TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.1, January 2014, pp. 703 ~ 710
DOI: http://dx.doi.org/10.11591/telkomnika.v12i1.3189

Chaos Adaptive Improved Particle Swarm Algorithm for Solving Multi-Objective Optimization

Bing Xiang Liu[1], Yan Wu[1], Xing Xu*[1, 2], Na Hu[1], Xiang Cheng[1]
[1]Information engineering college,Jingdezhen ceramic institute, Jingdezhen, JiangXi, 333403
[2]State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China;
*Corresponding author: whuxx84@aliyun.com

Abstract

To overcome the problem of premature convergence on Particle Swarm Optimization (PSO), this paper proposes both the improved particle swarm optimization methods (IPSO) based on self-adaptive regulation strategy and the Chaos Theory. Given the effective balance of particles' searching and development ability, the self-adaptive regulation strategy is employed to optimize the inertia weight. To improve efficiency and quality of searching, the learning factor is optimized by generating Chaotic Sequences by Chaos Theory. The improved method proposed in this paper achieves better convergence performance and increases the searching speed. Simulation results of some typical optimization problems and comparisons with typical multi-objective optimization algorithms show that IPSO has a fast convergence speed, the diversity of non-dominated and the ideal convergence. The algorithm meets the requirements of multi-objective optimization problem.

Keywords*: Particle Swarm Optimization, multi-objective optimization, Chaos Theory, self-adaptive regulation strategy*

1. Introduction

Multi-objective optimization problem (MOP) is the Frequently Asked Questions in the field of engineering, which is composed of the multiple pending optimization targets. Its main characteristic is a conflict exists among the targets, all of which are not able to obtain the optimal value at the same time [1]. While designing a new product, the yield, quality, the cost consumption, profits and so on are usually considered. In order to achieve the high yield, high-quality, low-consumption, low-cost, high-profit, one needs to establish the most optimization design models containing a multiple of targets.

The MOP method can be divided into the traditional multi-objective optimization methods and algorithms based on swarm intelligence multi-objective optimization method. These traditional methods are: evaluation function method [2], interactive Planning Law [3], layered Solving method [4], etc. whose essence is to transfer the objective function of each sub multi-objective to single objective function and solve it. These traditional methods encountered much difficulty in solving high-dimensional complex multi-objective problems, such as the different nature of the sub-target unit, thus can not compare and solve unsatisfactory results related [5].

In recent years, the intelligent algorithm groups used in the multi-objective optimization are widely Concerned by a lot of scholars, and many of the multiobjective optimization algorithms based on swarm intelligence algorithm are put forward [6, 7]. Such as, the Fonseca and Fleming put forward MOGA algorithm [8]. This algorithm is too dependent on the choice of the shared function, and it needs to determine the shared radius which may generate a larger selective pressure, so as to the immature convergence. Horn and Nafpliotis have also put forward NPGA algorithm [9], but it is difficult to select the radius of the niche for the algorithm in adjustment, as well as select the suitable comparison set of the proper scale, so that the optimization results are not ideal. Srinivas and Deb have put forward NSGA algorithm [10], whose advantage is the arbitrary choice of multiple targets optimization, and the ability of getting uniformly distributed non-inferiority optimal solution, but its drawback is a lower computational efficiency with calculating complexity degrees O (MN3) (M- target of the quantity, N for the population size). Thiele and Zitzler put forward multi-objective evolutionary algorithm

Received May 13, 2013; Revised July 8, 2013; Accepted August 18, 2013

SPEA [11]. This algorithm adopts the elite reserves strategy, though the computation efficiency is high, but its computational complexity degree is as high as the cube of the population. Deb and Pratap put forward the second generation of NSGA, referred to as the NSGA-II [12], whose the computational complexity degree O(MN2), the complexity of the algorithm significantly reduce. This algorithm introduces the fast non-inferiority sorting and the new the diversity of protection method, to overcome the shortcomings of the NSGA. We put forward a crowded conception, and have the elitist strategy, is conducive to maintain the uniformity of the excellent individuals reconciliation, to improve the level of the populations as a whole evolutionary. NSGA-II algorithm, as a kind of typical multi-objective optimization algorithm, is based on the NSGA-II algorithm. However, this algorithm has some problems, for example, the ability to adapt to changes in the search space. The algorithm is easy to cause premature convergence and strong randomness of the particle search in an iterative process, resulting in inefficient search.

