An Improved Particle Swarm Optimizer with Shuffled Sub-swarms and Its Application in Soft-sensor of Gasoline Endpoint

Hui Wang Feng Qian

State-Key Laboratory of Chemical Engineering, Ecust China University of Science and Technology, Shanghai 200237, P. R. China

Abstract

This paper proposes a shuffled sub-swarms particle optimizer algorithm (SSPSO) to enhance the diversity of particles in the swarm to improve the performance of PSO. SSPSO is tested with a series of benchmark functions and compared with other version PSO algorithms. Experimental results show that SSPSO improves the search performance on the benchmark functions significantly. Furthermore, SSPSO is used to train NN to construct an artificial neural network SSPSONN. Then SSPSONN is applied to construct a soft-sensor of gasoline endpoint and compared with actual industrial data, the results show that the constructed soft-sensor is feasible and effective.

Keywords: Particle swarm optimizer, Sub-swarm, Shuffled, Gasoline endpoint, Soft-sensor

1. Introduction

Particle swarm optimizer (PSO) is proposed by Kennedy and Eberhart in 1995 [1], which is inspired by the social behavior of animals such as fish schooling and bird flocking has already come to be widely used in many areas [2] [3] [4][5][6]. As a population-based algorithm PSO has a faster convergence rate than other evolutionary algorithms on some problems and has few parameters to adjust this makes it easy to implement [7]. However Angeline point out that PSO may outperform in the initial iterations, the algorithm might encounter problems in reaching optimum solutions for some function approximation problems [8].And all the particles are attracted towards the best position so far by any of particles, this makes particle swarm lost its diversity and induces PSO algorithm easy trapped in the local optima. Therefore the study of PSO principle is ongoing [9] and many methods have been proposed out to improve the performance of PSO recently. Løbjerg, Rasmussen and Krink propose a hybrid PSO with breeding and subpopulation [10], Parsopoulos propose a new scheme that harnesses the local and global variants of PSO which named as UPSO in [11]. Some interaction mechanisms between particles have been taken in [12] to improve the performance. Chatterjee and Siarry propose nonlinearly varying inertia weight variant of PSO in [13] and some researchers combine PSO with other techniques to improve the performance of PSO [14][15][16].

This paper proposes a shuffled sub-swarms particle swarm optimizer (SSPSO) algorithm which is devoted to solve the problems of premature convergence and lacking in diversity encountered in many applications of PSO.

The structure of this paper is the following. Section 2 presents SSPSO and SSPSO is used to optimize some benchmark functions comparing with other version PSO in Section 3. Subsequently, SSPSO applies in soft-sensor of gasoline endpoint in Section 4 .Finally, Section 5 summaries the study.

2. Particle swarm optimizer improved with shuffled sub-swarms

In PSO particles are randomly initialized with position and velocities and fly to the best position of the swarm which represents the optimum value of the function to be optimized. Each particle is treated as a point in a D -dimensional space and the i th particle is represented as $X_I = (x_{i1}, x_{i2}, ..., x_{iD})$ .The best previous position of the i th is record and represented as $P_I = (p_{i1}, p_{i2}, ..., p_{iD})$ .The index of the best particle of all the particles in the swarm is represented by p_g . The velocity of particle i is represented as $V_I = (v_{i1}, v_{i2}, ..., v_{iD})$ .The performance of each particle is measure by a pre-defined fitness function that is problem specific. In each iteration the velocity of each particle is updated according to their best encountered position and the best position encountered by any particle. And each particle updates the velocity and position according to the following equations:

$$v_{id}(k+1) = wv_{id}(k) + c_1R_1(p_{id} - x_{id}(k)) + c_2R_2(p_{gd} - x_{id}(k)) \quad (1)$$

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k+1) \quad (2)$$

Where c_1 and c_2 are two positive constants, R_1 and R_2 are random numbers in the range of [0, 1] and w is employed to trade-off between the global and local exploration abilities of the particles [17].

