An adaptive Gauusian probability distribution based QPSO algorithm for engineering design problems

Qidong Chen
School of Internet of Things
Jiangnan University
Wuxi, China
cqd_jnu@hotmail.com

Jun Sun
School of Internet of Things
Jiangnan University
Wuxi, China
sunjun_wx@hotmail.com

Xiaoqian Shi
School of Internet of Things
Taihu School
Wuxi, China
18806186316@163.com

Wei Fang
School of Internet of Things
Jiangnan University
Wuxi, China
fangwei@jiangnan.edu.cn

Liwei Li
School of Internet of Things
Jiangnan University
Wuxi, China
1002728846@qq.com

Si Wang
School of Internet of Things
Jiangnan University
Wuxi, China
905406307@qq.com

***Abstract*—This paper presents an adaptive Gaussian probability distribution based quantum-behaved particle swarm optimization algorithm for solving engineering design optimization problems with multiple constraints. Through adjusting the bondage domain centered on learning inclination points adaptively by using mutation operator with adaptive Gaussian probability distribution, the proposed algorithm can enhance the local search ability without lose much global search capacity. In order to verify performance of the proposed algorithm, two well-studied engineering design problems are described and evaluated with several runs. Our results shows that our algorithm handle these two problems efficiency in terms of precision and robustness compared to the algorithms presented in the literature.**

Keywords—engineering design problems, bondage domain, learning inclination points, adaptive Gaussian probability distribution

I. INTRODUCTION

Recently, many researchers use swarm intelligence algorithms, a new class of metaheuristics, to solve EDPs. PSO has become the focus in optimization community and has been extensively studied over the past twenty years. PSO begins with a population of candidate solutions, also called particles group, and then it improves each candidate solution iteratively until the termination condition is reached. Since particles aggregate to their local best position or the global best position, sometimes the algorithm falls into the local optima and in some situations premature convergence and stagnation occur [21-23]. At the same time, the performance of PSO also depends on the algorithm parameters [24].

Generally, particle swarm optimization models can be divided into two types, namely are global optimization models and local optimization models, according to whether the particles exchange the information with the whole population [25]. Kennedy proved that the global model has a faster convergence speed, while it is easier to trap it into local optima [17]. For the purpose of mitigating the influence of those defects, researchers have proposed various improvements with some adjustments or some modifications. These can be divided into four categories: particle swarm initialization [26], neighborhood topology [29], parameter selection [27-28], and blending strategies [30].

In order to solve EDPs, He and Wang first introduce an effective co-evolutionary particle swarm optimization (PSO) algorithm to solve EDPs [1], and then many improved PSO algorithm are proposed and had implemented in EDPs[4][6]. Khamsawang et al. propose a hybrid PSO-DE algorithm to enhance the local search ability of PSO, and performs well on economic dispatch problem [2]. Garg and Harish also propose a hybrid PSO-DE algorithm for solving constrained algorithms [3]. Quantum-behaved particle swarm optimization (QPSO) algorithm is one of most robustness variants PSO algorithm [7]. Coelho considered the slow convergence speed of QPSO and proposed the gaussian quantum-behaved particle swarm optimization (G-QPSO) algorithm so as to enhance the local search ability of QPSO[5]. In G-QPSO algorithm, random numbers are generated using Gaussian distribution sequences with zero mean and unit variance for the stochastic coefficients of QPSO, which may avoid particles to move away from the current point and escape from local minima.

Despite G-QPSO algorithm obtained good results on EDPs compared to QPSO, but it sacrifice its global search performance to accelerate convergence for better results. Therefore, we propose an adaptive Gaussian probability distribution based quantum-behaved particle swarm optimization (AG-QPSO) algorithm to enhance its local search ability for solving EDPs. In AG-QPSO, the bondage domain centered on learning inclination points (LIPs) is shrinking adaptively by adjusting the Gaussian probability distribution to generate Gaussian sequence rather than random sequence generated by uniform distribution, which can enhance the local search ability without lose much global search capacity.

The rest of the paper is organized as follows: The features of AG-QPSO algorithm and the procedure of constrains handling based on penalty are described in Section II. In Section III, two well-studied engineering design problems are described in detail. Section IV presents the results of the optimization by AG-QPSO and the existing algorithms that deal with EDPs. Finally, Section V outlines the conclusion of paper.