Over the past decade, the particle swarm optimization algorithm is applied in solving multi-objective optimization problem and has been concerned by many scholars widespread. PSO and genetic algorithm (GA) are all swarm intelligence algorithms, but compared to GA, PSO, with less parameter to adjust, has stronger global search ability, and is easier to implement, so it is favored by researchers. But in the application of particle swarm algorithm to solve multi-objective optimization problems, the researchers found that: particle swarm optimization in the optimization process is prone to premature convergence, and especially more difficult in solving high-dimensional complex objective function.

The convergence speed is very slow in approaching or entering optimal region, resulting in a local extreme, with a less ideal solution. As to PSO premature convergence problem, domestic and foreign scholars have proposed a variety of improvement programs. Xuesong Yan [13] proposed orthogonal particle swarm optimization algorithm and used the proposed algorithm for digital circuit optimization design; S.-H. Zhou [14] used fuzzy membership functions and adaptive strategy to improve the PSO algorithm and proposed adaptive fuzzy particle swarm optimization (AFPSO); A QoS-based hybrid particle swarm optimization (GHPSO) to schedule applications to cloud resources is presented and in GHPSO, crossover and mutation of genetic algorithm is embedded into the particle swarm optimization algorithm [15]. Chen MinYou [16] proposed a multi-objective optimization method based on adaptive particle swarm algorithm. Xu Gang et al [17] proposed a hybrid of particle swarm multi-objective optimization algorithm and multi-objective search for particle algorithm. Most of them are improved by adjusting the parameters of the algorithm, such as putting forward a variety of linear and non-linear inertia weight dynamic adjustment strategy, or introducing a shrinkage factor. The improved PSO algorithm has been improved on the performance and efficiency, but there is still much room for improvement. Therefore, in order to provide a better performance, a more efficient, but lower-cost particle swarm optimization algorithm, in new ways to improve algorithm have been explored and tried by academic and industry researchers [6, 18, 19].

In view of the proposed research questions, we use the chaos theory and adaptive power adjustment strategy to improve the standard particle swarm algorithm, and propose the improved chaotic adaptive particle swarm optimization (IPSO). Inertia weight of the algorithm is evolutionary by the adaptive strategy, and the learning factor is optimized by chaotic sequences generated by chaos theory. Improved particle swarm algorithm can improve the algorithm's premature problem and improve the search speed. Finally, the algorithm is applied to multi-objective optimization problem to verify the performance convergence accuracy and speed, global convergence effect of the algorithm.

2. Chaos Adaptive Particle Swarm Optimization (APSO)

2.1. Standard Particle Swarm Optimization (PSO)

Particle Swarm Optimization (Particle Swarm Optimization, PSO) [20] is an evolutionary algorithm group based on the social behavior for birds, propsed by scholars Eberhart and Kennedy in 1995. PSO algorithm is inspired by the behavior of the biological communities for solving optimization problems. In the particle optimization process, the potential solution is envisaged to a "particle" in the n-dimensional space. The particles fly in the solution space at a certain speed and direction. In an iterative process, each particle has two global variables, the best position of all particles (pbest) and the best position of itself. Assuming that in an n-

dimensional search space, a population $X = \left(x_{1,} x_{2,} \mathrm{K} , x_{n} \right)^{T}$ is composed of m particles, the position of the i-th particle is expressed as $x_{i} = \left(x_{i,1,} x_{i,2,} \mathrm{K} , x_{i,n} \right)^{T}$, its speed $x_{i} = \left(x_{i,1,} x_{i,2,} \mathrm{K} , x_{i,n} \right)^{T}$, its individual extreme $p_{i} = \left(p_{i,1,} p_{i,2,} \mathrm{K} , p_{i,n} \right)^{T}$, global extreme value $p_{g} = \left(p_{g,1,} p_{g,2,} \mathrm{K} , p_{g,n} \right)^{T}$. In the (k + 1) th iteration process, the particles update their own pace and location by the Eq. (1) and (2).