An important variation of PSO is the constriction factor approach PSO (CPSO) [18], the velocity of it is update by the fellow equation:

$$v_{id}(k+1) = \chi \cdot (v_{id}(k) + c_1R_1(p_{id} - x_{id}(k) + c_2R_2(p_{gd} - x_{id}(k))) \quad (3)$$

Where,

$$\chi = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \phi = c_1 + c_2, \phi > 4.$$

All particles in the swarm fly to the best position of any particles so far this easy to make PSO algorithm falling into local optima for the particle diversity lose. In SSPSO, we partition the swarm into sub-swarms to increase the particle diversity and interaction mechanism between sub-swarms will be done to enhance the ability of particles. The divisions of sub-swarm are not randomly but based on the fitness of particles and each sub-swarm has the same population size. Within each sub-swarm, the individual particles hold ideas of searching for the destination, that can be influenced by the ideas of other particles, and evolve through a process of standard PSO algorithm. After a defined number of standard PSO evolution generations, all sun-swarms will be shuffled to be a new swarm and ideas are passed among sub-swarms in this shuffling process. If the stop condition of the optimum problem is not satisfied, the new swarm will be partitioned into several new sub-swarms to compute again. This process will be continued until the stop condition satisfied.

The scheme of SSPSO can be seen in Fig1.

SSPSO algorithm can be summarized in the following:

Step1: Randomly initialize the positions and velocities of all particles. Set $m =$ the number of sub-swarm and $n =$ the number of particles in each sub-swarm.

Step2: Compute the fitness of the optimal functions of each particle.

Step3: Rank particles by their fitness.

Step4: Partition particles into sub-swarms according to their fitness. For example, for the number of sub-swarm $m = 3$, rank 1 goes to the first sub-swarm, rank 2 goes to the second sub-swarm, rank 3 goes to the third sub-swarm, rank 4 goes to the fist sub-swarm, and so on.

Step5: In each sub-swarm, particle updates its velocity and position on base of Eq. (1) and Eq.(2) to get the best position of the swarm by comparison with its neighbors and itself past in a defined generations .

Step6: Shuffle sub-swarms to be a new swarm after they have iterated for a defined generations and rank particles by their fitness.

Step7: If the stop condition is not satisfied go to Step4.Otherwise, stop and obtain the results from the global best position and the global best fitness.

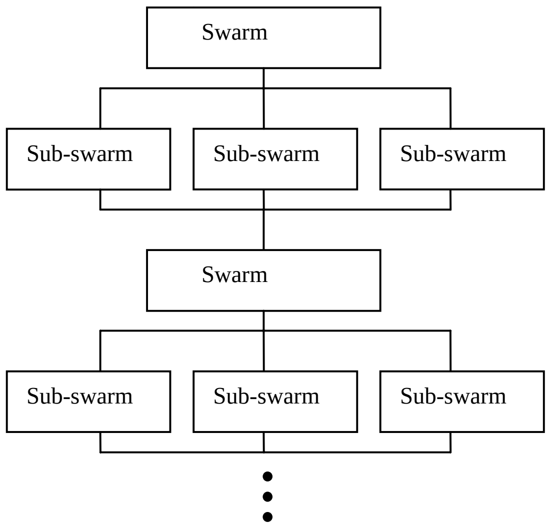

Fig.1 The division and shuffle of sub-swarm mode。

3. Experimental and results

3.1. Test functions

To evaluate the performance of the propose SSPSO, SPSO and CPSO are used for comparisons. And a set of 5 benchmark functions is employed in our experimental studies.

The 5 functions can be described as fellows:

The first test function is Generalized Sphere function given by the equation:

$$f_1(x) = \sum_{i=1}^{n} x_i^2$$

Sphere function is unimodal function, with its global minimum located at $x = (0,...,0)$, with $f(x) = 0$. This function has no interaction between its variables and gradient information always points toward the global minimum.

The second test function is Generalized Rosenbrock function given by the equation:

$$f_2(x) = \sum_{i=1}^{n}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$$

Generalized Ronsenbrock function is a unimodal function, but not all unimodal functions are simple to optimize. The fitness landscape is simple from afar but banana shaped when close to the minimum. Rosenbrock's variables are strongly dependent and gradient information often misleads algorithms. Its global minimum of $f(x) = 0$ is located at $x = (1,...,1)$.