II. AG-QPSO Algorithm

Engineering design problems (EDPs) are constrained optimization problems (COPs) that can be described as:

$$minf(X)\ s.t.\begin{cases} g_j(X) \le 0 \\ h_k(X) = 0 \\ XMIN_i \le X_i \le XMAX_i \end{cases} \quad (1)$$

where $j = 1,2,\cdots,J$, $k = J+1,J+2,\cdots,D$, and $i = 1,2,\cdots,D$, D is the dimension of COPs. $X_i = (X_{i,1},X_{i,2},\cdots,X_{i,D})$ represents a candidate solution, $g_j(X)$ and $h_k(X)$ is jth inequality constraint and kth equality constraint respectively. Generally, equality constraint $h_k(X)$ can be transfer to two inequality constraints which can be described as $|h_k(x)| \le \varepsilon$, ε is a threshold belongs to $[10^{-5},10^{-3}]$. The goal of solving COPs is to search for the global best solution in the feasible region that satisfy the constraints.

QPSO is inspired by quantum mechanics theories and trajectory analysis of the canonical PSO undertaken by Clerc [17]. It is a probabilistic algorithm with the update equation of the particle's position very different from the canonical PSO. In QPSO, the update formula of each particle is:

$$X_{i,t+1}^j = p_{i,t}^j \pm \alpha_t \cdot |C_t^j - X_{i,t}^j| \cdot ln(1/u_{i,t}^j) \quad (2)$$

where $X_{i,t+1}^j$ represents the jth component of particle i at time $t+1$. p called learning inclination points (LIPs) or attractor point of each particle in [7]. α is the parameter to control the convergence speed of QPSO, the smaller the parameter, the faster the convergence speed. $|C_t^j - X_{i,t}^j|$ determine the bondage domain centered on LIPs, which is shrinking in the search process. $u_{i,t}^j$ is the stochastic coefficients generated using the uniform probability distribution functions in the range [0, 1].

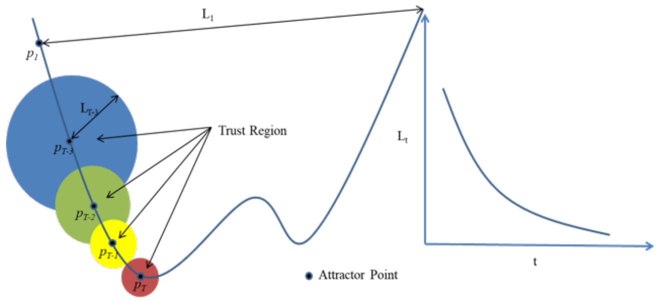

Fig. 1. The bondage domain (Trust Region) centered on LIPs (Attractor Point) are shinrking in the search process using QPSO

During the search process, QPSO algorithm enable particles to reach any position in the search space with a certain probability, and the length L ensure the position of most particles can generate in the area of the bondage domain also called trust region with a high probability as show in Fig.1. Note that the length L is declined during the search process. Therefore, the algorithm has a strong global search ability, but it also weakens the local search ability of the algorithm especially in solving EDPs. The reason is that the range of global or local optima in the end of the search is small, particles is easier to skip the global or local optima due to the large range of random values. Therefore, Gaussian probability distribution is introduced to control the bondage domain centered on LIPs. At the same time, it should be noted that the variance of Gaussian probability distribution is declined in order to shrink the bondage domain to enhance its local search ability during the search process. The update formula of adaptive gaussian quantum-behaved particle swarm optimization (AG-QPSO) is:

$$\begin{cases} X_{i,t+1}^j = p_{i,t}^j + 0.5 \cdot L_t^j \cdot ln(N(0,\sigma_t)), if\ k > 0.5 \\ X_{i,t+1}^j = p_{i,t}^j - 0.5 \cdot L_t^j \cdot ln(N(0,\sigma_t)), otherwise \end{cases} \quad (3)$$

$$\begin{cases} L_t^j = 2 \cdot \alpha_t \cdot |C_t^j - X_{i,t}^j|, \\ C_t^j = \sum_1^M P_{i,t}^j \\ p_{i,t}^j = \varphi_{i,t}^j P_{i,t}^j + (1-\varphi_{i,t}^j) G_t^j \end{cases} \quad (4)$$

where k and φ_1 are random values generated using the uniform probability distribution functions in the range [0, 1].

$$\sigma_t = \sigma_1 - (\sigma_1 - \sigma_0) \cdot t/T \quad (5)$$

$$\alpha_t = \alpha_1 - (\alpha_1 - \alpha_0) \cdot (T-t)/T \quad (6)$$

The parameter σ and α is declined linear in the search process. T is the maximum number of fitness evaluation. C is the current mean personal best position calculated by (6).

