New Multiobjectve PSO Algorithm for Nonlinear Constrained Programming Problems

Chun'an Liu[1,2]

[1]Department of Mathematics, Baoji University of Arts and Sciences, Baoji 721013, P. R. China
[2]Computer and Information Institute, Baoji University of Arts and Sciences, Baoji 721013, P. R. China

Abstract

A new approach is presented to solve nonlinear constrained programming problems (NLCPs) by using particle swarm algorithm(PSO). It neither uses any penalty functions, nor distinguish the feasible solutions and the infeasible solutions including swarm. The new technique treats the NLCPs as a bi-objective optimization problem, one objective is the original objective of NLCPs, and the other is the degree violation of constraints. As we prefer to keep the ratio of infeasible solutions so as to increase the diversity of swarm and avoid the defect of conventional over-penalization, a new fitness function is designed based on the second objective. In order to make the PSO escape from the local optimum easily, we also design a adaptively dynamically changing inertia weight. The numerical experiment shows that the algorithm is effective.

Keywords: Nonlinear constrained programming, Swarm algorithm, Multiobjectve optimization, Adaptively dynamically changing

1. Introduction

Nonlinear constrained programming problems (NLCPs) are encountered in numerous applications. Structural optimization, engineering design, economic, allocation and location problems are just a few of the scientific fields in which NLCPs are frequently met. The key point in the NLCPs process is how to deal with the constraints. The traditional methods usually convert the problem into non-constrained problem to solve, such as [1],[2]. However, these methods demand the high quality of the function, and they can only solve those better quality functions.

Particle swarm algorithms (PSO)[3] originally developed by Kennedy and Eberhart is a swarm based algorithm. PSO is initialized with a swarm of candidate solutions, each candidate solution is called particle and move according the velocity-location equation. PSO have been found to be fast in solving nonlinear, non-differentiable and multi-modal optimization problems. In the last years, several important effort has been reported in the literatures. Coath and Halgamuge[4] proposed the feasible solutions method(FSM) and the penalty function method(PFM) to handle constraints in PSO. however, both of the two method have disadvantages. FSM demands all particles must be include in the feasible region, and the PFM required careful fine tuning of the penalty function parameters.

Recently, some genetic algorithm based on the multiobjective optimization concepts have been proposed to handle constraints[5],[6]. Its main ideal is consider each constraints as an objective function, and transform the NLCPs into multiobjective optimization problem with $m + 1$ objectives, where m is the number of the constraints of NLCPs. Then using the Pareto dominance concepts of multiobjective optimization or Pareto ranking to select the candidates individuals and consist the next swarm. However, this method has a serious drawback according to the following two case if all objectives are considered as the same importance. one case is that an feasible solution which is seen to the true optimal solution of the NLCPs, but it has a small fitness and be seen as a bad solution to delate according to the the Pareto dominance or Pareto rank. However, this solution should survive in the next generation. The other case is that one infeasible solution which is far away from the true optimal solution of the NLCPs or the boundary of constraints, maybe its has a very big fitness value to survive according to the Pareto dominance or Pareto rank. However, this solution should keep away from the next generation.

In this paper, the constraints of NLCPs are firstly transformed into a degree violation and thus the NLCPs is transformed into a bi-objective problem. In order to increase the diversity of population and avoid the defect of conventional overpenalization, a new fitness function is designed based on the degree violation. and in order to make the PSO escape from the local optimum easily, we also design a adaptively dynamically changing inertia weight. The numerical experiment shows that the algorithm is effective.

2. Transformation of NLCPs

Consider the following nonlinear constrained programming problems(1):

$$\begin{cases} \min\limits_{x\in D\subset[L,U]} f(x) \\ s.t. \quad g_i(x)\le 0 \quad i=1\sim m \end{cases} \tag{1}$$

Where $[L,U]=\{x=(x_1,x_2,\cdots,x_n)|l_i\le x_i\le u_i, i=1\sim n\}\subset R^n$. $D=\{x|x\in[L,U], g_i(x)\le 0, i=1\sim m\}$ is called feasible region and each point is called a feasible point.