The third test function is Generalized Rastrigin function given by the equation:

$$f_3(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$$

Generalized Rastrigin function is a multimodal function, characterized by deep local minima arranged as sinusoidal bumps. The global minimum is $f(x) = 0$, where $x = 0$. An optimization algorithm can easily become trapped in a local minimum on its way to the global minimum.

The fourth test function is Generalized Griewank function given by the equation:

$$f_4(x) = \frac{1}{400}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i - 100}{\sqrt{i}}) + 1$$

Generalized Griewank function is strongly multi-modal with significant interaction between its variables, caused by the product term. This function has the interesting property that the numbers of local minima increase with dimensionality. However, the influence of the product term also diminishes dramatically in these circumstances. In fact, it becomes negligible once the $n > 30$. The global minimum is $f(x) = 0$, where $x = (100,...,100)$.

The fifth test function is Ackley function given by the equation:

$$f_5(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{n}x_i^2}\right)$$
$$-\exp\left(\frac{1}{30}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e$$

Ackley is a multi-modal function with many local optima. The global minimum is $f(x) = 0$, where $x = 0$. This function is difficult because optimization algorithms can easily be trapped in a local minimum on its way to the global minimum.

These benchmark functions can be grouped as unimodal (function $f_1 - f_2$) and multimodal functions (function $f_3 - f_5$) where the number of local minima increases exponentially with the problem dimension.

3.2. Experimental settings

The dimension of each function n, feasible solution space, and $f_{\min}$ are listed in Table1.

Function	n	Feasible solution space	$f_{\min}$
f_1	30	$[-100,100]^n$	0
f_2	30	$[-30,30]^n$	0
f_3	30	$[-5.12,5.12]^n$	0
f_4	30	$[-600,600]^n$	0
f_5	30	$[-32,32]^n$	0

Table 1: Basic characters of the test functions.

The parameters used for these three algorithms are designed as follows: The maximum velocity $V_{\max}$ and minimum velocity $V_{\min}$ are set at half value of the upper bound and lower bound, respectively. The maximum generations 2000 is set to all algorithms, w decays linearly from 0.9 to 0.4 is used for SSPSO and SPSO. Furthermore, for SSPSO the number of sub-swarm is set to 4 and each sub-swarm evolve 400 generations before they mix. The parameters setting for all algorithms could be seen in Table 2. All experiments are repeated for 100 times independently.

	Population Size	χ	c_1	c_2
SSPSO	100	NA	2.0	2.0
SPSO	100	NA	2.0	2.0
CPSO	100	0.73	2.05	2.05

Table 2: parameters settings.

3.3. Experimental results and comparison

The experimental results for each algorithm on each test function are list in Table3.

From Table3, it can be easy see that SSPSO algorithm is robust and get better global optima than SPSO and CPSO for all the test functions.

In SSPSO, the shuffling of sub-swarms enhances the ability of particles after interaction between different sub-swarms. So with sub-swarms shuffling and evolution, SSPSO enhances the diversity of particles evade falling to the local optima and gets better global optima.

Function	Mean function value(Standard deviation)		
	SSPSO	SPSO	CPSO
f_1	2.65×10^{-38} (5.83×10^{-36})	1.9×10^{-12} (3.7×10^{-12})	2.3×10^{-15} (4.9×10^{-15})
f_2	21.46 (15.74)	52.83 (38.38)	39.70 (31.51)
f_3	18.54 (4.96)	21.56 (5.12)	43.76 (12.13)
f_4	3.1×10^{-3} (2.91×10^{-4})	1.4×10^{-2} (1.6×10^{-2})	1.9×10^{-2} (2.0×10^{-2})
f_5	8.31×10^{-15} (6.52×10^{-15})	9.0×10^{-8} (9.3×10^{-8})	1.6×10^{-8} (1.8×10^{-8})

Table 3: comparison between SSPSO, SPSO and CPSO.

3.4. Discussions

In SSPSO algorithm, the number of particles in each sub-swarm shouldn't too large which induces lost diversity of particles. On the other hand if there are too few particles in each sub-swarm the advantage of local evolution will lose. And, if the generations of every sub-swarm evolutions once a time is too small, the sub-swarm will be shuffled frequently reducing the search ability in each sub-swarm; on the other hand, if it is too large, the algorithm will be shrunk into local optima for lack of intercourse between sub-swarms.

