU
Tai'an, China

***Abstract*—**In this paper a new algorithm for solving constrained problem is proposed which is called the AQPSO algorithm (Adaptive Quantum-behaved Particle Swarm Optimization). The AQPSO algorithm outperforms well compared with QPSO algorithm or and PSO algorithm, because the adaptive mechanism is more approximate to the learning process of social organism with high-level swarm intelligence and can make the population evolve persistently. A non-stationary multi-stage assignment penalty is adopted in solving constrained problem to improve the convergence and gain more accurate results. The approach is tested on several accredited benchmark functions.

Keywords*-*AQPSO; QPSO; PSO; swarm intelligence

I. INTRODUCTION

Generally speaking, the constrained optimization problem can be described as the following nonlinear programming problem:

$$\min_x f(x), x \in S \subset \Re^n \tag{1}$$

subject to the linear or nonlinear constraints:

$$\begin{aligned} & g_i(x) \le 0, \quad i = 1,2 \ldots m \\ & \mathrm{h}_j(x) = 0, \quad j = 1,2 \ldots k \\ & a(i) \le x_i \le b_i, 1 \le i \le n. \end{aligned} \tag{2}$$

where $f(x)$ is an objective function, $g_i(x)$ and $h_j(x)$ are equality and inequality constrained functions respectively, $a(i)$ and $b(i)$ are the search space up-bound and low-bound for x_i. The formulation of the constraints in (2) is not restrictive, since an inequality constraint of the form $g_i(x) \ge 0$ can also be represented as $-g_i(x) \le 0$,and the equality constraint $g_i(x) = 0$ is equal to two inequality constraints $g_i(x) \ge 0$ and $g_i(x) \le 0$.

In this paper, a non-stationary function multi-stage assignment penalty function is adopted. The penalty values are dynamically modified according to equality constraints $g_i(x)$ and inequality $h_j(x)$ constraints. A penalty function is generally defined as:

$$F(x) = f(x) + h(k)H(x), \qquad x \in S \subset \Re^n, \tag{3}$$

where $f(x)$ is the original objective function of the CO problem in Equation(1); $h(k)$ is a dynamically modified penalty value, where k is the algorithm's current iteration number; and $H(x)$ is a penalty factor, defined as $H(x) = \sum_{i=1}^{m} \theta(q_i(x)) q_i(x)^{\gamma(q_i(x))}$ where $q_i(x) = \max\{0, g_i(x)\}. \ i = 1 \ldots\ldots m$ The function $q_i(x)$ is a relative violated function of the constraints; $\theta(q_i(x))$ is a multi-stage assignment function; $\gamma(q_i(x))$ is the power of the penalty function; and $g_i(x)$ are the constraints described in Equation (2).

The function $h(\cdot)$, $\theta(\cdot)$ and $\gamma(\cdot)$ are problem dependent. Details of the penalty function used in this study are given in Section 4.

In this paper, we consider to solve the Constrained Optimization (CO) Problems by AQPSO algorithm. The CO problem is tackled through the minimization of a non-stationary multi-stage assignment penalty function. In the next section, the Particle Swarm Optimization (PSO) and the Quantum-Behaved Particle Swarm Optimization (QPSO) is briefly described. In section 3, the Adaptive Quantum-Behaved Particle Swarm Optimization (AQPSO) is reported .The test problems and the results of experiments are reported in Section 4. The paper ends with the conclusion and ideas for future research in Section 5.

II. PSO AND QPSO

Particle Swarm Optimization (PSO), [1] originally proposed by Kennedy and Eberhart in 1995, is a population-based evolutionary computation technique, which differs from other evolution-motivated evolutionary computation in that it is motivated from the simulation of social behavior. In a PSO system, a particle corresponding to individual of the organism, which depicted by its position vector $\vec{x}$ and its velocity vector

Taian Science and Technology plan under Grant No. 2012020517

$\vec{v}$, is a candidate solution to the problem. That is the trajectory of the particle is determined. Then the optimal solution of the probability of moving out the trajectory is ignored. Therefore, in general, PSO can obtain good solutions in high-dimensional spaces but the ignorance of optimal solution does exist and PSO stumbles on local minima.

Keeping to the philosophy of PSO, we proposed a Delta potential well model of PSO in quantum world (QPSO) [4]. Because $\vec{x}$ and $\vec{v}$ of a particle are not determined simultaneously according to uncertainty principle, the term trajectory is meaningless in quantum world.