For the problem (1), it can be transformed into the following bi-objective optimization problem (2):

$$\min F(x)=(f_1(x),f_2(x)) \tag{2}$$

Where $f_1(x)$ is objective function of the original optimization problem (1), $f_2(x)=\frac{1}{m}\sum_{i=1}^m \frac{c_i(x)}{c(x)+\varepsilon}$, $c_i(x)=\max\{0,g_i(x)\}$, $i=1\sim m$, $c(x)=\max\limits_{1\le i\le m}\{c_i(x)\}$, $\varepsilon>0$. It is obvious that to minimize the first objective function of optimization problem (2) means to find a point x^* so as to obtain the optimal value of the problem (1). Since the second objective function of problem (2) is defined as the function of degree violation of constraints. Thus, to minimize $f_2(x)$ means to search the point x^* in order to meet all the constraints. Therefore, to minimize the two objectives of problem (2) simultaneously means to search for the point so as to meet all the constraints and make the first objective $f_1(x)$ minimize.

3. New multiobjective PSO algorithm

3.1. PSO algorithm

PSO was initially proposed by Kennedy and Eberhart[3]. It is a population evolution based optimization method inspired by the behavior of bird flocks, which employs a swarm of particles to probe the search space. In PSO, each individual (called particle) is described by three main concepts: its current location in the search space, a memory of its best previous location and information regarding the best location and information regarding the best location ever attained by a topological neighborhood of it. Suppose that $X_{id}(t)$ is the present position of $i-$th particle (termed as same as its present location $X_{id}(t)$) in generation t, $P_{id}(t)$ be its own best previous position, and $P_{gd}(t)$ be the best position ever attained by the swarm. Then, the particle $X_{id}(t)$ is manipulated according to the following velocity-location equations:

$$V_{id}(t+1)=\omega\cdot V_{id}(t)+c_1\cdot r_1(P_{id}(t)-X_{id}(t))+$$

$$c_2\cdot r_2(P_{gd}(t)-X_{id}(t)) \tag{3}$$

$$X_{id}(t+1)=X_{id}(t)+V_{id}(t+1), i=1\sim N \tag{4}$$

Where $V_{id}(t)$ is the previous velocity, $V_{id}(t+1)$ is the present velocity, $X_{id}(t+1)$ is the new position, c_1,c_2 are realizations of uniformly distributed random variables in [0,1]. The parameters c_1 and c_2 are called cognitive and social parameters. The parameter ω is called inertia weight and it is used to control the trade-off between the global exploration and the local exploitation ability of the swarm.

3.2. Selection operator

For particle x, we defined the new fitness function as following:

$$F(x)=\begin{cases} f_1(x) & x\in D \\ \max(\frac{p}{N}\cdot f_1^{min}+\frac{N-p}{N}\cdot & \\ f_1^{max}, f_1(x))+f_2(x) & x\notin D \end{cases} \tag{5}$$

Where N is the particle swarm size, p is the number of feasible particles of current generation. f_1^{min} and f_1^{max} is the smallest value and the biggest value of feasible particles of current generation, respectively. for any two particles, we can see from the fitness function (5):

1. If the two particles are both feasible, then the particle with a minimum value of the first objective function of problem (2) wins.
2. If both of the particles are infeasible, the particle with a minimum value of the second objective function of problem (2) wins.
3. If one is infeasible, the other is feasible. then compare the deviation degree of The two particle far away from the f_1^{min}, when the bigger

of the proportion of feasible particle, the bigger of the probability to choice the infeasible in current particle swarm. On the contrary, the bigger of the proportion of infeasible particle in current particle swarm, the bigger of the probability to choice the feasible.

3.3. Self-adaptive variation of ω

From (3), It's obvious that a large inertia weight can make the PSO to explore the search space of problem, while a small one tends to facilitates exploitation. Hence, the inertia weight is a very important parameter to balance the global and local search. To evaluate the diversity of the swarm, we defined a diversity measure as follows:

$$\lambda(t) = \frac{\sum_{i=1}^{N}(f_i^t - \frac{(\sum_{i=1}^{N} f_i^t)}{N})^2}{N \cdot \max\limits_{k\in\{1,2,\cdots,t\}}\{\sum_{i=1}^{N}(f_i^k - \frac{(\sum_{i=1}^{N} f_i^k)}{N})^2\}} \qquad (6)$$

where f_i^t present the fitness value of the $i-$th particle in $t-$th generation. The bigger the value of $\lambda(t)$, the better the diversity of the swarm is. the smaller the value of $\lambda(t)$, the more crowded the swarm is. When the swarm become very crowded, it is difficult for the algorithm to jump out from the local optimal solution. Thus, we can define the self-adaptive variation inertia weight based on the dynamic parameter $\lambda(t)$ as $\omega(t) = \frac{g_{size}-t}{(\lambda(t)+\epsilon)\cdot(g_{size}+0.4)}$, where g_{size} be the maximum generation of swarm, t be the current generation. ϵ is a very small positive number, It can be seen from the $\omega(t)$ that when particle are crowded, that's to say $\lambda(t)$ is small, $\omega(t)$ will become big in order to enhance the ability of global search of the algorithm.

