Quantum-Behaved Particle Swarm Optimization for Integer Programming

Jing Liu, Jun Sun, and Wenbo Xu

Center of Intelligent and High Performance Computing,
School of Information Technology, Southern Yangtze University
No. 1800, Lihudadao Road, Wuxi,
214122 Jiangsu, China
{liujing_novem, sunjun_wx, xwb_sytu}@hotmail.com

Abstract. Based on our previously proposed Quantum-behaved Particle Swarm Optimization (QPSO), this paper discusses the applicability of QPSO to integer programming. QPSO is a global convergent search method, while the original Particle Swarm (PSO) cannot be guaranteed to find out the optima solution of the problem at hand. The application of QPSO to integer programming is the first attempt of the new algorithm to discrete optimization problem. After introduction of PSO and detailed description of QPSO, we propose a method of using QPSO to solve integer programming. Some benchmark problems are employed to test QPSO as well as PSO for performance comparison. The experiment results show the superiority of QPSO to PSO on the problems.

1 Introduction

An Integer programming problem is an optimization problem in which some or all of variables are restricted to take on only integer values. Thus the general form of a mathematical Integer Programming model can be stated as:

$$\begin{aligned} &\min_{x} \ f(x) \\ &s.t. \quad g(x) \le b, \quad x \in X \end{aligned} \tag{1}$$

where

$$X = \{x \in \Re^n : x_i \in Z^n, \forall i \in M\}, \ \ M \subseteq [1..n]$$

This type of model is called a mixed-integer linear programming model, or simply a mixed-integer program (MIP). If $M = [1...n]$, we have a pure integer linear programming model, or integer program (IP). Here we will consider only the simple and representative minimization IP case, though maximization IP problems are very common in the literature, since a maximization problem can be easily turned to a minimization problem. For simplicity, in this paper it will be assumed that all of the variables are restricted to be integer valued without any constraints.

Evolutionary and Swarm Intelligence algorithms are stochastic optimization methods that involve algorithmic mechanisms similar to natural evolution and social behavior respectively. They can cope with problems that involve discontinuous

I. King et al. (Eds.): ICONIP 2006, Part II, LNCS 4233, pp. 1042 – 1050, 2006.

objective functions and disjoint search spaces [7][8]. Early approaches in the direction of Evolutionary Algorithms for Integer Programming are reported in [9][10]. The performance of PSO method on Integer Programming problems was investigated in [6] and the results show that the solution to truncate the real value to integers seems not to affect significantly the performance of the method.

In this paper, the practicability of QPSO to integer programming is explored. For QPSO is global convergent, it can be expected to outperform PSO in this field. To test the algorithm, numerical experiment is implemented. The paper is organized as follows. In Section 2 we describe the concepts of QPSO. Section 3 presents the numerical results of both QPSO and PSO on several benchmark problems. The paper is concluded in Section 4.

2 Quantum-Behaved Particle Swarm Optimization

In this section, the concept of Quantum-behaved Particle Swarm Optimization is described following the introduction of the original Particle Swarm Optimization.

2.1 Particle Swarm Optimization

The PSO algorithm is population based stochastic optimization technique proposed by Kennedy and Eberhart in 1995[1]. The motivation for the development of this method was based on the simulation of simplified animal social behaviors such as fish schooling, bird flocking, etc.

In the original PSO model, each individual is treated as volumeless and defined as a potential solution to a problem in D-dimensional space, with the position and velocity of particle i represented as $X_i = (x_{i1}, x_{i2}, \dots x_{iD})$ and $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$, respectively. Each particle maintains a memeory of its previous best positon $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})$ and p_{gd}, designated g, represents the position with best fitness in the local neighborhood. The particle will move according to the following equation:

$$\begin{aligned} v_{id} &= v_{id} + \varphi_1 * rand(\ ) * (p_{id} - x_{id}) + \varphi_2 * rand(\ ) * (p_{gd} - x_{id}) \\ x_{id} &= x_{id} + v_{id} \end{aligned} \tag{2}$$

where φ_1 and φ_2 determine the relative influence of the social p_g and cognition p_i components, which are the embodiment of the spirit of cooperation and competition in this algorithm.