The difficulty in solving constrained optimization problems using swarm intelligence algorithms such as particle swarm optimization is to effectively deal with constraint conditions. Generally speaking, these processing methods can be divided into four types [18]:

1) Retain the solutions that meet the constraints, that is, only retain the solutions in the feasible region every iteration.

2) Repair the solutions that do not meet the constraint conditions.

3) Penalty function method. The penalty function method is the most commonly used method to deal with such problems, and is generally divided into static penalty functions and dynamic penalty functions.

4) Hybrid algorithm. Combining evolutionary algorithms together to solve constrained optimization problems.

Essentially, the first method retains eligible particles every iteration is an exhaustive method, which has high time complexity and wastes computing resources. The second method and third method have strong local search capabilities. However, many parameters are added using hybrid strategy to repairing individuals who do not meet the conditions. It is difficult to adjust the parameters and the applicability of hybrid algorithms are not well. Therefore, this paper uses the penalty function method to deal with the constrains.

Generally, the penalty function method can be described as:

$$V = \begin{cases} f(X) & if\ g_i(X) \le 0 \\ f(X) + r \cdot q \sum_n^{i=1} g_i(X) & if\ g_i(X) > 0 \end{cases} \quad (7)$$

where r is a positive rational number to control penalties, q is the number that candidate solution X don't satisfy the constrains. $f(X)$ is the objection function, the evaluation value equals to fitness value $f(X)$ if X satisfies all the constraints $g(X)$. Otherwise, the evaluation value equals to fitness value plus penalty value.

The flowchart of AG-QPSO algorithm to deal with EDPs is shown in Fig.2.

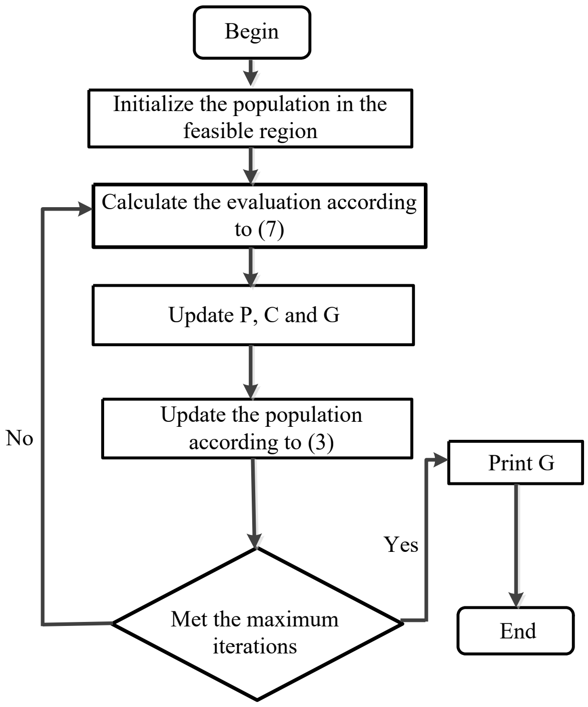
Begin
Initialize the population in the feasible region
Calculate the evaluation according to (7)
Update P, C and G
Update the population according to (3)
No
Yes
Met the maximum iterations
Print G
End

Fig. 2. The Flow chart of AG-QPSO based on penalty function method

III. EDPs Examples

In this section, two EDPs are introduced briefly, namely, the pressure vessel design problem and the tension string design problem.

A. Pressure vessel design problem

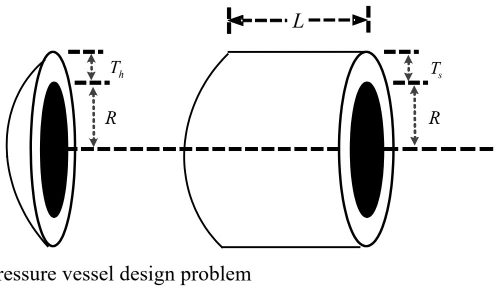

Fig. 3. Pressure vessel design problem

The goal of pressure vessel design problem is to minimize the sum of material cost, fabrication cost, and welding cost as shown in Fig.3.