4. SSPSO application in soft-sensor of gasoline endpoint

Gasoline endpoint is an important product quality indicator of the atmospheric tower. But gasoline endpoint can't be measured on-line directly. So, SSPSO is applied to train NN to construct neural networks called SSPSONN which is used to construct a soft-sensor of gasoline endpoint of crude distillation unit.

4.1. The SSPSONN algorithm

SSPSONN assumes the weight and threshold of NN as the position of particle of PSO, evaluates the fitness of particle according to objective function of system, and then searches for the global best weight and the global best threshold by PSO. If the search is accomplished, the position of the global best particle is the combination of the best weight and the best threshold of SSPSONN.

SSPSONN train algorithm can be summarized as follows:

Step1. Initialize the structure, activation function and objective function of SSPSONN.

Step2. Initialize the algorithm parameters of SSPSO.

Step3. Evaluate fitness of each particle and partition particles into different sun-swarms.

Step4. In each sub-swarm, update particles' velocities and positions by Eq.(1) and Eq.(2); update the individual best fitness and the individual best position of each particle; update the global best fitness and the global best position of the sub-swarm.

Step5. Shuffle sub-swarms after a defined generations to be a new swarm. If the stop condition is not satisfied, go to step 3.Otherwise, stop iterating and obtain the best weight and the best threshold from the global best position.

4.2. Soft-sensor of gasoline endpoint based on SSPSONN

The structure of SSPSONN for the soft-sensor of gasoline endpoint is 8-14-1. The activation function of neurons of SSPSONN takes sigmoid function.

The eight input signals of the soft-sensor include: the pressure of the top of the tower, the temperature of the top of the tower, the temperature of the reflux of the top of the tower, the ratio of the external reflux flow to the inlet flow, the pressure of the inlet feed, the temperature of the first atmospheric side-stream, the ratio of the outlet flow rate of the first atmospheric side-stream to the inlet flow rate, the heat removal of the first middle reflux to the enthalpy. The output signal of the soft-sensor is gasoline endpoint.

The objective function of the soft-sensing model can be expressed as follow:

$$\min E = \frac{1}{2}\sum_{kk}^{np}(t^{kk} - y^{kk})^2$$

Where t is the real value of gasoline endpoint, y is the estimated value of gasoline endpoint, kk is the serial number of samples, np is the total number of samples.

In searching for the global best weight and threshold of NN by SSPSO, population of swarm is set to 100; the number of sub-swarm is set to 4;the maximum number of generations is set to 4000; error limit of objective function is 0.18; c_1 and c_2 are set to 2.0; w is decreased linearly from 0.9 to 0.4.

4.3. Experimental results and comparison

This paper have trained the soft-sensing model by 110 groups of training sample data which is processed by error-detection, smoothing, normalization. After model learning and accounting, average variance is 0.8671° C; average value of error is 0.6858° C. This result shows that the fitness precision between the model's predictive values and real values is high. Then the soft-sensing model is examined by another 50 groups of examining sample data. The comparison between predictive values and real values is showed in Fig. 2.

In 50 groups of examining sample data, average variance is 0.8875°C; average value of error is 0.7001°C. These data and Fig2 show that the predictive precision between the model's predictive values and real values is high.

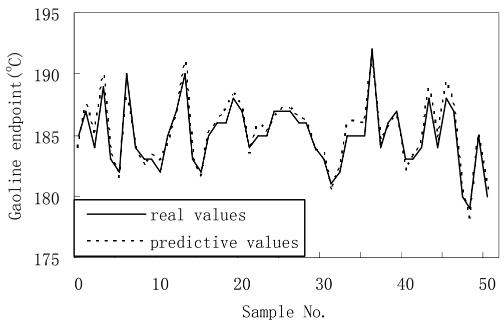

Fig2.The comparison between predictive values with SSPSONN and real values

5. Conclusions

This paper proposes an improved PSO algorithm—SSPSO and some benchmark problems are tested to evaluate the performance of it. The results show that it has better performance than SPSO and CPSO. The SSPSO is also applied to construct a soft-sensor of gasoline endpoint and the results show that this soft-sensor is feasible and effective.