Since the introduction of PSO method in 1995, considerable work has been done in the aspect of improving its convergence, diversity and precision etc. Generally, in population-based search optimization methods, proper control of global exploration and local exploration is crucial in finding the optimum solution effectively. In[2] Eberhart and Shi show that PSO searches wide areas effectively, but tends to lack search precision. So they proposed the solution to introduce ω, a linearly varying inertia weight, that dynamically adjusted the velocity over time, gradually focusing PSO into a local search:

$$v_{id} = \omega * v_{id} + \varphi_1 * rand(\ ) * (p_{id} - x_{id}) + \varphi_2 * rand(\ ) * (p_{gd} - x_{id}) \tag{3}$$

The improved PSO is called Standard PSO algorithm(in this paper PSO-w denoted).

Then Maurice Clerc introduced a constriction factor[3] , K ,that improved PSO's ability to prevent the particles from exploding outside the desirable range of the search space and induce convergence. The coefficient K is calculated as:

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \text{ where } \varphi = \varphi_1 + \varphi_2, \varphi > 4 \tag{4}$$

and the PSO is then

$$v_{id} = K * (v_{id} + \varphi_1 * rand(\ ) * (p_{id} - x_{id}) + \varphi_2 * rand(\ ) * (p_{gd} - x_{id})) \tag{5}$$

2.2 Quantum-Behaved Particle Swarm Optimization

Even though many improvements on PSO methods were emerged, some questions around traditional PSO still exist. In traditional PSO system, a linear system, a determined trajectory and the bound state is to guarantee collectiveness of the particle swarm to converge the optimal solution. However, in such ways, the intelligence of a complex social organism is to some extend decreased. Naturally, Quantum theory, following the Principle of State Superposition and Uncertainty, was introduced into PSO and the Quantum-behaved PSO algorithm was proposed by Jun Sun et al[4].

Keeping to the philosophy of PSO, a Delta potential well model of PSO in quantum world is presented, which can depict the probability of the particle's appearing in position x from probability density function $|\psi(x,t)|^2$, not limited to determined trajectory, with the center on point p(pbest). The wave function of the particle is:

$$\psi(x) = \frac{1}{\sqrt{L}} \exp(-\|p - x\| / L) \tag{6}$$

And the probability density function is

$$Q(x) = |\psi(x)|^2 = \frac{1}{L} \exp(-2\|p - x\| / L) \tag{7}$$

The parameter $L(t+1) = 2 * \alpha * |p - x(t)|$ depending on energy intension of the potential well specifies the search scope of a particle. From the expression of L, we can see that it is so unwise to deploy the individual's center pbest to the swarm that unstable and uneven convergence speed of an individual particle will result premature of the algorithm when population size is small. Then a conception of Mean Best Position (mbest) is introduced as the center-of-gravity position of all the particles [5]. That is

$$mbest = \sum_{i=1}^{M} p_i \Big/ M = \left(\sum_{i=1}^{M} p_{i1} \Big/ M, \sum_{i=1}^{M} p_{i2} \Big/ M, \ldots, \sum_{i=1}^{M} p_{id} \Big/ M \right) \tag{8}$$

here M is the population size and p_i is the pbest of particle i. Thus the value of L is given by $L(t+1) = 2 * \beta * |mbest - x(t)|$. We can see the only parameter in this algorithm is β, called Creativity Coefficient, working on individual particle's convergence speed and performance of the algorithm.

Through the Monte Carlo stochastic simulation method, derived from probability density function, the position of a particle that is vital to evaluate the fitness of a particle can be given by $x(t) = p \pm \frac{L}{2} \ln\left(\frac{1}{u}\right)$. Replacing parameter L, the iterative equation of Quantum-behaved PSO (denoted QPSO-β) is:

$$x(t+1) = p \pm \beta * |mbest - x(t)| * \ln\left(\frac{1}{u}\right) \tag{9}$$

3 Experiments

3.1 Experiment Setting and Benchmark Problems

The method of Integer Programming by PSO and QPSO algorithm is to truncate each particle of the swarm to the closest integer, after evolution according to Eq(2) and Eq(9). In our experiments, each algorithm was tested with all of the numerical test