4. The multiobjective PSO algorithm

Step1. (Initialization) Given the particle swarm size N , randomly generate initial swarm $p(0)$ in $[L, U]$, and let $t = 0$.
Step2. (Update velocity and position)For each of the particle in $p(t)$, it was manipulated to find its good position and velocity based on the new selection operator in current generation. Then update each particle's velocity and position according to the velocity-position equation and constitute a temporary particle swarm $c(t)$.
Step3. Utilize the crossover operator proposed in [6] to generate the offspring of the particles in $c(t)$.
Step4. (Crossover)Select N individuals by the selection operator from $p(t)\bigcup c(t)$ and constitute the next particle swarm $p(t+1)$, and use the best solution $x_t' = \arg\min\limits_{x\in D\cap p(t)} f_1(x)$ to replace the particle which its fitness value is biggest in the $p(t+1)$, let $t = t+1$.
Step5. (Stop criterion)If $t = T$, the best particle in $p(t)$ is as the optimal and the algorithm is stopped; otherwise, go to Step 2.

5. Simulation results

5.1. Test Functions

Test Problem $F1$**([7])**
$\min f(x) = (x_1 - 10)^2 + (x_2 - 20)^2$.
Subject to: $g_1(x) = (x_1-5)^2+(x_2-5)^2-100 \geq 0$, $g_2(x) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$, $13 \leq x_1 \leq 100$,$0 \leq x_2 \leq 100$. The best known solution is $f^* = -6961.81388$.
Test Problem $F2$**([7])**
$\min f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$.
Subject to: $g_1(x) = 127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_3 \geq 0$, $g_2(x) = 196-23x_1-3x_2^2-6x_6^2-8x_7 \geq 0$, $g_3(x) = 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0$, $g_4(x) = -4x_1^2 - x_2^2 + 3x_1x_2 - 5x_6 + 11x_7 \geq 0$, $-10 \leq x_i \leq 10$,$i = 1, 2, \cdots, 7$. The best known solution is $f^* = 680.63005$.
Test Problem $F3$**([8])**
$\max f(x) = |\frac{\sum\limits_{i=1}^{n} cos^4(x_i)-2\prod\limits_{i=1}^{n}\cos^2(x_i)}{\sqrt{\sum\limits_{i=1}^{n} ix_i^2}}|$.
Subject to: $g_1(x) = 0.75-\prod\limits_{i=1}^{n} \leq 0$, $g_2(x) = \sum\limits_{i=1}^{n} x_i - 0.75n \leq 0$, n=20,$0\leq x_i \leq 10(i = 1, 2, \cdots, n)$. The global maximum is unknown; the best reported solution[10] is $f^* = -0.803619$. Constraint g_1 is close to being active($g_1 = -10^{-8}$).
Test Problem $F4$**([8])**
$\max f(x) = \frac{\sin^3(2\pi x_1)sin(2\pi x_2)}{x_1^3(x_1+x_2)}$.
Subject to: $g_1(x) = x_1^2 - x_2 + 1 \leq 0$, $g_2(x) = 1 - x_1 + (x_2 - 4)^2 \leq 0$, $0 \leq x_1, x_2 \leq 10$. the best reported solution is $f^* = -0.095825$.
Test Problem $F5$**([8])**
$\min f(x) = e^{x_1x_2x_3x_4}$.
Subject to: $6x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$, $x_2x_3 - 5x_4x_5 = 0$, $x_1^3 + x2^3 + 1 = 0$,$-2.3 \leq x_i \leq 2.3, i = 1, 2$,$-3.2 \leq x_i \geq 3.2(i = 3, 4, 5)$. the best reported solution is $f^* = 0.0539498$.

5.2. Result and comparison

In the simulation, we termed our approach as MOPSO, The particle swarm size $N = 200$, $c_1 = c_2 = 0.5$, $r_1, r_2 \in rand[0, 1]$. For each problems, the best reult(B.R), the mean(M.) and the worst result(W.R) obtained by MOPSO in all 20 runs were recorded. All the result obtainted by MOPSO was compared with the existing ones obtainted from the refrences [6], [7], [8] and [9] in table 1.

