Solving Constrained Optimization via a Modified Genetic Particle Swarm Optimization

Liu Zhiming[1,2], Wang Cheng[1], Li Jian[1]

1.Hubei Key Laboratory of Digital Valley Science and Technology
Huazhong University of Science & Technology
2.Hubei University of Education
Wuhan, Hubei 430074, China

Leejan4ever@gmail.com

***Abstract* – The genetic particle swarm optimization (GPSO) was derived from the original particle swarm optimization (PSO), which is incorporated with the genetic reproduction mechanisms, namely crossover and mutation. Based on which a modified genetic particle swarm optimization (MGPSO) was introduced to solve constrained optimization problems. In which the differential evolution (DE) was incorporated into GPSO to enhance search performance. At each generation GPSO and DE generated a position for each particle, respectively, and the better one was accepted to be a new position for the particle. To compare and ranking the particles, the lexicographic order ranking was introduced. Moreover, DE was incorporated to the original PSO with the same method, which was used to be compared with MGSPO. MGPSO were experimented with well-known benchmark functions. By comparison with original PSO algorithms and the evolution strategy, the simulation results have shown its robust and consistent effectiveness.**

Index Terms - Particle swarm optimization, genetic algorithm, constrained optimization.

I. INTRODUCTION

Constrained optimization problems are the problems to minimize the object function under given constraints, which can be formulated as solving the object function as follows.

Minimize $f(X), X = (x_1, \cdots, x_D) \in R^D$

Subject to: $g_j(X) \le 0, (j = 1, \cdots, q)$

$$x^d_{\min} \le x^d \le x^d_{\max}, 1 \le d \le D$$

Where the search space (S) is defined by

$$S = \{X \in R^D \mid x^d_{\min} \le x^d \le x^d_{\max}, i \in \{1, \cdots, D\}\}$$

and the feasible space (F) is defined by

$$F = \{X \in R^D \mid g_j(X) \le 0 (\forall j \in \{1, \cdots, q\})\}$$

There exist many studies on solving constrained optimization problems with particle swarm optimization [1], [2], [3], [4], [5]. Where the original PSO was introduced, in which the position of each particle is a set of real value variables. A genetic particle swarm optimization was proposed in [6], where the position is a set of binary coding, and genetic reproduction mechanisms, namely crossover and mutation are incorporated into PSO. This paper introduced GPSO to solve constrained optimizations, and to enhance the search performance a differential evolution (DE) algorithm was employed. In which GPSO and DE competed and cooperated during the evolution.

To compare and rank particles, the constraints were treated in different ways. The penalty function method was adopted in [2], [3], where an extended object function was defined by adding the constraint violation to the object function as a penalty. In [4], [5] the object function and the constraint violation were used separately, where both the object function and the constraint violation were optimized by a lexicographic order in which the constraint violation precedes the object function. Which can be described as follows 1) give two feasible particles, pick the one with better fitness value; 2) if both particles are infeasible, pick the particle with lower constraint violation; 3) given a pair of feasible and infeasible particles, pick the feasible one. The stochastic ranking technique was proposed [7], which was firstly employed in the evolution strategy (ES). Our approach employed the lexicographic order, which is easy to implement.

Section II introduced the original PSO and the GPSO with the differential evolution mutation. The simulation results and discussion were presented in Section III. The conclusions of this paper were given in section IV.

II. PARTICLE SWARM OPTIMIZATION WITH DIFFERENTIAL EVOLUTION MUTATION

The particle swarm optimization consists of a swarm of particle flying through the search space. Each particle i in the swarm contains parameters for position (X_i^t) and velocity (V_i^t) where $X_i^t \in R^D$, $V_i^t \in R^D$ ($t \in N, i \in N$), and D is the dimension of the search space, t is the number of current iteration. The position of each particle represents a potential solution to the optimization problem. At each iteration step, each particle adjusts its velocity based on its momentum and the influence of its best position (P_i^t) tracked and the best position found by the neighbour (P_G^t) so far, then computes a new position to examine. The PSO formulas are given as follows.

