A Novel Scheme for Particle Swarm Optimization

He Wei[1,2,a], Xu Yuanming[1,b], Zhou yaoming[1,c], Meng zhijun[1,d], Li yuankai[2,e]

[1.] School of Aeronautic Science and Engineering, Beihang University, Beijing, China 100191

[2.] Aviation Theory Department, Aviation University of Airforce, Changchun, China 130022

[a]heweismiling@163.com [b]xuymg@sina.com [d]zhouyaoming204216@163.com [d]mengzhjun@buaa.edu.cn [e]lykok1@126.com

Keywords: Particle Swarm Optimization; Vanguard scheme; Global optimization; Local optimization; Severance rule

Abstract. Particle Swarm Optimization (PSO) is one of the most popular optimization algorithms, but it suffers from the contradiction between the exploitation ability and the exploration ability. This paper presents a scheme that introduced main swarm to search for global solution and sub-swarm which was named as vanguard swarm to search for accurate local solution, a severance rule was introduced to re-disperse the particles of vanguard swarm after the local search. The Particle Swarm Optimization with this novel search schem was called Vanguard Scheme Particle Swarm Optimization (VPSO). A comparison between the VPSO and the other 4 variants of PSO on 7 test functions indicates that the proposed algorithm can balance the contradiction between the exploitation ability and the exploration ability, and possesses a more broad application prospects.

Introduction

Particle Swarm Optimization [1] (PSO) was first introduced by Kennedy and Eberhart in 1995 based on swarm intelligent. PSO algorithm is a classic stochastic multi-point method, it does not need the derivative of the function to be optimized, and this makes the PSO possess a stronger global optimization capability than traditional optimization algorithms which need the information of derivative. PSO algorithm has simple principle、less control parameters, and it dose easy to be implemented, as the advantages above, the PSO algorithm attract the attention of many researchers, and was widely used to solve practical engineering problems [2,3,4]. A number of variants have been introduced to improve PSO algorithm, but there still some problems left, the most critical one is how to balance the contradictory between the exploitation ability and the exploration ability. The exploitation ability means to find an accuracy solution and the exploration ability means to find a solution more globally. To solve this contradiction, researchers did lot of improvements based on the Original PSO (OPSO) algorithm. Vanguard Scheme Particle Swarm Optimization (VPSO) was proposed in this paper, where a vanguard scheme based on local optimization was introduced to improve the exploitation ability of the algorithm, while the global optimization of main swarm and severance rule of vanguard particles combined together to enhance the exploration ability. VPSO was compared with other 4 PSO variants through 7 classic test functions, the result of experiments showed that the VPSO performs better both on exploitation ability and exploration ability.

This paper is organized as follows: Section II reviews the OPSO algorithm and the variants of it, highlights 4 variants which were used in the experiments. Section III, Introduces the basic ideas and the details of VPSO algorithm. Section IV, VPSO and other 4 PSO variants were tested, the results were analyzed. Section V summarizes the work and points out the direction of future works.

Overview of PSO

American social psychologist James Kennedy and electrical engineer Russell Eberhart introduced Particle Swarm Optimization (PSO) algorithm based on swarm intelligence in 1995. PSO solve the optimization problem by simulate the behavior of bird flocking when foraging. In PSO each bird is seemed as a particle and the whole flocking is a swarm. The position of i-th particle is $x_i{}^d = x_i^1, x_i^2, \cdots, x_i^n$, in which $i \in (1,m)$, m is the number of the particles of the whole swarm, $d \in (1,n)$, n is the number of the variables of the problem. The particles move in the design space with the velocity $V_i^d = v_i^1, v_i^2, \cdots v_i^n$, within every iteration, the particles update their positions and velocity by (1) and (2).

$$V^d{}_i(t+1) = V^d{}_i(t) + c_1 r_1 (P_i^d(t) - X_i{}^d(t)) + c_2 r_2 (G^d(t) - X_i{}^d(t)) \qquad (1)$$

$$X_i{}^d(t+1) = X_i{}^d(t) + V_i^d(t+1) \qquad (2)$$

In equation (1) and (2), t represents the iteration of last time and $t+1$ represents the iteration of current. $P_i(t)$ is the best position of i-th particle in the past t iterations, and it is usually called as personal experience. $G(t)$ is the best position has been found so far, it is the social experience of the whole swarm in all the past t iterations. Constant c_1 and c_2 are personal acceleration coefficient and social acceleration coefficient [5], these two coefficients influence the step length of the particles fly towards $P_i(t)$ and $G(t)$. r_1 and r_2 are random numbers both in the range of $[0,1]$, these two random coefficients represent the stochastic properties of PSO. We named the algorithm introduced by James Kennedy and Russell Eberhart as Original Particle Swarm Optimization (OPSO).