Table 1. Benchmark problems

F	Mathematical Representation	Solution	Solution
F1	$F_1(x) = \|x\|_1$	X=0	$F_1(x) = 0$
F2	$F_2(x) = x^T x$	X=0	$F_2(x) = 0$
F3	$F_3(x) = -(15\ 27\ 36\ 18\ 12)x + x^T \begin{pmatrix} 35 & -20 & -10 & 32 & -10 \\ -20 & 40 & -6 & -31 & 32 \\ -10 & -6 & 11 & -6 & -10 \\ 32 & -31 & -6 & 38 & -20 \\ -10 & 32 & -10 & -20 & 31 \end{pmatrix} x$	$x = (0,11,22,16,6)^T$ $x = (0,12,23,17,6)^T$	$F_3(x) = -737$
F4	$F_4(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$	$x = (3,2)^T$	$F_4(x) = 0$
F5	$F_5(x) = (9x_1^2 + 2x_2^2 - 11)^2 + (3x_1 + 4x_2^2 - 7)^2$	$x = (1,1)^T$	$F_5(x) = 0$
F6	$F_6(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$	$x = (1,1)^T$	$F_6(x) = 0$
F7	$F_7(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$	$x = (0,0,0,0)^T$	$F_7(x) = 0$

Table 2. Dimension,swarm size and maximum numberof iterations for Test Functions F1-F7

Function	Dim	Swarm Size	Max Iteration
F1	5	20	1000
	10	50	1000
	15	100	1000
	20	200	1000
	25	250	1500
	30	300	2000
F2	5	20	1000
F3	5	150	1000
F4	2	20	1000
F5	2	20	1000
F6	2	50	1000
F7	4	40	1000

problems shown in Table 1[6]. The solution of the equation F(x)=0 except the function F3.. In Table 2 exhibit the swarm's size, the maximum number of iterations as well as dimension for all test functions. For all experiments the initial swarm was taken uniformly distributed inside $[-100,100]^D$,where D is the dimension of the corresponding problem.

In QPSO algorithm, the only parameter setting is Creativity Coefficient β [5],which was gradually decreased for each of the intervals $[1.2, 0.4]$, $[1.0, 0.4]$, $[0.8, 0.4]$ with the number of iterations. And in PSO algorithm, the parameters used for all experiments were $\varphi_1 = \varphi_2 = 2$ and ω was gradually decreased for each of the intervals $[1.2, 0.4]$, $[1.0, 0.4]$, $[0.8, 0.4]$ during the maximum allowed number of iterations. Each of the experiments was repeated 50 runs and the success rate to correct solution as well as the mean number of iterations for each test were recorded.

3.2 Results

The results of PSO and QPSO for the test problems $f_1 - f_7$ are shown in Table 3 and Table 4. Its mean iteration is generated from all tests included incorrect experiments. From the point view of success rate, as shown in Table3, to PSO algorithm, PSO-w is the best choice when ω is gradually from 1.0 to 0.4. Also, to QPSO, β from 1.2 to 0.4 is better than the other two internals as shown in Table 4.

But from the point view of mean iterations, based on the 100 percent success rate, QPSO mostly can reach the correct solution faster than PSO as shown in Table 5. Especially, to test function f1, when dimension is high, the results show that PSO is a better choice, which is because Quantum-behaved PSO algorithm is much fit for global search, especially for higher dimension, and more particles [5].

The convergence graphs for selected test problems are shown in Figure 1, which plot test function value with the number of iteration. As we can see the convergence speed in QPSO is much faster than PSO algorithm.

Table 3. Dimension, Success Rate, Mean Iterations for PSO-w for test F1-F7

F	D	PSO-w					
		w: [1.2,0.4]		w: [1.0,0.4]		w:[0.8,0.4]	
		Succ Rate	Mean Iter	Succ Rate	Mean Iter	Succ Rate	Mean Iter
F1	5	100%	422.27	**100%**	72.8	100%	20.27
	10	100%	434.53	**100%**	90.26	100%	24.77
	15	100%	439.1	**100%**	94.36	100%	25.8
	20	100%	441.67	**100%**	96.3	100%	28.97
	25	100%	653.33	**100%**	99.44	100%	31.23
	30	100%	863.93	**100%**	103.14	100%	34.2
F2	5	100%	423.5	**100%**	77.82	100%	20.22
F3	5	43.3%	718.63	**100%**	125.03	78%	246.56
F4	2	88%	459.44	**100%**	81.24	83.3%	193.36
F5	2	100%	209.64	**100%**	32.9	100%	9.07
F6	2	80%	520.33	**100%**	41.4	56.7%	442.77
F7	4	100%	444.3	**100%**	79.6	100%	34.8