Acknowledgements

This work is partially supported by National Science Fund for Distinguished Young Scholars of China(Grant No.60625302), Major State Basic Research Development Program of China (973 Program Grant No.2002CB312200), National High-Tech Research and Development Program of China (863 Program Grant No.20060104Z1081) and Major State Basic Research Development Program of Shanghai (Grant No. 05DJ14002).

References

[1] J. Kennedy, R.C. Eberhart, Particle swarm optimization. *Prof. of the IEEE International Conference on Neural Networks*, IEEE Press, pp. 1942–1948, 1995.

[2] X. Hu, Y.Shi, R.C. Eberhart, Recent advances in particle swarm. *Prof. of the IEEE International Conference on Evolutionary Computation*. Portland, pp.90-97, 2004.

[3] A. A. Adly, S. K. Abd-El-Hafiz, Field computation in non-linear magnetic media using particle swarm optimization. *Journal of Magnetism and Magnetic Materials*, pp. 690–692, 2004.

[4] J.P. Coelho, P.B. de M. Oliveira, J. B. Cunha, Greenhouse air temperature predictive control using the particle swarm optimisation algorithm. *Computers and Electronics in Agriculture*, 49:330–344, 2005.

[5] C. D. Bocaniala, J. S. da Costa, Application of a novel fuzzy classifier to fault detection and isolation of the DAMADICS benchmark problem. *Control Engineering Practice*, 14:653–669, 2006.

[6] S. Paterlini, T. Krink, Differential evolution and particle swarm optimisation in partitional clustering. *Computational Statistics & Data Analysis*, 50:1220 – 1247, 2006.

[7] R.C. Eberhart, Y. Shi, Particle swarm optimization: developments, applications and resources. *Proc. Of the IEEE International Conference on Evolutionary Computation*. Seoul, Korea, pp.81–86, 2001.

[8] P. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance difference. *Evolutionary Programming Ⅶ*, Springer, pp.601-610, 1998.

[9] F. van den Bergh, A. P. Engelbrecht, A study of particle swarm optimization particle trajectories. *Information Sciences*, 176: 937-971, 2006.

[10] M. Løbjerg, T.K. Rasmussen, K. Krink, Hybrid particle swarm optimizer with breeding and subpopulations. *Proc. of the Third Genetic and Evolutionary Computation Conference*, 1:469–476, 2001.

[11] K.E. Parsopoulos, M.N. Vrahatis, UPSO: A unified particle swarm optimization scheme. In: Lecture Series on Computer and Computational Sciences. Vol. 1, Proc.Int. Conf. Comput. Meth. Sci. Eng. (ICCMSE 2004), VSP International Science Publishers, Zeist, The Netherlands, pp. 868–873,2004.

[12] S.C. Chu, J.S. Pan, Intelligent Parallel Particle Swarm Optimization Algorithms. *Studies in Computational Intelligence*, 22: 159–175, 2006.

[13] A. Chatterjee, P. Siarry, Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. *Computers &Operations Research*, 33:859-871, 2006.

[14] B. Liu, L. Wang, Y. Jin, et al, Improve particle swarm optimization combined with chaos. *Chaos, Solitons and Fractals*, 25:1261-1271, 2005.

[15] D. Yi, X.R. Ge, An improved PSO-based ANN with simulated annealing technique. *neurocomputing*, 63:527-533, 2005.

[16] P. Yin, S. Yu, P. Wang, et al, A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems. *Computer Standards & Interfaces*, 28:441–450, 2006.

[17] Y.Shi, R.C. Eberhart, A modified particle swarm optimizer. *Prof of the IEEE International Conference on Evolutionary Computation*, NJ: IEEE Press, Piscataway, pp. 69–73, 1998.

[18] M. Clerc, J. Kennedy, The particle Swarm—explosion, stability and convergence in a multidimensional complex space. *IEEE Trans on Evolutionary Computation*, 6(1):58–73, 2002.