The object function of this problem is determined by four decision variables, namely the thickness of the pressure vessel T_s, the thickness of the head T_h, the radius of the container R, and the length of the container without the head L.

Therefore, the optimization problem of the pressure vessel design problems $Y = [T_s, T_h, R, S] = [X_1, X_2, X_3, X_4]$ can be described as:

$$Minf(X) = 0.6224X_1X_3X_4 + 1.7781X_2X_3^2 + 3.1661X_1^2X_4 + 19.84X_1^2X_3 \tag{8}$$

subject to

$$g_1(X) = -X_1 + 0.0193X_3 \leq 0 \tag{9}$$

$$g_2(X) = -X_2 + 0.0954X_3 \leq 0 \tag{10}$$

$$g_3(X) = -\pi X_3^2X_4 - \frac{4}{3}\pi X_3^3 + 1296000 \leq 0 \tag{11}$$

$$g_4(X) = X_4 - 240 \leq 0 \tag{12}$$

The range of these four decision variables are:

$$\begin{cases} 1 \times 0.0625 \leq X_1, X_2 \leq 99 \times 0.0625 \\ 10 \leq X_{3,} X_4 20 \end{cases} \tag{13}$$

B. Tension string design problem

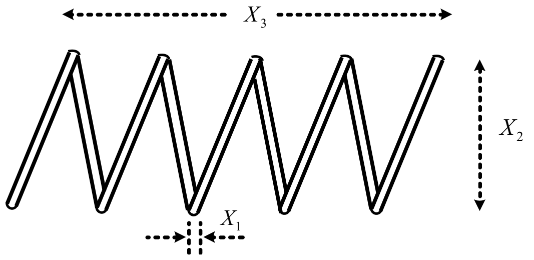

Fig. 4. Tension string design problem

The goal of the tension string design problem is to minimize the weight of the string, as shown in Fig.4. X_1, X_2 and X_3 represents the diameter of tension string, the width of tension string and the number of tension string respectively. The object function of tension string design problem is:

$$minf(X) = (X_3 + 2)X_2X_1^2 \tag{14}$$

Subject to

$$g_1(X) = 1 - \frac{X_2^3X_3}{71785X_1^4} \leq 0 \tag{15}$$

$$g_2(X) = \frac{4X_2^2 - X_1X_2}{12566(X_2X_1^3 - X_1^4)} + \frac{1}{5108X_1^2} - 1 \leq 0 \tag{16}$$

$$g_3(X) = 1 - \frac{140.45X_1}{X_2^2X_3} \leq 0 \tag{17}$$

$$g_4(X) = \frac{X_1 + X_2}{1.5} - 1 \leq 0 \tag{18}$$

$$\begin{cases} 0.05 \le X_1 \le 2 \\ 0.25 \le X_2 \le 1.3 \\ 2 \le X_3 \le 15 \end{cases} \quad (19)$$

IV. Experiment Results and Analysis

In order to verify the performance of AG-QPSO, all the results are obtain by conducting 50 runs independently. The experiment was run in a single thread in a python 2.7 environment. The configuration of CPU of the computer we used in the experiment is1.8GHz intel core i5, and RAM is 4G 1600 MHz DDR3.

A. Parameters setting

The population of the particle swarm is 80, the maximum iteration T is 1000, the initial and end variance of gaussian distribution α_1 and α_0 is 5 and 0.001 respectively. The initial and end value of expand-shrink parameter σ_1 and σ_0 is 1.0 and 0.5. The positive rational number r is 5000 in our experiment.

TABLE 3 COMPARISON BETWEEN AG-QPSO AND EXISTING ALGORITHMS

Variable	Sandgren [8]	Zhang [9]	Cao [10]	Deb [11]	Coello [12]	G-QPSO [5]	AG-QPSO
X_1	1.125	1.125	1	0.9375	0.8125	0.8125	0.841934
X_2	0.625	0.625	0.625	0.5	0.4375	0.4375	0.416171
X_3	48.3807	58.29	51.1958	48.329	40.3239	42.0984	43.62348
X_4	11.7449	43.693	90.7821	112.679	200	176.6372	158.6130
g_1	-0.1913	-0.025	-0.0119	-0.00475	-0.0034324	-8.7999E-07	-1.0482E-06
g_2	-0.1634	-0.0689	-0.1366	-0.038941	-0.052847	-3.5881E-02	-3.9889E-06
g_3	-75.875	6.5496	-13584.5631	-3652.876838	-27.105845	-0.2179	-0.3381269
g_4	-128.619	-196.307	-149.2179	-127.321	-40	-63.3628	-81.3869307
f(X)	8048.619	7197.7	7108.616	6370.7035	6288.7445	6059.7208	**5885.33277**