To improve the performance of PSO, researchers improved many variants on PSO. Krink [6] summarized all the parameters of PSO variants within the following equations.

$$V_i(t+1) = \chi(\omega \cdot V_i(t) + \phi_1(P_i - X_i) + \phi_2(G - X_i)) \qquad (3)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \qquad (4)$$

In equ（3） $\phi_1 = c_1 r_1$ and $\phi_2 = c_2 r_2$. Kennedy and Eberhart [1,7] chose $c_1 = c_2 = 2$ in their OPSO. Venter and Sobieski [5] set $c_1 = 1.5$ 、 $c_2 = 2.5$ and got good results. Yuhui Shi [8] introduced the concept of inertia weight ω， in the later study he introduced dynamic inertia weight [9], This PSO algorithm was called as Standard Particle Swarm Optimization (SPSO). Clerc and Kennedy[10] introduced the constriction factor χ .Kennedy [11,12]、 Suganthan [13]、Nasir [14] brought different kind of topologies into PSO algorithm. Jacques Riget [15] introduced Attractive and Repulsive PSO (ARPSO). Liang [16] Lovbjerg [17] and Blackwell [18] use multiple swarms method and proved that the cooperative optimizer was more efficiency than single swarm algorithms. Jun Sun [19,20] introduced the particles having quantum behavior PSO(QPSO), in which he recalled the velocity of particles. Angeline [21,22] and Higashi [23] incorporated the Genetic Algorithm (GA) with PSO. Lovbjerg [17] introduced Hybrid Particle Swarm Optimization (HPSO), after each iteration, a certain number of particles were chosen to execute the operation of breeding, and the offspring were used to replace the parents particles. Dou Quansheng [24] combined PSO algorithm with Simulated Anealing algorithm, Shelokar [25] hybrid the PSO algorithm with Ant Colony algorithm, both these two methods were efficient to improve the performance of PSO.

A Novel Scheme of Particle Swarm Optimization

A. *Introduction of Novel Scheme*

Kennedy and Eberhart [1] indicated that a high value of the personal acceleration coefficient, compared with the social acceleration coefficient, will enhance the exploration ability. In contrast, a relatively high value of the social acceleration coefficient may lead to premature, but the exploitation ability could be enhanced. The Novel scheme which proposed here was based on this idea: all the particles in design space were set as the main swarm, parameters which fit for global search were used in the search procession of main swarm; after each iteration of the main swarm, a certain number of particles near the optimum point currently were selected to composite a vanguard swarm, parameters fit for local search were used for the procession of vanguard swarm; the vanguard swarm search for local best in the space which covered by vanguard swarm. Vanguard swarm optimization is nested with main swarm optimization, that every search step of the main swarm followed by a complete optimization procession of vanguard swarm. The PSO based on novel scheme was called Vanguard Scheme Particle Swarm Optimization (VPSO).

Pseudo-code of VPSO：

```
Step1: Initialize the population.
Step2: Calculate fitness values of each particle.
Step3: Based on these values find the global best $G(t)$ and the personal best $P_i(t)$ for each particle.
Step4: Update velocity of each particle.
Step5: Update position of each particle.
Step6: Update fitness values of each particle.
Step7: Based on these values find the global best $G(t)$ and the personal best $P_i(t)$ for each particle.
Sub-step1: Chose the Vanguard particles
Sub-step2: Set the boundary of Vanguard particles’ search space
Sub-step3: Update velocity of each Vanguard particle.
Sub-step4: Update position of each Vanguard particle.
Sub-step5: Update fitness values of each Vanguard particle.
Sub-step6: Based on these values find the global best $G(t)$ and the personal best $P_i(t)$ for each Vanguard particle.
Sub-step7: Determine whether to stop the process.
No, jump to sub-step3.
Yes, jump to sub-step8.
Sub-step8: Stop the iteration and record the parameters.
Sub-step9: Severance the vanguard particles.
Sub-step10: Evaluate the vanguard particles that have been severance.
Sub-step11: Based on these values update the global best $G(t)$ and the personal best $P_i(t)$ for.
Step8: Determine whether to stop the process.
No, jump to step4.
Yes, jump to step9.
Step9: Stop the iteration and record the parameters.
```