$$v_{i,d}^{k+1} = \omega v_{i,d}^{k} + c_{1} rand()(p_{i,d}^{k} - x_{i,d}^{k}) + c_{2} rand()(p_{g,d}^{k} - x_{i,d}^{k}) \quad (1)$$

$$\mathrm{x}_{\mathrm{i,d}}^{\mathrm{k+1}} = \mathrm{x}_{\mathrm{i,d}}^{\mathrm{k}} + \mathrm{v}_{\mathrm{i,d}}^{\mathrm{k+1}} \quad (2)$$

where ω is inertia weight, which maintains particle inertia with the ability to extend the search space; C1 and C2 are learning factors, on behalf of each particle into the best position in the statistical acceleration term weight; rand () is a random number between (0,1), $v_{i,d}^{k}$ and $x_{i,d}^{k}$, the speed and position of the d-dimensional particles in the k-th iteration; $p_{i,d}^{k}$ the position of the d-dimensional individual extreme of particle i. $p_{g,d}^{k}$ the location of global extreme value in d-dimensional groups.

2.2. Chaos Adaptive Particle Swarm Optimization

PSO algorithm has the advantages as follows: being easy to describe, easy to implement, little parameters to adjust, high convergence speed, low computational cost, and no demands on memory and CPU speed; It has been proven to be an effective method of optimization problem. Standard PSO algorithm has its own limitations, for example, the algorithm implementation process has a great relationship with the value of parameters; In the high dimensional complex optimization problems, the algorithm is very easy to stay at some point, which is not the global optimum yet. It is the premature convergence; in addition, algorithm convergence speed has become markedly slow when approaching or entering the optimal solution area.

In order to solve the disadvantages mentioned above and to improve the premature convergence of the algorithm and the speed of convergence, the article optimizes weighting factor inertia weight ω to the standard PSO algorithm, learning factor C1, C2 three parameters with adaptive weight adjustment strategy and chaos theory. We get the chaotic Self-adaptive particle swarm Optimization (IPSO) algorithm. Inertia weight factor is adjusted by the Eq. (3).

$$\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min}) \left\{ \frac{(iter_{\max} - iter)^{n}}{(iter_{\max})^{n}} \right\} \quad (3)$$

where iter is the current number of iterations of the algorithm. $iter_{\max}$ is the maximum number of iterations of PSO algorithm allowed to perform. n is the non-linear modulation index.

In the process of PSO algorithm optimization iteration, the learning factor C1, C2 are adjusted by the chaotic sequence generated through the chaos theory. Because the chaotic variables change with randomness, ergodicity and regularity, the IPSO algorithm can maintain the diversity of the population to overcome the problem of premature convergence and improve the performance of global search. This article takes advantage of the classic LORENZ's

equation to achieve the evolution of the chaotic variable and optimize the search, as shown in Eq. (4) above.

$$\begin{cases} \dfrac{dx}{dt} = -(y+z) \\ \dfrac{dy}{dt} = x + ay \\ \dfrac{dz}{dt} = b + xz - cz \end{cases} \quad (4)$$

In Eq.4, parameters a, r and b are control parameters. In Lorenz equation, valid values for a, r and b, are a = 10, R = 28 and b = 8/3 respectively. The IPSO algorithm performs the following steps:

(1) initialize the particle swarm

To initialize the position and velocity of the particles in the PSO algorithm, initial position and velocity of the particle are generated randomly. And each particle current position is set as the particle individual extreme, and the optimal individual extreme value is selected as the global optimum.