TABLE 6 COMPARISON BETWEEN AG-QPSO AND EXISTING ALGORITHMS IN TENSION STRING DESIGN PROBLEMS

Variable	Arora [13]	Belengundu [14]	Coello [12]	Ray [15]	Ray [16]	G-QPSO [5]	AG-QPSO
X_1	0.053396	0.050000	0.050417	0.050417	0.0521602	0.051515	0.05
X_2	0.315900	0.3159000	0.321532	0.321532	0.3681587	0.352529	0.348913
X_3	9.185400	14.250000	13.979915	13.979915	10.648442	11.538862	10.56234
g_1	0.000019	-1.2672E-03	-3.33E-03	-1.925E-03	-7.4527E-09	-4.8341E-05	-2.37389E-06
g_2	-0.000018	-0.1490	-0.1357	-0.1556	-0.1314	-3.5774E-05	-2.04849E-06
g_3	-4.123842	-3.9383	-4.0263	-3.8994	-4.0758	-4.0455	-6.08278E+02
g_4	-0.698283	-0.7561	-0.7312	-0.7520	-0.7198	-0.73064	-1.1992758
f(X)	0.012730	-0.01273027	0.01270	0.0130602	0.01266	0.012665	**0.01095788**

B. Results on pressure vessel design problem

TABLE 1 COMPARISON OF AG-QPSO, PSO, QPSO AND GQPSO ALGORITHMS ON PRESSURE VESSEL PROBLEM

Algorithm	Worst	Best	Mean	Median	Std.
PSO	14076.32	6693.72	8756.68	8424.48	1492.56
QPSO	8017.28	6059.72	6839.93	6818.23	479.26
G-QPSO	7544.49	6059.72	6440.37	6257.59	448.47
AG-QPSO	6003.52	5885.33	**5890.93**	5885.53	**350.26**

TABLE 2 THE BEST SOLUTION OF PRESSURE VESSEL PROBLEM USING AG-QPSO

Variable	Parameter	AG-QPSO
X_1	Ts	0.84
X_2	Th	0.41
X_3	R	43.62
X_4	L	158.61
g_1	Eq.(9)	-1.04E-06
g_2	Eq.(10)	-3.98E-06
g_3	Eq.(11)	-0.33
g_4	Eq.(12)	-81.38
f(X)	Eq.(8)	5885.33

We first have compared AG-QPSO algorithm with PSO, QPSO [7], G-QPSO [5] on pressure vessel design problems. It is clearly that AG-QPSO not only obtain best mean value, but it is more robust according the Std. value, as show in Table 1. Simulation results had shown that for the first EDPs, the QPSO, G-QPSO and the proposed AG-QPSO algorithms perform satisfactorily. The results in Table 1shows that the classical PSO was outperformed by QPSO, G-QPSO and AG-QPSO approaches for the first example. Table 1 with best results in bold font. The best solutions found by AG-QPSO approaches present smaller standard deviation and mean values than the results obtained by PSO, QPSO and G-QPSO. The best mean from the 50 runs performed 5885.33. Table 2 shows the best solution that AG-QPSO had found on pressure vessel problem, and the value of constraints obtained under the best solution.

Subsequently, we compared the proposed algorithm with some existing algorithms, including Sandgren [8], who used a branch-and-bound approach, Kannan and Krame [9], who used an augmented Lagrangian Multiplier approach, Deb [10], using Genetic Adaptive Search, Cao and Wu [11], who employed an approach of improved evolutionary programming, Cao and Wu [12], who used cellular automata based on a genetic algorithm, Lin, Wang, and Hwang [13], who proposed a hybrid differential evolution method, Hu, Eberhart, and Shi [19], using particle swarm optimization, and Schmidt and Thierauf [20], who applied a threshold accepting algorithm with differential evolution to solve the pressure vessel problem.

In conclusion, we can see that the best solution we find satisfy all the constraints. In addition, the value of constraints is very small, which proved that AG-QPSO has a strong capacity of local search ability.

C. *Results on tension string design problem*

Tension string design problem was solved previously by Belengundu [13], using eight optimization approaches. Arora [14], Coello [12], Ray and Saini [15] and Ray and Liew [16] also solved this problem using other optimization methods.