B. *Details about VPSO*

1) *Equations*

The evolution equation contains inertia weight was selected as basic evolution equation of VPSO. The same evolution equations with different parameters were used in main swarm optimization and vanguard swarm optimization of VPSO. In the main swarm optimization we use inertia weight w_m and accelerate coefficients of c_{m1} and c_{m2}, the equations were expressed as bellow:

$$V_i(t+1) = w_m \cdot V_i(t) + c_{m1}r_1(P_i(t) - X_i(t)) + c_{m2}r_2(G(t) - X_i(t)) \tag{5}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{6}$$

The parameters of inertia weight and accelerate coefficients in vanguard swarm were w_v、c_{v1} and c_{v2}, the evolution equations were expressed as bellow:

$$V_i(t+1) = w_v \cdot V_i(t) + c_{v1}r_1(P_i(t) - X_i(t)) + c_{v2}r_2(G(t) - X_i(t)) \tag{7}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{8}$$

Based on this conclusion on accelerate coefficients in [1,26] we chose $\omega_m = 0.9$ 、 $c_{m1} = 2.5$ and $c_{m2} = 0.5$; $\omega_v = 0.4$ 、 $c_{v1} = 0.5$ and $c_{v2} = 2.5$. With these parameters the main swarm will search the whole space roughly to insure the exploration ability, the vanguard swarm will search the local space to insure the exploitation ability.

2) Selection of vanguard particles

The task of vanguard swarm is search for high-precision solution in the local space. The selection of the vanguard particles is significant to the performance of the VPSO algorithm. The selection of vanguard particles was based on the distance d_i between the current optimal value and the main swarm particles, we chose a certain number of particles with the smallest d_i . The maximum value of all the vanguard particles on each dimension were set as the search boundaries, within the boundaries is the search space of the vanguard swarm. In order to confirm the number of vanguard particles, a scale factor SF was set and the number of vanguard swarm $m_v = m_m \cdot SF$, m_m is the total number of the main swarm, and m_v is the number of vanguard swarm.

3) The severance rule

All the vanguard particles will gathered closely around the local optimal point of the vanguard search space after the local optimization of vanguard swarm. If these particles were applied to global search of main swarm directly, the diversity of the main swarm would be cut down, and the exploration ability will be weak. So these vanguard particles need to re-disperse into the design space. The re-disperse of the vanguard particles was called as severance, a severance rule should be followed. Due to the stochastic of the algorithm, we hope the particles could be dispersed in the design space without the local space which has been searched by the vanguard swarm. For the one-dimension design space as in Figure 1. One equation should be chosen between (9) and (10) when the vanguard particles were severance. If the first equation was chosen, the value of i-th particle will increase, and the point will randomly falls between the upper boundaries of vanguard space and the whole design space; when the second equation was chosen, the value of i-th particle will decrease and the point will randomly falls between the lower boundaries of vanguard swarm and whole design space. $rand(u)$ is a random number in the range of $[0,1]$.

$$x_i = vx_i + (range\max - vx_i) + (x\max - range\max)\cdot rand(u) \qquad (9)$$

$$x_i = vx_i + (range\min - vx_i) + (x\min - range\min)\cdot rand(u) \qquad (10)$$

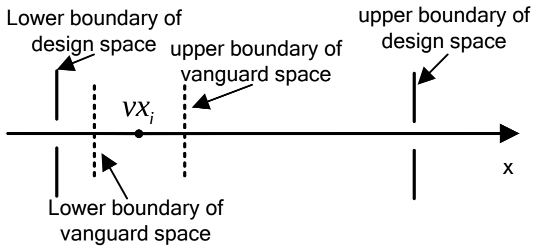

Lower boundary of design space
upper boundary of vanguard space
upper boundary of design space
vx_i
x
Lower boundary of vanguard space