Table 4. Dimension, Success Rate, Mean Iterations for QPSO- β for test F1-F7

F	D	QPSO- β					
		β: [1.2,0.4]		β: [1.0,0.4]		β: [0.8,0.4]	
		Succ Rate	Mean Iter	Succ Rate	Mean Iter	Succ Rate	Mean Iter
F1	5	**100%**	27.2	100%	21.62	100%	15.14
	10	**100%**	48.26	100%	50.52	100%	27.44
	15	**100%**	64.46	100%	82.26	100%	38.08
	20	**100%**	72.24	100%	120.02	100%	47.74
	25	**100%**	83.86	100%	161.56	100%	58.58
	30	**100%**	92.56	100%	202.44	100%	67.86
F2	5	**100%**	21.2	100%	21.56	100%	15.9
F3	5	**100%**	166.9	84%	284.64	48%	543.9
F4	2	**100%**	19.35	100%	25.8	100%	16.82
F5	2	**100%**	8.95	98%	28.66	98%	28
F6	2	**100%**	14.9	94%	88.14	90%	120.8
F7	4	**100%**	65.7	100%	43.9	100%	33.6

Table 5. Success Rate, Mean Iteration for PSO and QPSO

F	D	PSO-w		QPSO- β	
		w: [1.0,0.4]		β:[1.2,0.4]	
		Succ Rate	Mean Iter	Succ Rate	Mean Iter
F1	5	100%	72.8	100%	27.2
	10	100%	90.26	100%	48.26
	15	100%	94.36	100%	64.46
	20	100%	96.3	100%	72.24
	25	100%	99.44	100%	83.86
	30	100%	103.14	100%	92.56

Table 6. (*Continued*)

F2	5	100%	77.82	100%	21.2
F3	5	100%	125.03	100%	166.9
F4	2	100%	81.24	100%	19.35
F5	2	100%	32.9	100%	8.95
F6	2	100%	41.4	100%	14.9
F7	4	100%	79.6	100%	65.7

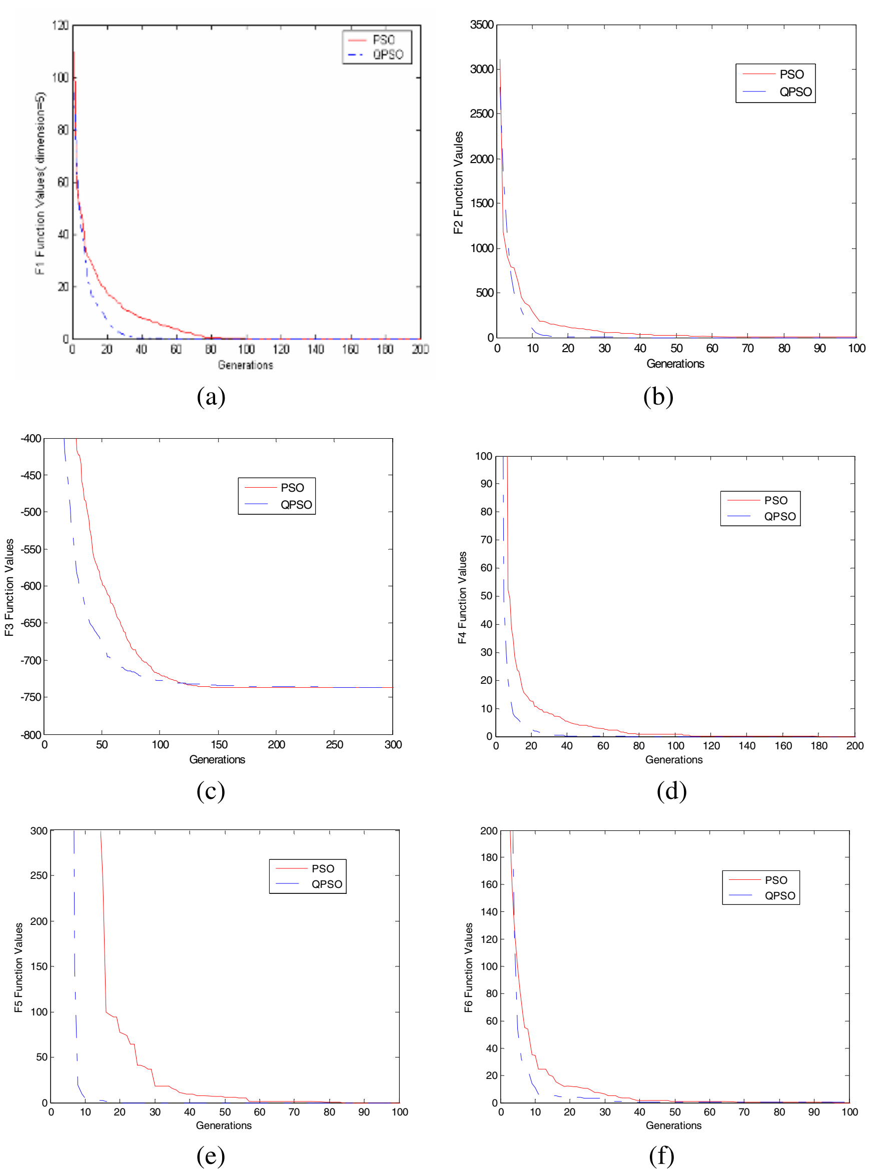
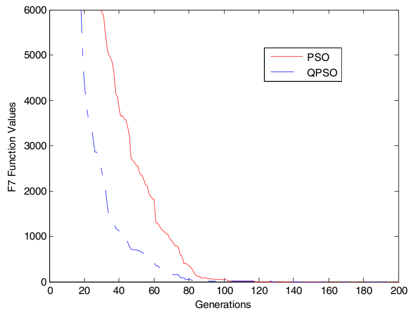