The results of PSO, QPSO, G-QPSO and AG-QPSO approaches are presented in Table 4. Table 4 with best results in bold font. As can be seen, the best mean and median values from the 50 runs performed was using AG-QPSO with *f(X)* = 0.01095.

Therefore, we still find that AG-QPSO not only has obtained better result compare to the canonical PSO, the classical QPSO and gaussian QPSO from Table 4, but also performs better than existing algorithms that design for tension string design problem from Table 6.

TABLE 4 COMPARISON OF AG-QPSO, PSO, QPSO AND GQPSO ALGORITHMS IN TENSION STRING DESIGN PROBLEM

Algorithm	**Worst**	**Best**	**Mean**	**Median**	**Std.**
PSO	0.071	0.012	0.020	0.013	0.012
QPSO	0.018	0.013	0.013	0.014	0.001
G-QPSO	0.016	0.013	0.013	0.013	0.0013
AG-QPSO	0.011	0.011	**0.011**	0.011	7.6E-17

TABLE 5 THE BEST SOLUTION OBTAINED BY AG-QPSO IN TENSION STRING DESIGN PROBLEM

Variable	**Parameter**	**AG-QPSO**
X_1	d	0.050
X_2	D	0.348
X_3	N	10.562
g_1	Eq.(15)	-2.37E-06
g_2	Eq.(16)	-2.04E-06
g_3	Eq.(17)	-6.08E+02
g_4	Eq.(18)	-1.199
f(X)	Eq.(14)	0.011

D. *Time complexity analysis*

TABLE 7 TIME COMPLEXITY ANALYSIS OF AG-QPSO

Algorithm	**Time complexity**			
	Initialization	**Evaluation**	**Position Update**	**Overall**
PSO	O(MD)	O(MD)	O(MD)	O(MD)
QPSO	O(MD)	O(MD)	O(MD)	O(MD)
GQPSO	O(MD)	O(MD)	O(MD)	O(MD)
Hybrid Algorithms	O(MD)	O(MD)	O(MD)	O(MD)
AG-QPSO	O(MD)	O(MD)	O(MD)	O(MD

It should be noted that the best solution obtained by AG-QPSO satisfies all the constrains. In Table 7, *M* represents the population the particle swarm and *D* is the dimension of the optimization problem. In this paper, the analysis of evaluating the time complexity of the optimization algorithm is divided into four aspects, namely the time complexity of the initialization, evaluation, particle position update, and the overall time complexity. It is obvious that the time complexity of AG-QPSO algorithm is the same as PSO, QPSO algorithm, and GQPSO algorithm.

V. CONCLUSION

Through controlling the bondage domain centering on LIPs in QPSO, we propose AG-QPSO to solve EDPs. As the bondage domain shrinks, AG-QPSO can stably migrate from a strong global search capability at the beginning of the search to a strong local search capability at the end. We found that the performance of the AG-QPSO algorithm was better than other existing algorithms, according to the results obtained on the pressure vessel design and the tension string design problem. In addition, controlling the Gaussian distribution to generate Gaussian sequence instead of random sequence with uniform distribution in QPSO had proved is a powerful strategy to enhance the ability of QPSO in preventing premature convergence to local optima.

The aim of future works will focus on other real world optimization problems, such as electronic power system and control system. Furthermore, other relevant studies can be realized: such as: (i) use better LIPs to attractor particles (ii) design a better length *L* to enhance the search performance of QPSO.

REFERENCES

[1] He Q, Wang L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems [J]. Engineering Applications of Artificial Intelligence, 2007, 20(1):89-99.

[2] Khamsawang S, Wannakarn P, Jiriwibhakorn S. Hybrid PSO-DE for solving the economic dispatch problem with generator constraints [C]. Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on. Los Alamitos: IEEE Computer Society Press, 2010. 804-813.

[3] Garg, Harish. A hybrid PSO-GA algorithm for constrained optimization problems [J]. Applied Mathematics and Computation, 2016, 274:292-305.

[4] Guedria N B. Improved Accelerated PSO Algorithm for Mechanical Engineering Optimization Problems [J]. Applied Soft Computing, 2015, 2016(40):455–467.

[5] Coelho L D S. Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems [J]. Expert Systems with Applications, 2010, 37(2):1676-1683.