Figure 1. Severance of one dissension problem

As shown in Figure 1, it's obviously that the value of i-th particle has a higher probability of being increased than being decreased. In order to reflect the difference of probability, we have designed a severance factor μ and a weight factor ω_c .

$$\mu = (rand(u) < \omega_c) \qquad (11)$$

$$\omega_c = \frac{x\max_j - range\max_j}{(x\max_j - range\max_j) + (range\min_j - x\min_j)} \qquad (12)$$

When $\mu = 1$, the equation (9) was chosen and the particle will moved to the upper boundary of the design space; when $\mu = 0$, the particle will move to the lower boundary of the design space, because the second equation was chosen. After being severance, the velocity of the vanguard particles were randomly reset, it's like that the vanguard particles were re-initialized. To increase the diversity of the whole swarm, the values $P_i(t)$ of all the vanguard particles were reset to. Because during the local optimization, all the vanguard particles may converged to a local best position, and $P_i(t)$ was replaced by the same local best position. In this condition, the diversity of the main swarm would be decreased and it's harmful for the performance of VPSO algorithm.

Experiments and results

The performance of the VPSO algorithm has been compared with the other 4 PSO-variants on 7 numerical test functions.

A. Experimental setup

1) Algorithms compared

The PSO variants were tested here include: OPSO [1]、SPSO [9]、ARPSO [15]、HPSO [17]. The settings of the parameters were listed in Table I.

2) Test functions

All the algorithms were tested on 7 standard test functions [26,27], the initial range was $[x\min, x\min + 0.1\times(x\max - x\min)]^n$. The parametric settings of all test functions were summarized in Table II.

3) Parameters compared

The comparison of different PSO variants was based on the analysis of data obtained from the experiments, the parameters were below:

Table I. Parameters settings of PSO variants

Algorithm	ω	c_1	c_2	$v_{\max}$	Others
OPSO		2.0	2.0	$0.2*range$	
SPSO	$0.9\sim0.4$	2.0	2.0	$0.2*range$	
ARPSO	$0.9\sim0.4$	2.0	2.0	$0.2*range$	$d_{low}=5\times10^{-6}$ $d_{high}=0.25$
HPSO	$0.9\sim0.4$	2.0	2.0	$0.2*range$	$pb=0.2$

Table II. Test functions and their parameters

Type	**name**	**Search range**	**Int_ range**	**G_best**	**F_min**	**Threshold**
unimodal	Sphere	$[-100,100]^n$	$[-100,-80]^n$	$[0]^n$	0	1×10^{-6}
	Rosenbrock	$[-10,10]^n$	$[-10,-8]^n$	$[1]^n$	0	100
multimodal	Schaffer	$[-100,100]^n$	$[-100,-80]^n$	$[0]^n$	0	1×10^{-6}
	Rastrigin	$[-5.12,5.12]^n$	$[-5.12,-4.096]^n$	$[0]^n$	0	100
	Ackley	$[-32,32]^n$	$[-32,-25.6]^n$	$[0]^n$	0	1×10^{-6}
	Griewank	$[-600,600]^n$	$[-600,-480]^n$	$[0]^n$	0	1×10^{-6}
	Schwefel	$[-500,500]^n$	$[-500,-400]^n$	$[420.96]^n$	0	2000

Average: mean of the final solution values over 100 independent runs for each algorithm.

STD（Standard Deviation）: standard deviation of the final solution values over 100 independent runs for each algorithm.

FEs（Function Evaluations）: the mean values of function evaluations over which acceptable solution has been found during the 100 independent runs for each algorithm.

SR（Success rate）: the ratio of runs out of the 100 independent runs of a function that have find the acceptable result.

SP（Success Performance）:FEs/SR%, it indicates the efficiency of an algorithm.

TFEs (Terminate Function Evaluations): the number of Function Evaluations which was set as stop conditions for each algorithm.