$$V_i^t = \omega \cdot V_i^{t-1} + c_1 \cdot rand_1() \cdot (P_i^{t-1} - X_i^{t-1}) + c_2 \cdot rand_2() \cdot (P_G^{t-1} - X_i^{t-1})$$

$$X_i^t = X_i^{t-1} + V_i^t$$

Where c_1 and c_2 are constants known as the cognitive and social acceleration coefficients, ω is the inertia weight

and set within [1, 0] $rand_1()$ and $rand_2()$ are random numbers uniformly distributed between 0 and 1. [8], [9], [10],

There are four neighbourhood topology types were proposed in [11] viz., circles ($lbest$), wheels, starts ($gbest$) and random edges, and our approach adopts the circle topology. In which each particle communicates with its k immediate neighbours. In this paper, $k=2$ a particle communicates with its immediately adjacent neighbours [5].

Yin [6] proposed a genetic PSO with genetic reproduction mechanisms. Based on which, each component of the position is defined as follows:

$$B_{i,d}^t = (X_{i,d}^t - x_{\min,d})/\Delta_d \quad (1)$$

$$\Delta_d = (x_{\max,d} - x_{\min,d})/(2\,\hat{}\,u - 1) \quad (2)$$

Where d is the number of each component of position, u is the index of the binary bit, $B_{i,d}^t$ is the binary position of i^{th} particle. And then our approach employed crossover and mutation operators to generate new positions as follows:

$$B_{i,d}^{u,t+1} = \omega(0,\omega_1).rand(B_{i,d}^{u,t}) + \omega(\omega_1,\omega_2).rand(BP_{i,d}^{u,t}) + \omega(\omega_2,1).rand(BP_{G,d}^{u}) \quad (3)$$

Where BP_i^t and BP_G^t is the binary individual best position and binary neighbour best position, respectively, calculated with (1) and (2). $\omega()$ and $rand()$ are the threshold function and the probabilistic bit flipping function, respectively, and they are defined as follows.

$$\omega(a,b) = \begin{cases} 1(if: a \le r_1 < b) \\ 0, otherwise \end{cases}$$

$$rand(y) = \begin{cases} 1-y, if: (r_2 < p_m) \\ y, otherwise \end{cases}$$

Where r_1 and r_2 are the random numbers uniformly distributed in [0, 1], thus with $0<\omega_1<\omega_2<1$, only one of the four terms on right hand side of (1) will remain, and $rand()$ mutates the binary bit y with a small mutation probability pm. And then ω_1 and ω_2 are used to select bits from B_i^t, BP_i^t and BP_G^t. They work similarly to what c_1 and c_2 do in the original PSO.

In GPSO the position generated by with combinatorial operators, and to enhance the search performance a differential evolution was employed [5], [11]. The differential evolution equation is given as follows.

$$X_{i,d}^{t+1} = X_{r1,d}^t + r \times (X_{r2,d}^t - X_{r3,d}^t) \quad (4)$$

Where $r1$, $r2$ and $r3$ are random integral numbers between 1 and the total number of the population of the swarm, d is the number of each position vector， r is a coefficient set to 0.5.

At t^{th} generation, based on the position of $(t-1)^{th}$ generation, the GPSO and DE will generate a new binary and a real value position for each particle, respectively. And the binary position will be converted to real value position to calculate object function value and constraint violations. And then with the lexicographic order ranking the better position will be accepted as the position of the particle. If GPSO wins, the binary will be converted into real value position for DE in the next generation, and if DE wins, the real value position will be converted into binary position for GPSO. The flowchart of the proposed algorithm is given as Fig.2.

The DE was employed in [5] too, and there are some differences. Firstly, in our approach, the GPSO and DE generated new positions based on the original position of the last generation, respectively. But in PESO, DE generated a position based on the position generated by PSO in the same generation. Secondly, in PESO the new position was used to update the individual best position. Finally, the authors employed the random mutation too, which was absent in our approach. To validate the effectiveness of GPSO, a modified PSO (MPSO) was also employed, which was similar to MGPSO, where the original PSO took the place of GPSO in Fig.1.