[6] Ray T, Saini P. Engineering design optimization using a swarm with an intelligence information sharing among individuals [J]. Engineering Optimization, 2001, 33(6): 735-748.

[7] Sun J. Quantum-Behaved Particle Swarm Optimization: Analysis of Individual Particle Behavior and Parameter Selection [J]. Evolutionary Computation, 2012, 20(3):349-393.

[8] Sandgren, E. Nonlinear integer and discrete programming in mechanicaldesign[C]: In Proceedings of the ASME design technology conference,,Kissimine, FL, USA, 1989, 95–105.

[9] Zhang, C., & Wang, H. P. (1993). Mixed-discrete nonlinear optimization withsimulated annealing[J]. Engineering Optimization, 17(3), 263–280.

[10] Cao, Y. J., & Wu, Q. H.. Mechanical design optimization. In IEEE conference on evolutionary computation[C]. Indianapolis, USA, 1997: 443–446.

[11] Deb, K, GeneAS: A robust optimal design technique for mechanical component design[J]. Evolutionary algorithms in engineering application. Berlin: Springer-Verlag, 1997: 497–514.

[12] Coello, C. A. C. Use of a self-adaptive penalty approach for engineering optimization problem[J]. Computers in Industry,2000,4(2): 113–127.

[13] Arora, J. S. Introduction to optimum design[M]. New York, NY, USA: MGraw-Hill, 1989.

[14] Belengundu, A. D. A study of mathematical programming methods forstructural optimization[M]. Iowa, USA: Department of Civil and Environmental Engineering,University of Iowa, 1982.

[15] Ray, T., & Saini, P. Engineering design optimization using a swarm with an intelligent information sharing among individuals[J]. Engineering Optimization, 2001, 33(3):35–748.

[16] Ray, T., & Liew, K. M. Society and civilization: An optimization algorithmbased on the simulation of social behavior[J]. IEEE Transactions on Evolutionary Computation, 2000, 7(4): 386–396.

[17] Clerc M, Kennedy J. The particle swarm - explosion, stability, and convergence in a multidimensional complex space[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(1):58-73.

[18] Michalewicz Z. Evolutionary algorithms for constrained parameter optimization problems [J]. Evolutionary Computation, 1996, 4(1):1-32.

[19] Hu, X., Eberhart, R. C., & Shi, Y. Engineering optimization with particle swarm[C]. In IEEE conference on swarm intelligence, Indianapolis, USA, 2003.

[20] Schmidt, H., & Thierauf, G. A combined heuristic optimization technique[J]. Advances in Engineering Software, 2005: 36(1), 11–19.

[21] Clerc, M.: Stagnation analysis in particle swarm optimisation or what happens when nothing happens[J], 2006.

[22] Langdon W B, Poli R: Evolving problems to learn about particle swarm and other optimisers[C]. 2005 IEEE Congress on Evolutionary Computation, 2005: 81–88.

[23] Ling S H, Iu H H C, Leung F H F, Chan K Y: Improved hybrid particle swarm optimized wavelet neural network for modeling the development of fluid dispensing for electronic packaging[J]. IEEE transactions on industrial electronics, 2008, 55(9): 3447–3460.

[24] Dos S C L., Herrere B M: Fuzzy identification based on a chaotic particle swarm optimization approach applied to a nonlinear yo-yo motion system[J]. IEEE Transactions on Industrial Electronics. 2007, 54(6): 3234–3245.

[25] Kennedy, J.: Swarm intelligence. In: Handbook of nature-inspired and innovative computing[M], Springer, 2006: 187–219.

[26] Glerc, M.: Initialisations for particle swarm optimization. Online at http://clerc, maurice, free. fr/pso, 2008.

[27] Del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J.C., Harley, R.G.: Particle swarm optimization: basic concepts, variants and applications in power systems[J]. IEEE Transactions on evolutionary computation 2008, 12(2): 171–195.

[28] Fan, H.: Study on vmax of particle swarm optimization[C]. In Proceeding of the Workshop on Particle Swarm Optimization, 2001.

[29] Kennedy, J., Mendes, R.: Population structure and particle swarm performance[C]. In: Proceedings of the 2002 Congress on Evolutionary Computation. 2002, 2:1671–1676.

[30] Panda, A., Mallipeddi, R., Das, S.: Particle swarm optimization with a modified learning strategy and blending crossover[C]. IEEE Symposium Series on Computational Intelligence. 2017: 1–8.