B. Comparison with other PSO variants

To make the comparison fair enough, Every test function was run for 100 independent times, and the experiment data were recorded. All the PSO-variants were made to run with the $TFEs = 500000$. All the functions were tested with the 30 dimensions. The results of SF value with 0.3 and 0.4 were used in this comparison, because the performance of VPSO was more stable under these two SF values. The details of experiments were summarized in Table III. Convergence characteristics of all PSO variants have been shown for 7 tests in Figure 2. The plots were based on the median run of each algorithm where the runs of the same algorithm are ranked according to their best solution values.

1) Comparison of the exploitation ability

Unimodal function：For the Sphere function，quite high precision of solutions were obtained when $SF = 0.3$, the value is 1.91^{-20}; when $SF = 0.4$, the solutions were not so precision as when $SF = 0.3$, but there were still much better than the solutions of the other algorithms. For Rosenbrock，VPSO performed not so good as in Sphere both with $SF = 0.3$ and $SF = 0.4$, but it still more accurate than the other PSO variants.

Multimodal functions: VPSO performed better on multimodal functions than the other PSO variants just like it performed on the unimodal functions. In Ackley and Griewank functions，VPSO with $SF = 0.4$ and $SF = 0.3$ both got high precision solutions, for Ackley function it achieve the order of 5.62^{-11}, for Griewank the solutions got the order of 1.06^{-2}and2.6^{-2}. In Schaffer, the algorithms of VPSO、SPSO and ARPSO get the global best position 0 and performed very brilliant exploitation ability. But the convergence characteristics in Figure 2 show that the VPSO has tremendous advantage in the aspect of converge speed. In Rastrigin, the difference in optimization accuracy was not

Table III. Comparison of experiment results

function		**VPSO**		**OPSO**	**SPSO**	**ARPSO**	**HPSO**
	SF	0.3	0.4				
	TFs	500000	500000	500000	500000	500000	500000
Sphere	SR	1	0	0	0	0	0
	Average	1.91E-20	8.50E-01	915.42	21.94	20.61	120.12
	STD	4.23E-21	1.14	157.64	8.52	8.68	19.63
	FEs	52625.22	NaN	NaN	NaN	NaN	NaN
	SP	52625.22	NaN	NaN	NaN	NaN	NaN
Rosenbrock	SR	1	0.99	0	0.54	0.41	0
	Average	3.53	2.66	1446.79	108.77	112.34	210.17
	STD	7.41	16.24	309.14	54.81	44.54	179.42
	FEs	18896.34	19254.55	NaN	512746.66	518700	NaN
	SP	18896.34	19449.04	NaN	949530.86	1265122	NaN
Schaffer	SR	1	1	0.02	1	1	0.26
	Average	0	0	1.78E-04	0	0	3.75E-06
	STD	0	0	2.89E-04	0	0	3.83E-06
	FEs	58954.08	61091.88	4855.9	153822.6	156516.6	468773.1
	SP	58954.08	61091.88	242795.5	153822.6	156516.6	1802973
Rastrigin	SR	1	1	0	0.94	0.9	0.01
	Average	52.99	60.95	154.53	71.96	73.56	164.19
	STD	12.42	15.42	18.51	18.4	19.88	30.96
	FEs	8304.42	19460.04	NaN	404261.48	395072.7	40643.07
	SP	8304.42	19460.04	NaN	430065.41	438969.6	4064308
Ackley	SR	1	1	0	0	0	0
	Average	5.62E-11	5.62E-11	7.58	6.007	6.29	19.98
	STD	1.91E-11	1.19E-11	4.63E-01	6.99	7.27	4.59E-02
	FEs	138787.9	146293.1	NaN	NaN	NaN	NaN
	SP	138787.9	146293.1	NaN	NaN	NaN	NaN
Griewank	SR	0.48	0.36	0	0	0	0
	Average	1.06E-02	2.60E-02	8.88	1.19	1.2	2.075
	STD	1.48E-02	3.81E-02	1.51	8.27E-02	9.06E-02	1.69E-01
	FEs	52487.5	56983.33	NaN	NaN	NaN	NaN
	SP	109349	158287	NaN	NaN	NaN	NaN
Schwefel	SR	0	0.53	0	0	0	0
	Average	5.50E+03	2.04E+03	6.97E+03	6.28E+03	6.25E+03	5.05E+03
	STD	707.4975	983.9145	6.09E+02	281.1824	295.4701	2.05E+02
	FEs	NaN	1.53E+05	NaN	NaN	NaN	NaN
	SP	NaN	2.89E+05	NaN	NaN	NaN	NaN