(2) calculate the value of the adaptation of the particle in group;

(3) compare the adaptation value of each particle with the adapted value in the best position. If better, the current position is set as the best location;

(4) compare the fitness value of each particle with the fitness value of the global best position. if better, set the current position as the global best position;

(5) compute learning factor C1, C2 and inertia weight weight ω to get a new inertia weight and learning factors, to optimize the speed and position of the particle;

(6) If the end condition meets, the global best position is the optimal solution, save the results and exit. Otherwise, returns to step (2).

3. Results and Analysis

In order to test the performance of the chaotic adaptive particle swarm optimization, choose two standard test functions proposed by Schaffer [21] and ZDT3 proposed by Deb [6] as a test case; test functions are as follows.

Test function SCH1:

$$\min f(x) = (f_1(x), f_2(x))$$

$$s.t. x \in [-5, 7]$$

where $f_1(x) = x^2$, $f_2(x) = (x-2)^2$

Test function SCH2 :

$$\min f(x) = (f_1(x), f_2(x))$$

$$s.t. x \in [-5, 10]$$

where $f_1(x) = \begin{cases} -x, (x \le 1) \\ -2 + x, (1 < x \le 3) \\ 4 - x, (3 < x \le 4) \\ -4 + x, (x > 4) \end{cases}$, $f_2 = (x-5)^2$

Test function ZDT3 :

$$\min f(x) = (f_1(x), f_2(x))$$

$$s.t. \chi_i \in [0, 1]$$

where $f_1(x) = x_1$, $f_2(x) = g(x)[1-\sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)]$,

$g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1)$

Perform simulation optimization experiments on the test function SCH1, SCH2, ZDT3 with IPSO algorithm. Set the number of particles and the number of iterations as 100 and 50. Inertia weight ω and learning factors C_1, C_2 are calculated according to the formula (3) and (4) respectively. The test results are shown in Figure 1, Figure 2, and Figure 3.

From Figure 1 – Figure 3, the three test functions present interface accurately, and the three test functions are simulated by the algorithm and the complete Pareto curve is presented. Especially for the ZDT3 test function which is more difficult, the target vector is well-distributed. Therefore, it is valuable to the multi-objective optimization problem in the project.