A. Dynamics of classical PSO

In a classical PSO system proposed by Kennedy and Eberhart, the particles are manipulated according to the following equation:

$$v_i(t+1) = wv_i(t) + \varphi_1(p_i - x_i(t)) + \varphi_2(p_g - x_i(t)) \quad (4)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (5)$$

$$i = (1,2,\cdots,M)$$

Where x and v denotes the position and velocity of particle i among the population correspondingly, $\varphi_1{}^T$ and $\varphi_2{}^T$ are two random vectors in the range [0,1].

In (4), vector p_i is the best position (the position giving the best fitness value) of the particle i and vector p_g is the position of the best particle among all the particles in the population. Parameter w is the inertia weight [2], which does not appear in the original version of PSO [1]. In [3], M. Clerc and J. Kennedy analyze the trajectory and prove that, whichever model is employed in the PSO algorithm, each particle iin the PSO system converges to its local point p , whose coordinates are $p_d = (\varphi_{1d} p_{id} + \varphi_{2d} p_{gd}) / (\varphi_{1d} + \varphi_{2d})$ so that the best previous position of all particles will converge to an exclusive global position with $t \to \infty$.

B. Dynamics of quantum PSO

In Quantum-Behaved Particle Swarm Optimization (QPSO) [1, 4], the particles move according to the following equation:

$$mbest = \frac{1}{M}\sum_{i=1}^{M} p_i = \left(\frac{1}{M}\sum_{i=1}^{M} p_{i1}, \frac{1}{M}\sum_{i=1}^{M} p_{i2}, \ldots, \frac{1}{M}\sum_{i=1}^{M} p_{id}\right) \quad (6)$$

$$p_{id} = \varphi * p_{id} + (1-\varphi) * p_{gd}, \varphi = rand() \quad (7)$$

$$X_{id} = p_{id} \pm \alpha * |mbest_d - X_{id}| * \ln(\frac{1}{u}), u = rand() \quad (8)$$

where mbest is the mean best position among the particles. φ and u are a random number distributed uniformly on [0,1] respectively and α is the only parameter in QPSO algorithm.

When Quantum-behaved Particle Swarm Optimization algorithm is applied to practical problems, there are several control methods for parameter α .A simple one is that α is set to be a fixed value when the algorithm is running. But this approach is lack of robustness.

Another efficient method is linear-decreasing method that decreasing the value of α linearly as the algorithm is running. That is, the value of α is determined by

$$\alpha = (\alpha_1 - \alpha_2) \times \frac{(MAXITER - t)}{MAXITER} + \alpha_2 \quad (9)$$

The Quantum-behaved Particle Swarm Optimization (QPSO) Algorithm in is described as follows:

1) Initialize an array of particles with random position and velocities inside the problem space

2) Determine the mean best position among the particles by (6)

3) Evaluate the desired objective function (for example minimization) for each particle and compare with the particle's previous best values: If the current value is less than the previous best value, then set the best value to the current value. That is, if $f(x_i) < f(p_i)$ then $x_i = p_i$.

4) Determine the current global position minimum among the particle's best positions. That is:

$g = \arg \min_{1 \le i \le M} f(p(i))$ (M is the population size).

5) Compare the current global position to the previous global: if the current global position is less than the previous global position; then set the global position to the current global.

6) For each dimension of the particle, get a stochastic point between p_{id} and p_{gd}

$$p_{id} = \varphi * p_{id} + (1-\varphi) * p_{gd}, \varphi = rand()$$

7) Attain the new position by stochastic equation (6)

8) Repeat steps 2)-7) until a stop criterion is satisfied OR a pre-specified number of iterations are completed.

III. AQPSO

Like many other evolutionary algorithms, the major problem confronts Quantum Particle Swarm Optimization Algorithm is premature convergence, which results in great performance loss and sub-optimal solutions. With QPSOs the fast information flow between particles seems to be the reason for clustering of particles. Diversity declines rapidly, leaving the QPSO algorithm with great

difficulties of escaping local optima. Another problem with QPSO in multi-modal optimization is computational cost. With the dimension of the optimization problem increasing, the population size must be enlarged to ensure the algorithm have a good performance, which, however, makes the algorithm computationally expensive.

To solving the aforementioned problems, we propose in this paper a Adaptive Quantum Particle Swarm Optimization (AQPSO) Algorithm.

In QPSO, Contraction-Expansion Coefficient is a vital parameter to the convergence of the individual particle in QPSO, and therefore exerts significant influence on convergence of the algorithm [2]. Mathematically, there are many forms of convergence of stochastic process, and different forms of convergence have different conditions that the parameter must satisfy. In this paper, we do not mean to analyze theoretically the convergence process of the individual particle in QPSO, but implement stochastic simulation to discover the knowledge about convergence of the particle.