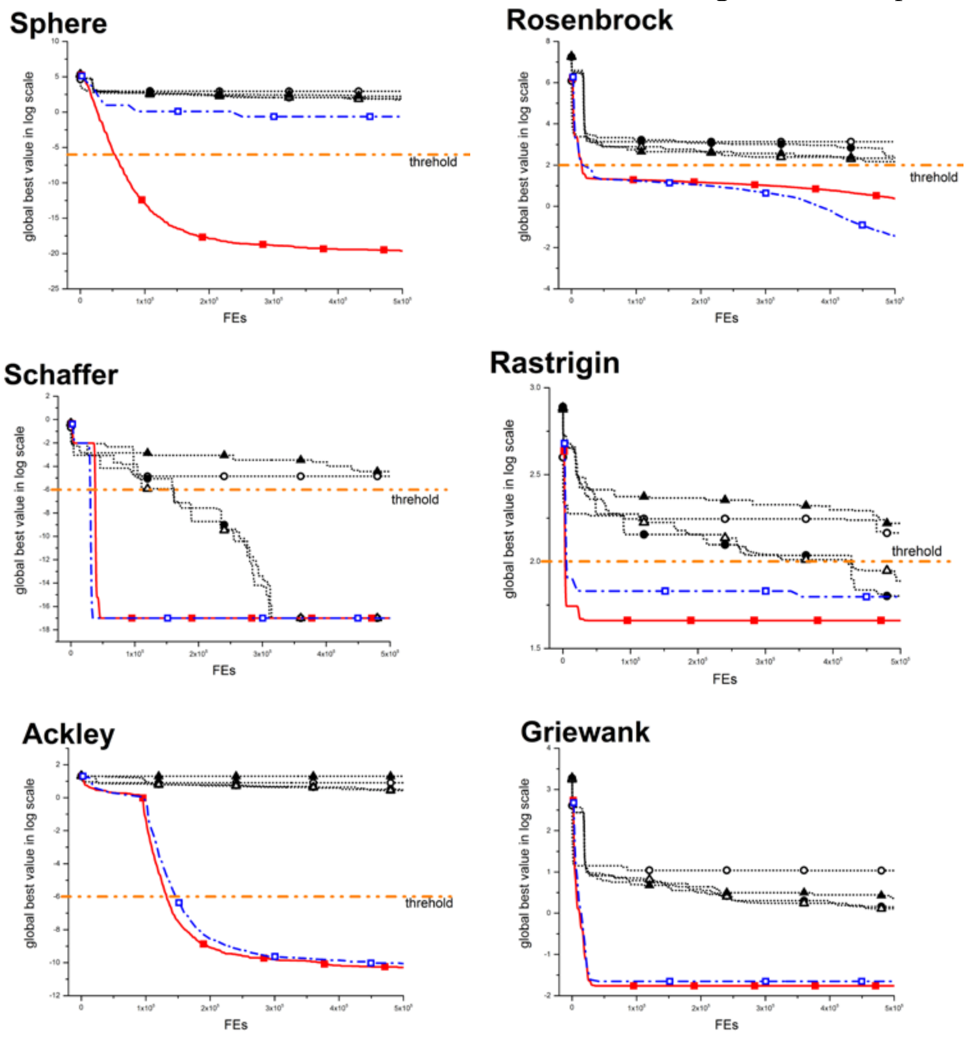
so obviously among VPSO, SPSO and ARPSO algorithms, but VPSO algorithm also showed an absolute advantage on convergence speed. In Schwefel, convergence speed and accuracy of VPSO was also superior to other algorithms, and the advantages on performance of VPSO algorithm was more obviously when $SF = 0.4$.

It has been shown that the exploitation ability of VPSO has been enhanced by the local optimization of vanguard swarm. The results of Sphere and Schwefel functions indicated that the value of SF in VPSO affects the performance of the algorithm.