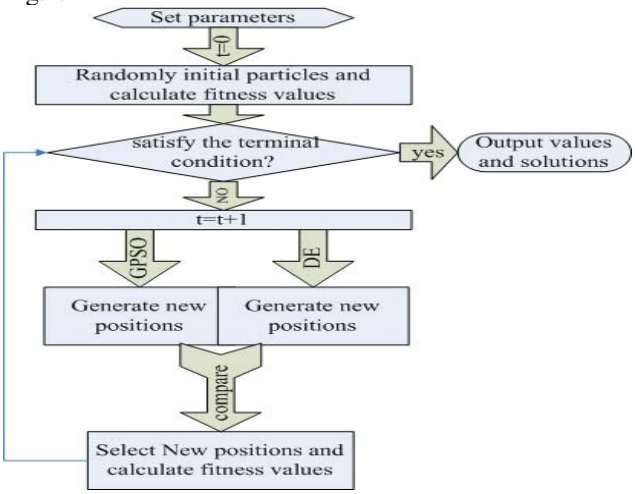

Fig. 1 Flowchart of MGPSO.

III. SIMULATION RESULTS

Nine benchmark functions were selected from [7]. G02 and G08 are maximization problems, which were transformed to minimization problems using $-f(x)$. The summary of the functions are given in table I.

For each benchmark problems, 30 independent runs were performed with $\omega_1 = 0.1$ and $\omega_2 = 0.55$, $ratio$ deceased from 1E-2 to 1E-4 during the generation linearly (1E-1 and 1E-3 for G06). Each component of position was converted to a 32 bit binary chain. All runs were terminated after 2000 generations with 30 particles; hence there were 30*2*2000=120,000 evaluations in each run in our approach. The simulation results are given in table II. These include the best, median, mean, and worst objective value found and the standard deviation. All equality constraints have been converted to inequality constraints as $|h(x)|-\delta \le 0$

with $\delta = 1E-4$. As a result of the approximation, some results might be better than the optimum.

To compare the original PSO and the genetic PSO, the differential evolution was incorporated into original PSO too, with $c_1 = c_2 = 0.5$, ω decreased linearly from 0.9 to 0.4 and the results of MGPSO and MPSO are given in table II. And the results of ES [7] and PESO [5] are list in table III and table IV.

TABLE I

SUMMARY OF MAIN PROPERTIES OF THE BENCHMARK PROBLEMS (NE: NONLINEAR EQUALITY, NI: NONLINEAR INEQUALITY, LI: LINEAR INEQUALITY, α : THE NUMBER OF ACTIVE CONSTRAINTS AT OPTIMUM, D: THE DIMENSION OF THE FUNCTION)[1]

fun	Optimal	D	type	LI	NE	NI	α	$\lvert F\rvert/\lvert S\rvert$
G01	-15.000	13	Q	9	0	0	6	0.0011%
G02	-0.803619	20	N	0	0	2	1	99.847%
G04	-30665.539	5	Q	0	0	6	2	52.123%
G05	5126.498	4	C	2	3	0	3	<0.00001%
G06	-6961.814	2	C	0	0	2	2	0.0066%
G07	24.306	10	Q	3	0	5	6	0.000%
G08	-0.095825	2	N	0	0	2	0	0.8250%
G09	680.630	7	P	0	0	4	2	0.5121%
G10	7049.248	8	L	3	0	3	6	0.0010%