Pr.	Opt.	Met.	B.R	M.	W.R
		MOPSO	0.7523	0.7545	0.7586
		Ref.[7]	-6961.837	-6961.774	-6961.456
$F1$	-6961.81	Ref.[8]	-6961.814	-6875.940	-6350.262
		Ref.[9]	-6961.814	-6961.813	-6961.810
		Ref.10]	N.A.	N.A.	N.A.
		MOPSO	0.7523	0.7545	0.7586
		Ref.[7]	680.636	68.683	680.876
$F2$	680.630	Ref.[8]	680.630	680.656	680.763
		Ref.[9]	680.630	680.631	680.634
		Ref.[10]	N.A.	N.A.	N.A.
		MOPSO	0.7523	0.7545	0.7586
		Ref.[7]	N.A.	N.A.	N.A.
$F3$	-0.8036	Ref.[8]	-0.803515	-0.781975	-0.726288
		Ref.[9]	-0.803376	-0.793281	-0.769291
		Ref.[10]	-0.787933	-0.751301	-0.689747
		MOPSO	0.7523	0.7545	0.7586
		Ref.[7]	N.A.	N.A.	N.A.
$F4$	-0.0958	Ref.[8]	-0.095825	-0.095825	-0.095825
		Ref.[9]	-0.095825	-0.095825	-0.095825
		Ref.[10]	-0.095825	-0.095825	-0.095825
		MOPSO	0.7523	0.7545	0.7586
		Ref.[7]	N.A.	N.A.	N.A.
$F5$	0.0539	Ref.[8]	0.053945	0.054179	0.056224
		Ref.[9]	N.A.	N.A.	N.A.
		Ref.[10]	N.A.	N.A.	N.A.

Table 1: Comparison of the best, mean and worst results among MOPSO and other algorithms in difrence reference [6], [7], [8] and [9] for each test problems. N.A. represents the coresponding result is not available.

6. Conclusions

In this paper, we first transform the nonlinear constrained programming problem into a bi-objective optimization problem, and then a new multiobjective PSO algorithm which dose not requires the use of a penalty function is proposed. The numerical simulations on five test problems also indicate the effectiveness of the proposed algorithm.

Acknowledgement

This work was partially supported by the Natural Science Basic Research Plan Project in Shaanxi Province of China (No.2006A12), the Science and Technology Research Project of Education Department of Shaanxi Province of China (No.07JK180) and the Emphasis Research Plan Project of Baoji University of Arts and Sciences (No.ZK0619).

References

[1] K. Deb, S. A. Agrawal. Niched-penalty approach for constraint handing in genetic algorithms. *Artificial Neural Nets and Genetic Algorithms, Proc. of the inter. Conf.*, pp. 235-243, In Portoroz Slovenia,1999, Andrej, Dobnikar Eds, Springer Verlag, Wien, New York,1999.

[2] R. Farmani, J. Wright. Self-adaptive fitness formulation for constrained optimization. *IEEE Trans. Evol. Comput.*,7:445-455, 2003.

[3] J. Kennedy, R. Eberhart. Particle Swarm Optimization. *IEEE Int'1 Conf on Neural Networks*, pp. 678-682, Perth, Australia,Carlos A, Coello, Eds. Springer Verlag, 1995.

[4] A. Hernandez-Aguirre, C. Coello. Passss: An Implementation of a Novel Diversity Strategy to Handle Constraints. *In Proceeding of the 2004 Congress on Evolutionary Computation CEC-2004,* IEEE Press. pp. 503-410, 2004.

[5] S. Tsutsui, M. Yamamura. Multi-parent recombination with simplex crossover in real coded genetic algorithms, *In Proceeding of the Genetic and Evolutionary Computation Conference,* pp. 57-664, Banzhaf W, Daida J, Eiben E, eds., San Mateo, California, Morgan Kaufmann Publishers, 1999.

[6] K. E. Parsopoulos, M. N. Vrahatis. Particle swarm optimization method for constrained optimization problems. *In Proceedings of the 5th Conference on Parallel Problems solving from Nature,* pp. 254-259, Springer Verlag, Wien, Koziel, Mich alewics, 1999.

[7] T. P. Runarsson, X. Yao. Stockastic ranking for constrained evolutionary optimization.*IEEE transcation on envolutionary computations*, 4:284-294, 2000.

[8] A. H. Aguirre, S. B. Rionda, C. C. Carlos. IS-PAES: A constraints handling technique based on multi-objective concepts. C.M. Fonseca et al.(Eds.),*EMO2003, LNCS 2632*, pp. 73-87, 2003.

[9] C. C. Carlos. E. Mezura-Montes. Handleing constrains in genetic algorithm using dominace-based tournaments. *In Proceedings of the 5th Conference on Computational Intelligence and Multimedia Applications (ICCIMA'03)*, pp. 236-239, Springer Verlag, Wien, New York, 2003.