2) *Comparison of the exploration ability*

The initial population was restrict within a small space, the algorithm will converge to a local best position if the exploration ability was not strong enough, the exploration ability could be tested by this way. The exploration ability was reflected directly by SR in Table II, the high SR value represents the high ability of global search and the high reliability of the algorithm. The success rate of VPSO was superior to the other PSO variants in both unimodal functions and multimodal functions, only performed poorly in Sphere when $SF = 0.4$ and Schwefel when $SF = 0.3$. This indicated that the SF influenced the performance of VPSO. The converge history of Sphere indicated that the value of global best is still decreased stepped. We believed that the algorithm will finally get the best value that meet the requirements, if the algorithm run enough iterations. Such conclusion was based on the global search mechanism of main swarm and the severance rule of the vanguard swarm. These two methods insured the particles search the whole design space during the whole iterations.

In the experiments of multimodal functions Schaffer, Rastrigin and Ackley, the success rate of VPSO was almost 100%. For Griewank and Schwefel the success rate also reached nearly 40%. For SPSO and ARPSO the values of SR were lower than VPSO. OPSO algorithm has a poor



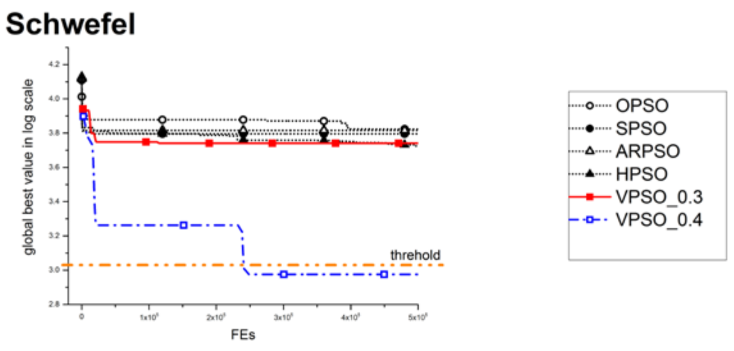


Figure 2. Sample convergence characteristics

performance with all test functions. For the HSPO algorithm, although the breeding operation was introduced to increase the diversity of the swarm, the promotion of performance was not obviously. In summary, the VPSO with global optimization method and Severance rule increased the diversity of the particles, and enhance the exploration ability effectively.

3) *Comparison of the search efficiency*

Unimodal function: for Sphere, only VPSO with $SF = 0.3$ found the available solution, where the SR=100% and the FEs=52625, all the other variants include the VPSO with $SF = 0.4$ did not find the available solution. For Rosenbrock, the SP values are 18896 and 19255 with $SF = 0.3$ and $SF = 0.4$ of VPSO. For SPSO and ARPSO algorithms the SP values are 949531 and1265122, they are much bigger than the SP values of VPSO. These SP values indicated that VPSO algorithm was more efficient than the other algorithms. The OPSO and HPSO algorithms haven't find the global best position.

Multimodal function: for Schaffer、Rastrigin、Ackley、Griewank and Schwefel VPSO with $SF = 0.3$ and $SF = 0.4$ found the global best point with very small SP, this indicated that the VPSO has high efficiency when dealing with multimodal functions. Well the other variants could not find the global best with comparatively small SP values or even could not find the global best till the algorithms stop.

Conclusions

Vanguard Scheme Particle Swarm Optimization (VPSO) is a new variant of PSO based on improvements in two aspects: in one hand a local optimization of vanguard swarm was nested in main optimize procession to enhance the exploitation ability; in the other hand, global search pattern and severance operation were introduced to guarantee the exploration ability. It has been proved by the experiments that the search efficient of VPSO was superior to the other PSO variants. The results of the experiments also proved that the VPSO algorithm was suitable for solving both unimodal and multimodal problems. Much future work remains to be conducted on the SF value, since the search ability of VPSO was influenced by the population number of the vanguard particles.

References

[1] Kennedy J, Eberhart R C. Particle Swarm Optimization [C] .Proceedings of IEEE International Conference on Neural Networks,1995:1942-1948.

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

[3] R. Poli, Analysis of the publications on the applications of particle swarm optimisation, J. Artif. Evol. Appl. (2008) 10, http://dx.doi.org/10.1155/2008/685175.

[4] A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization.Part I: background and development, Nat. Comput. 6 (4) (2007) 467–484.

[5] G. Venter, J.S. Sobieski, Particle swarm optimization, AIAA J. 41 (8) (2003) 1583–1589.