TABLE II
RESULTS OF MGPSO AND MPSO

	fun	Optimal	best	median	mean	st.dev	worst
MGPSO	G01	-15.000	-15.000	-15.000	-15.000	0E+00	-15.000
	G02	-0.803619	-0.803397	-0.796035	-0.794282	7.18E-03	-0.771180
	G04	-30665.539	-30665.539	-30665.539	-30665.539	6.07E-07	-30665.539
	G05	5126.498	5126.812	5173.040	5271.045	2.13E+02	5795.915
	G06	-6961.814	-6961.814	-6961.814	-6961.814	1.06E-12	-6961.814
	G07	24.306	24.307	24.531	24.557	6.08E-02	25.123
	G08	-0.095825	-0.095825	-0.095825	-0.095825	4.23E-17	-0.095825
	G09	680.630	680.630	680.635	680.647	5.51E-02	680.667
	G10	7049. 248	7049.615	7223.581	7310.018	3.23E-01	7562.282
	fun	Optimal	best	median	mean	st.dev	worst
MPSO	G01	-15.000	-15.000	-15.000	-13.733	1.68E+00	-9.000
	G02	-0.803619	-0.803397	-0.789797	-0.785069	1.28E-02	-0.738522
	G04	-30665.539	-30665.539	-30665.539	-30665.539	1.00E-15	-30665.539
	G05	5126.498	5126.497	5143.090	5248.758	1.99E-01	5947.0144
	G06	-6961.814	-6961.814	-6961.814	-6961.814	2.00E-15	-6961.814
	G07	24.306	24.407	24.533	24.758	6.14E-01	27.528
	G08	-0.095825	-0.095825	-0.095825	-0.095825	0.00E+00	-0.095825
	G09	680.630	680.630	680.633	680.635	7.06E-05	680.656
	G10	7049. 248	7049.248	7250.967	7651.232	1.04E-01	11100.00

TABLE III
RESULTS OF ES WITH STOCHASTIC RANKING [7] with $P_f = 0.45$

fun	Optimal	best	median	mean	st.dev	worst
G01	-15.000	-15.000	-15.000	-15.000	0.0E+00	-15.000
G02	-0.803619	-0.803515	-0.785800	-0.781975	2.0E-02	-0.726288
G04	-30665.539	-30665.539	-30665.539	-30665.539	2.0E-05	-30665.539
G05	5126.498	5126.497	5127.372	5128.881	3.5E+00	5142.472
G06	-6961.814	-6961.814	-6961.814	-6875.940	1.6E+02	-6350.262
G07	24.306	24.307	24.357	24.374	6.6E-02	24.642
G08	-0.095825	-0.095825	-0.095825	-0.095825	2.6E-17	-0.095825
G09	680.630	680.630	680.641	680.656	3.4E-02	680.763
G10	7049. 248	7054.316	7372.613	7559.192	5.3E+02	8835.655

TABLE IV
RESULTS OF PESO

fun	Optimal	best	median	mean	st.dev	worst
G01	-15.000	-15.0000	-15.0000	-15.0000	0E+00	-15.0000
G02	-0.803619	-0.792608	-0.731693	-0.721749	5.35E-02	-0.614135
G04	-30665.539	-30665.538672	-30665.538672	-30665.538672	0E+00	-30665.538672
G05	5126.498	5126.484154	5126.538302	5129.178298	5.1575E+00	5148.859414
G06	-6961.814	-6981.813876	-6981.813876	-6981.813876	0E+00	-6981.813876
G07	24.306	24.306921	24.371253	24.371253	6.96E-02	24.593504
G08	-0.095825	-0.095825	-0.095825	-0.095825	0E+00	-0.095825
G09	680.630	680.630057	680.630057	680.630057	2.617E-07	680.630058
G10	7049. 248	7049.459452	7069.926219	7099.101385	5.9329E+1	7251.396244

It can be observed from table I that MGPSO performed much better than MPSO for most functions. For G01, MGPSO consistently found the optimal solution in all the runs, while MPSO failed in some runs. For G02, the median, mean and worst values of MGPSO all are better than those of MPSO, and the best solution is equal. For G04, G06 and G08, both the algorithms consistently found the optimal solutions in all runs and the standard deviation of MPSO is a little better. For G05, MPSO outperformed MGPSO, except the worst value. For G07, MGPSO outperformed MPSO in all the terms, except the median value. For G09, MGPSO and MPSO found the optimal solution, and the other values are comparable. For G10,

MPSO found better solution (optimal) than MGPSO, while the median, mean and worst values are worse. Based on the comparison above, it can be concluded that with the incorporation of differential evolution, GPSO outperformed original PSO for most functions.