Fig. 1. Test function value with generations. (a) F1 (b) F2 (c) F3 (d) F4 (e) F5 f) F6 (g) F7.



(g)

Fig. 1. (*continued*)

4 Conclusions

In this paper, we have applied QPSO to integer programming problem. The experiment results on benchmark functions show that QPSO with proper intervals of parameter β can search out the global optima more frequently than PSO, for QPSO can be guaranteed to converge global optima with probability 1 when iteration number $t \to \infty$. Not only QPSO is superior to PSO in this type of problems, but in other optimization problem such as constrained nonlinear program also [11].

Integer programming (IP) is a very important discrete optimization problem. Many of combinatory optimization (CO) can be reduce to IP. Therefore, an efficient technique to solving IP problem can be employed to many CO problems. Based on the work in this paper, which is our first attempt to use QPSO to solve discrete optimization problem, the future work will focus on practicability of QPSO on some NP-complete combinatory problems.

References

1. J. Kennedy, R.C. Eberhart.: Particle Swarm Optimization. Proc of the IEEE InternationalConference on Neural Networks, Piscataway, NJ, USA (1995), 1942-1948
2. Y Shi, R.C.Eberhart: A Modified Particle Swarm Optimizer. Proc of the IEEE Conference on Evolutionary Computation, AK, Anchorage (1998), 69-73
3. M.Clerc, J.Kennedy.: The Particle Swarm: Explosion, Stability and Convergence in a Multi-dimensional Complex Space. IEEE Transactions on Evolutionary Computation (2002), Vol. 6 No.1, 58-73
4. J Sun, B Feng, Wb Xu.: Particle Swarm Optimization with Particles Having Quantum Behavior. IEEE Proc of Congress on Evolutionary Computation (2004), 325-331
5. J Sun, W Xu.: A Global Search Strategy of Quantum-Behaved Particle Swarm Optimization.IEEE conf. On Cybernetics and Intelligent Systems (2004), 111-116
6. K.E.Parsopoulos, M.N.Vrahatis.: Recent Approaches to Global Optimization Problems through Particle Swarm Optimization. Natural Computing, Kluwer Academic Publishers (2002), 235-306

7. D.B.Fogel.: Toward a New Philosophy of Machine Intelligence, IEEE Evolutionary Computation(1995), New York
8. J. Kennedy, R.C. Eberhart.: Swarm Intelligence, Morgan Kaufmann Publishers (2001)
9. D.A.Gall: A Practical Multifactor Optimization Criterion. Recent Advances in Optimization Techniques (1996), 369-386
10. G.Rudolph.: An Evolutionary Algorithm for Integer Programming. Parallel Problem Solving from Nature (1994) 139-148
11. Jing Liu, J Sun, WB Xu.: Solving Constrained Optimization Problems with Quantum Particle Swarm Optimization. Distributed Computing and Algorithms for Business, Engineering, and Sciences (2005) 99-103