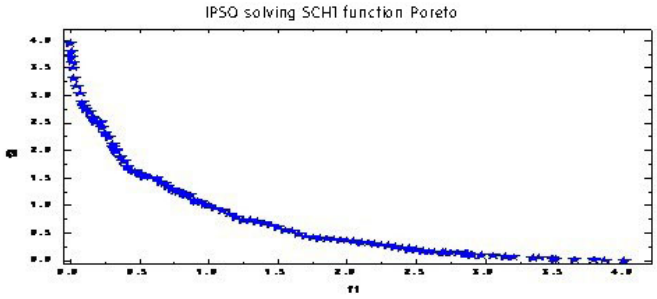

Figure 1. IPSO solving SCH1 function Pareto

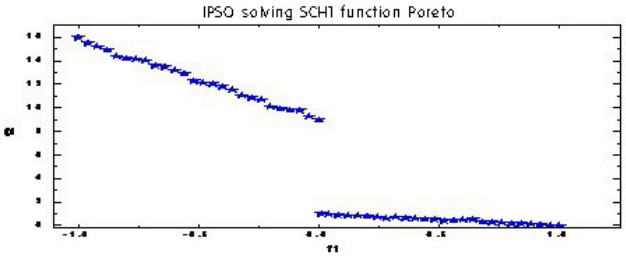

Figure 2. IPSO solving SCH1 function Pareto

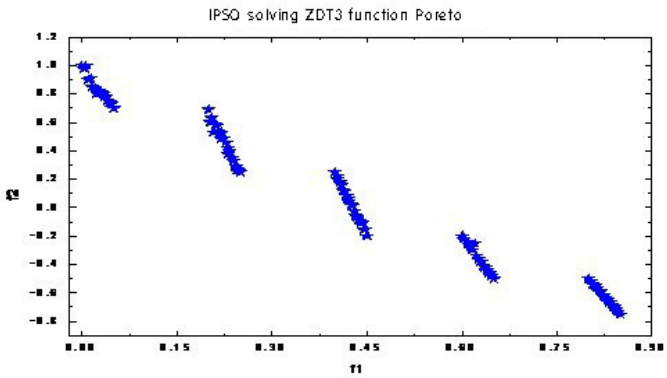

Figure 3. IPSO solving ZDT3 function Pareto

In order to assess the non-inferiority Convergence of Solutions and distribution uniformity, this article assesses the performance of the algorithm with the indicators of Convergence and distribution, which are defined respectively as follows [18, 19]:
(1) Gonvergence Distance (GD) GD is used to describe the distance between the non-dominated solution that the algorithm has searched and the true Pareto optimal front-end.

$$GD = \frac{\sqrt{\sum_{i=1}^{N} d_i^2}}{N}$$

where N is the number of non-dominated solutions which algorithm searched, d_i^2 means the shortest Euclidean distance between the noninferior solution i and the true Pareto optimal front-end solution.
(2) Distribution index SP, is used to evaluate the uniformity of distribution of non-dominated solution concentrated solution.

$$SP = \frac{[\frac{1}{n}\sum_{i=1}^{n}(\bar{d} - d_i)^2]^2}{\bar{d}},$$

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n} d_i$$

where n is the number of non--dominated solutions in centralized solution, d_i is the shortest Qu Distance between the i-th non-inferior solution and all the solutions among the real Pareto optimal front-end.

By using IPSO algorithm to run each test function for 30 times, get test function convergence indicators GD, distribution indicators SP and average computation time CT, and calculate the average value of three standard test functions GD, SP and CT, results are shown as the table 1 below.

Table 1. IPSO optimizing test function performance statistics

Algorithm	SCH1	SCH2	ZDT3	average
GD	0.000344	0.000336	0.000327	0.000336
SP	0.0034	0.0037	0.0031	0.0034
CT	2.023	2.134	2.213	2.123

Evaluation indicators GD, SP, CT confirmed the accuracy of the IPSO algorithm for multi-objective Pareto. GD shows that the noninferior solution is very close to the true Pareto optimal front-end; SP shows that the noninferior solution is well-distribution; CT shows that the running time spent is in the allowable range.

Compare the IPSO algorithm with the average of convergence indicators GD, distribution index SP and calculation time CT of the three test functions solved by classic non-inferiority classified multi-objective genetic algorithm (NSGA-II), multi-objective particle swarm optimization (MOPSO) algorithm. Results are shown in Table 2.

Table 2. contrast of results of the optimization test function in three algorithms

Algorithm	NSGA II	MOPSO	IPSO
GD	0.000353	0.000355	0.000336
SP	0.0036	0.0040	0.0034
CT	2.402	0.076	2.123

By the results in Table 2, IPSO algorithm GD is significantly better than NSGA II and MOPSO GD, the distance between the noninferior solution obtained and the true Pareto optimal front-end decreased by 4.8% and 5.4%. SP evaluate the distribution of the target space by calculating the distance changes between the each individual and the the neighbors. The smaller the value its Description distribution, the better the situation is. Table 2 shows that the SP value of IPSO is minimum, indicating that the IPSO algorithm non--inferior solution is more uniform distribution relative to the other two algorithms. In terms of computation time CT, the time spent in the process of IPSO running is less than NSGA II, but time-consuming is higher than the MOPSO algorithms, because the search process in the improved algorithm is not isometric step search and the standard PSO algorithm is, and its flight direction is single. It is clear that time-consuming of IPSO algorithm is more than the standard PSO algorithm. But IPSO time-consuming is within the scope allowed. In summary, by comparing performance indicators of GD, SP and CT with other algorithms, the feasibility and effectiveness of the proposed algorithm has been verified.

4. Conclusion

The paper proposes a chaotic Adaptive Improved Particle Swarm Optimization (IPSO). The algorithm uses chaos theory and adaptive strategy to optimize the parameters of the PSO algorithm to overcome the premature convergence of the PSO algorithm, and improve the convergence rate. The solution set is better distributed. The experimental results of three standard test function show that while the algorithm solving multi-objective problem, the resulting non-dominated solutions can be a good approximation of the Pareto optimal solution set, and evenly distributed. While comparing the proposed algorithm with other algorithms, the IPSO algorithm performs better.