For simplicity, we consider the evolution equation (8) of QDPSO [3] in one-dimensional space. The p_{id} is denoted as point p .In practice, when $t \to \infty$, the point p of the individual particle and the Mean Best Position point $mbest$ will converge to the same point, and consequently, the particle in QDPSO and that in QPSO have the same convergence condition for parameter α except that they have different convergence rate.

From the results of stochastic simulation, we can conclude that when $\alpha \le 1.7$, the particle will converge to the point p , and when $\alpha \ge 1.8$, it will diverge. Therefore there must be such a threshold value $\alpha_0 \in [1.7, 1.8]$ that the particle converges if $\alpha \le \alpha_0$, and diverges otherwise. To get more precise value of α_0 we need to do simulation experiment with α set to be the value between 1.7 and 1.8 by more times.

However, for practical application of QPSO, the knowledge about parameter α we acquired so far is adequate.

The better parameter control method is to use adaptive mechanism. Firstly, we introduce the following error function

$$\Delta F = (F_{pbest} - F_{gbest}) / MIN(ABS(F_{pbest}), ABS(F_{gbest})) \quad (10)$$

where Fi is the fitness of the ith particle, Fgbest is the fitness of gbest, ABS(x) gests the absolute value of x, and MIN(x1,x2) gets the minimum value between x1 and x2.

Let $z = \log(\Delta F)$. , then the function is

$$\alpha(z) = \begin{cases} 0.6 & z > 0 \\ 0.7 & -2 < z \le 0 \\ 0.6 + 0.1 \times k & -k-1 < z \le -k \ (k = 2,3,4) \\ 1.0 + 0.2 \times (k-4) & -k-1 < z \le -k \ (k = 5,6,7) \\ 1.8 & z \le -8 \end{cases} \quad (11)$$

The QPSO employing the above adaptive function is called Adaptive Quantum-behaved Particle Swarm Optimization (AQPSO).

IV. TEST PROBLEMS AND EXPERIMENTAL RESULT

In our experiments, the population of the swarm was set equal to 100. We recorded mean best fitness values for 10 runs of each test problem, and the PAQPSO algorithm run for 1000 iterations in each case. A violation tolerance was used for the constraints. Thus, a constraint $g_i(x)$ assumed to be violated, only if $g_i(x) > 10^{-5}$. Regarding the penalty parameters, the same values as the values reported in [5] were used, to obtain results comparable to the results obtained using different EA. Specially, if $q_i(x) < 1$, then $\gamma(q_i(x)) = 1$,otherwise $\gamma(q_i(x)) = 2$ .Moreover, if $q_i(x) < 0.001$,then $\theta(q_i(x)) = 10$,else if $q_i(x) \le 0.1$ the $\theta(q_i(x)) = 20$,else if $q_i(x) \le 1$ then $\theta(q_i(x)) = 100$,otherwise $\theta(q_i(x)) = 300$ .Regarding the function $h(\cdot)$,it was set to $h(k) = \sqrt{k}$ for Test Problem 1 and $h(k) = k\sqrt{k}$ for the rest problems.

The test problems are defined immediately below:

TABLE I. TEST FUNCTIONS

F	*Mathematical Representation*	*Subjection*	*Best solution*
f1	$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$	$x_1 = 2x_2 - 1,\ x_1^2/4 + x_2^2 - 1 \le 0$	$f^* = 1.3934651$