[6] Thiemo Krink,Jakob S.Vesterstrom and Jacques Riget.Particle Swarm Optimization with Spatial Particle Extension [C]. 2002:1474-1479.

[7] Y. Shi, R.C. Eberhart, Fuzzy adaptive particle swarm optimization, in: Proceedings of IEEE Congress on Evolutionary Computation, vol. 1, 2001, pp. 101–106.

[8] Shi Y,Eberhart R C.A modified particle swarm optimizer[C]. Proceedings of the IEEE International Conference on Evolutionary Computation,Piscataway,NJ:IEEE Press,1998:69-73.

[9] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in:Proceedings of IEEE Congress on Evolutionary Computation, July 1999,pp. 1945–1950.

[10] M. Clerc, J. Kennedy, The particle swarm-explosion, stability and convergence in a multidimensional complex space, IEEE Trans. Evolut. Comput. 6 (February (1)) (2002) 58–73.

[11] J. Kennedy, The particle swarm social adaptation of knowledge, in: Proceedings of IEEE International Conference on Evolutionary Computation, Indianapolis, IN, April 1997, pp. 303–308.

[12] J. Kennedy, R. Mendes, Population structure and particle swarm performance,in: Proceedings of IEEE Congress on Evolutionary Computation, Honolulu, HI,2002, pp. 1671–1676.

[13] P.N. Suganthan, Particle swarm optimizer with neighborhood operator,in: Proceedings of Congress on Evolutionary Computation, Washington, DC,1999, pp. 1958–1962.

[14] Md. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder, P.N. Suganthan, A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization, Inf. Sci. 209 (2012) 16–36.

[15] Jacques Riget, Jakob S.Vesterstrom.A Diversity-guided Particle Swarm Optimizer –the ARPSO[J].EVAlife Technical Report no.2002-02.

[16] J.J. Liang, P.N. Suganthan, Dynamic multiswarm particle swarm optimizer, in: Proceedings of Swarm Intelligence Symposium, June 2005, pp. 124–129.

[17] Lovbjerg M, Rasussen T K ,Krink T. Hybrid Particle Swarm Optimiser with Breeding and Subpopulations[C]. Proceedingsof the third Genetic and Evolutionary Computation Conferences, San Francisco, 2001 : 469-476.

[18] T. M. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments". LNCS No. 3005: Proceedings of Applications of Evolutionary Computing: EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoISAP, EvoMUSART, and EvoSTOC, Coimbra, Portugal. pp. 489-500, 2004.

[19] R.C.Eberhart,J.Kennedy,A new optimizer using particle swarm theory, in: Proceedings of 6th International Symposium on Micro Machine and Human Science, 1995, pp.39–43.

[20] Jun Sun,Bin Feng,Wenbo Xu.Particle swarm optimization with particles having quantum behavior[C].Proceedings of IEEE 2004 congress on evolutions.2002:325-331.

[21] Angeline P J.Using selection to improve particle swarm optimization[C].The 1998 IEEE International Congress on Evolutionary computation,1998:84-89.

[22] Angeline P J. Evolutionary Computation Versus Particle Swarm Optimization:Philosophy and performance Differences[J].Evolutionary Programming VII ,Lecture Notes in Computer Science.1998,1447:601-610.

[23] N. Higashi, H. Iba, Particle swarm optimization with Gaussian mutation, in: Proceedings of IEEE Swarm Intelligence Symposium, April 2003, pp. 72–79.

[24] Dou Quansheng, ZhouChunguang, Ma Ming. Two improvement strategies For particle swarm optimization[J].Journal of Computer Research and Development,2005,05:897-904.

[25] P.S. Shelokar, P. Siarry, V.K. Jayaraman, B.D. Kulkarni, Particle swarm and ant colony algorithms hybridized for improving continuous optimization, Appl.Math. Comput. 188 (May (1)) (2007) 129–142.

[26] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1997) 67–82.

[27] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evolut. Comput. 10 (June (3)) (2006) 281–295.

[28] Z.-H. Zhang, J. Zhang, Y. Li, Y. Shi, Orthogonal learning particle swarm optimization, IEEE Trans. Evolut. Comput. 15 (December (6)) (2011) 832–847.