Acknowledgments

This paper was supported by the National Science and Technology Support Plan (2012BAH25F02, 2013BAF02B01), State Key Laboratory of Software Engineering (SKLSE2012-09-35), the National Natural Science Foundation of Jiangxi Province (20122BAB211036, 20122BAB201044), the National Nature Science Foundation of China (61202313, 61203310), the Science Foundation of Jiangxi Provincial Department of Education (GJJ13633).

References

[1] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the Strength Pareto approach. *IEEE Transactions on Evolutionary Computation*. 1999; 3(4): 257-271.
[2] Kampolis IC, Giannakoglou KC. A multilevel approach to single- and multiobjective aerodynamic optimization. *Computer Methods in Applied Mechanics and Engineering*. 2008; 197(33-40): 2963-2975.
[3] Li LS, Lai KK. A fuzzy approach to the multiobjective transportation problem, *Computers & Operations Research*. 2000; 27(1): 43-57.
[4] Gunzburger MD, Lee J. A domain decomposition method for optimization problems for partial differential equations. *Computers & Mathematics with Applications*. 2000; 40(2-3): 177-192.
[5] Zitzler E, Deb K, Thiele L. Comparison of Multiobjective Evolutionary Algorithms. *Empirical Results, Evolutionary Computation*. 2000; 8(2): 173-195.
[6] Deb K. Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*. 1999; 7(3): 205-230.
[7] Laumanns M, Thiele L, Deb K, Zitzler E. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*. 2002; 10(3): 263-282.
[8] Fonseca C M, Fleming P J. *Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization*. Proceedings of the Fifth International Conference on Genetic Algorithms. Urbana-Champaign. 1993: 416-423.
[9] Horn J, Nafpliotis N, Goldberg DE. *Multiobjective optimization using the niched Pareto genetic algorithm*. IlliGAL Report number: 93005. 1993.
[10] Srinivas N, Deb K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*. 1994; 2(3): 221-248.

[11] Zitzler E, Thiele L. *Multiobjective optimization using evolutionary algorithms - A comparative case study*. Parallel Problem Solving from Nature. Amsterdam. 1998: 292-301.
[12] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. 2002; 6(2): 182-197.
[13] Yan X, Wu Q, Liu H. Orthogonal Particle Swarm Optimization Algorithm and Its Application in Circuit Design. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(6): 2926-2932.
[14] Zhou S, Ouyang C, Liu C, Zhu P. Adaptive fuzzy particle swarm optimization algorithm. *Computer Engineering and Applications*. 2010; 46(33): 46-48.
[15] Xue S, Wu W. Scheduling Workflow in Cloud Computing Based on Hybrid Particle Swarm Algorithm. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2012; 10(7): 1560-1566.
[16] Chen M, Zhang C, Luo C. An adaptive evolutionary particle swarm algorithm for multi-objective optimization. *Journal of System Simulation*. 2009; 21 (22):7061-7065.
[17] Xu G, Qu J. method for multi-objective optimization of hybrid particle swarm optimization algorithm. *Computer Engineering and Applications*. 2008; (33): 18-21.
[18] Tan KC, Lee TH, Khor EF. Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons. *Artificial Intelligence Review*. 2002; 17(4): 253-290.
[19] Zitzler E, Thiele L, Laumanns M, Fonseca CM, Fonseca VG. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*. 2003; 7(2): 117-132.
[20] Kennedy J, Eberhart R. *Particle swarm optimization*. IEEE International Conference on Neural Networks Proceedings. Perth. 1995: 1942-1948.
[21] Schaffer JD. *Multiple objective optimization with vector evaluated genetic algorithms*. Proceedings of the 1st international Conference on Genetic Algorithms. Pittsburgh. 1985: 93-100.