In comparison with the results of ES and PESO in table II and table III, the results of MGPSO are better for some functions. For G01, G04 and G08, the entire three algorithms found the optimal solution in all runs. For G02, MGPSO outperformed ES and PESO in the all terms, except the best value which is worse than ES but comparable. For G05 and G07, the results of MGPSO are the worst but comparable. For G06, both MGPSO and PESO found the optimal solution in all runs which are better than ES. For G09, all the algorithms found the optimal solution, and MGPSO outperformed ES in the terms of median, mean and worst values. And PESO performed the best. For G10, PESO outperformed the others while MGPSO outperformed ES in all the terms. It seems that MGPSO performed well for most functions, which outperformed ES for G02, G06, G09 and G10, and outperformed PESO for G02, and the others are comparable. PESO outperformed MGPSO for some function, while there were 350,000 evaluations in each run in PESO [5], which is almost 3 times of those in our approach (120,000).

IV. CONCLUSIONS

This paper introduced a modified genetic particle swarm optimization (MGPSO), which employed the genetic particle swarm optimization (GPSO) to solve constrained optimization problems, which was incorporated with the genetic reproduction mechanisms, namely crosser and mutation. Moreover a competition operator was employed, where a differential evolution (DE) was incorporated. At each generation, GPSO and DE generated a new position for a particle, respectively, to be compared with lexicographic order ranking. And only the winner could survive and be accepted as the new position of the particle. To compare the effectiveness of GPSO and original PSO, an original PSO edition competition PSO (MPSO) was performed where the original PSO has token the place of GPSO in MGPSO.

Both MGPSO and MPSO were tested on a set of 9 benchmark problems. Experimental results have been presented. And both the algorithms found high quality values for most problems, and MGPSO outperformed the MPSO. Moreover, By comparison with ES and PESO, MGPSO provided better or comparable solutions for most problems**,** which have shown that GPSO, a combinatorial edition PSO is effective to solve constrained optimization problems. The future work will include other constrained problems especially the constrained integer programming.

REFERENCES

[1] X. Hu, Eberhart. R, "Solving constrained nonlinear optimization problems with particle swarm optimization," In: Proceedings of the Sixth World Mutilconference on Systemics, Cybernetics and Informatics, Orlando, Florida, (2002)

[2] K. E. Parsopoulos, M. N. Vrahatis, "Particle swarm optimization method for constrained optimization problems," In: Intelligent Technologies-Theory and Application: New Trends in Intelligent Technologies, ser. Frontiers in Artificial Intelligence and Applications. Vol. 76, (2002) 214-220

[3] C. Wang, H. Yuan, Z. Huang,et, "A Modified Particle Swarm Optimization Algorithm and Its Application in Optimal Power Flow Problem," In: Proceeding of the Fourth International Conference on Machine Learning and Cybernetics,Guanghou, (2005)

[4] T. Takahama, S. Sakai, "Tuning Fuzzy Control Rules by the α constrained method which solves constrained nonlinear optimization problems," In: Electronics and Communications in Japan, Vol. 83. (2000) 1-12

[5] A. Zavala, A. Aguirre, E. Diharce, "Constrained optimization via Particle Evolutionary Swarm Optimization Algorithm (PESO)," In: GECCO, (2005) 209-216

[6] P. Y. Yin, "Genetic particle swarm optimization for polygonal approximation of digital curves," Pattern Recognition and Image Analysis., vol. 16, no. 2, pp. 223-233, Apr, 2006.

[7] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," IEEE Trans. Evol. Comput., vol. 4, no. 3, pp. 284-294, Sep. 2000

[8] Shi Y H, et al. Empirical Study of Particle Swarm Optimization[R]. Proceedings of Congress on Evolutionary Computation, 1945-1950, (1999).

[9] R. Eberhart, Y. Shi, "Particle Swarm Optimization: Developments, Applications and Resources," In: Proceedings of the 2001 Congr. Evolutionary Computation, Vol.1. (2001)

[10] A. Ratnaweera, S. Halgamuge, "Self-Organizing Hierarchical Particle Swarm Optimizer with Time-Varying Acceleration Coefficients," In: IEEE Transaction on Evolutionary Computation. Vol. 8 (2004) 240-255

[11] J. Kennedy, "Small Worlds and Mega-minds: Effects of Neighborhood Topology on Particle Swarm Performance," In: IEEE Congress on Evolutionary Computation. (1999) 1931-1938

[12] Storn. Rainer, K. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces," In: Technical Report TR-95-012, ICSI, March (1995).