f2	$f(x)=(x_1-10)^3+(x_2-20)^3$	$100-(x_1-5)^2-(x_2-5)^2\le 0,(x_1-6)^2+(x_2-5)^2-82.81\le 0,13\le x_1\le 100,0\le x_2\le 100$	$f^* = -6961\ .81381$
f3	$f(x)=(x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6+7x_6^2+x_7^4-4x_6x_7-10x_6-8x_7$	$-127+2x_1^2+3x_2^4+x_3+4x_4^2+5x_5\le 0,$ $-282+7x_1+3x_2+10x_3^2+x_4-x_5\le 0,$ $-196+23x_1+x_2^2+6x_6^2-8x_7\le 0,4x_1^2+x_2^2-3x_1x_2+2x_3^2+5x_6-11x_7\le 0,$ $-10\le x_i\le 10, i=1.....7$	$f^* = 680\ .630057$
f4	$f(x)=5.3578547x_3^2+0.8356891x_1x_5+37.293239x_1-40792.141$	$0\le 85.334407+0.0056858T_1+T_2x_1x_4-0.0022053x_3x_5\le 92,$ $90\le 80.51249+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^2\le 110$ $20\le 9.300961+0.0047026x_2x_5+0.0012547x_1x_3+0.0019085x_3x_4\le 25,78\le x_1\le 102,$ $33\le x_2\le 45,27\le x_i\le 45, i=3,4,5,$ *where* $T_1=x_2x_5$ *and* $T_2=0.0006262$	$f^*=-30665.538$
f5	$f(x)=5.3578547x_3^2+0.8356891x_1x_5+37.293239x_1-40792.141$	$0\le 85.334407+0.0056858T_1+T_2x_1x_4-0.0022053x_3x_5\le 92,$ $90\le 80.51249+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^2\le 110$ $20\le 9.300961+0.0047026x_2x_5+0.0012547x_1x_3+0.0019085x_3x_4\le 25,78\le x_1\le 102,$ $33\le x_2\le 45,27\le x_i\le 45, i=3,4,5,$ $T_1=x_2x_3, T_2=0.00026$	unknown
f6	$f(x,y)=-10.5x_1-7.5x_2-3.5x_3-2.5x_4-1.5x_5-10y-0.5\sum_{i=1}^{5}x_i^2$	$6x_1+3x_2+3x_3+2x_4+x_5-6.5\le 0,10x_1+10x_3+y\le 20,0\le x_i\le 1, i=1,.....5,0\le y.$	$f^* = -213\ .0$

For each test problem, the mean and the best solution obtained in all 10 runs were recorded. The results for all test problems are reported in Table2.. The mean run-time for 10 runs of each functions in Table 3. In most cases AQPSO outperformed the results reported in [5] for other QPSO and PSO algorithm. Especially for test problem 4 the result can reach around its theoretical value. So the result of test problem 5 obtained from AQPSO algorithm may be even closer to its unknown theoretical value. Proper fine-tuning parameters of AQPSO may result in better solutions.

TABLE II. MEAN AND THE BEST SOLUTION FOR EACH TEST PROBLEM IN 10 RUNS

F	*Method*	*Mean*	*Best Solution*	*F*	*Method*	*Mean*	*Best Solution*
f1	PSO-In	1.394006	1.393431	f4	PSO-In	-31526.304	-31543.484
	PSO-Co	1.393431	1.393431		PSO-Co	-31528.289	-31542.578
	QPSO	1.39346498	1.39346498		QPSO	-30665.535	-30665.5382
	AQPSO	1.393438	1.39346382		AQPSO	-30665.5387	-30665.5381
f2	PSO-In	-6960.866	-6961.798	f5	PSO-In	-31523.859	-31544.036
	PSO-Co	-6961.836	-6961.837		PSO-Co	-31526.308	-31543.312
	QPSO	-6961.7274	-6961.80434		QPSO	-31026.428	-31026.4277
	AQPSO	-6961.7938	-6961.8120		AQPSO	-31045.426	-31026.3652
f3	PSO-In	680.671	680.639	f6	PSO-In	-213.0	-213.0
	PSO-Co	680.663	680.635		PSO-Co	-213.0	-213.0
	QPSO	680.646034	680.635235		QPSO	-213.0	-213.0
	AQPSO	680.637	680.634521		AQPSO	-213.0	-213.0

V. CONCLUSION

According to the experimental results, in most cases the performance of the AQPSO method in coping with constrained optimization problems is better than QPSO and PSO algorithm [5], and than those obtained through other EAs.

Future work will focus on approaches of solving large-scale problems or apply it to problems with the run-time demands strictly.

REFERENCES

[1] Others: J. Kennedy and R. Eberhart, "Particle Swarm Optimization", Proc. IEEE Conf. On Neural Network, 1942-1948 (1995)

[2] Others: Y. Shi and R. Eberhart, "Empirical study of particle swarm optimization," Proc. Congress on Evolutionary Computation, 1945-1950 (1999)

[3] Others: J. Sun and Wenbo Xu, Parameter "Selection of Quantum-behaved Particle Optimization". ICNC 2005, LNCS 3612, pp. 543 – 552, 2005.© Springer-Verlag Berlin Heidelberg 2005.

[4] Others: Jun Sun and Wenbo Xu, "Particle Swarm Optimization with Particles Having Quantum Behaviour" IEEE Congress. Evolutionary Computation (2004)

[5] Others: K.E. Parsopoulos and M.N. Vrahatis, "Particle swarm optimization method for constrained optimization problems", Intelligent Technologies-Theory and Application, 214-220 (2002)