Orthogonal Permutation Particle Swarm Optimizer with Switching Learning Strategy for Global Optimization

XIANGHUA CHU, QIANG LU
Shenzhen Graduate School
Harbin Institute of Technology
Shenzhen 518055
CHINA
xianghua.chu@gmail.com;
qiang.lu.home@gmail.com

BEN NIU
College of Management
Shenzhen University
Shenzhen 518060
CHINA
drniuben@gmail.com

Abstract: - This paper aims to improve the performance of original particle swarm optimization (PSO) so that the consequent method can be more robust and statistically sound for global optimization. A variation of PSO called the orthogonal permutation particle swarm optimization (OPPSO) is presented. An orthogonal permutation strategy, based on the orthogonal experimental design, is developed as a metabolic mechanism to enhance the diversity of the whole population, where the energetic offspring generated from the superior group will replace the inferior individuals. In addition, a switching learning strategy is introduced to exploit the particles' historical experience and drive individuals more efficiently. Seven state-of-the-art PSO variants were adopted for comparison on fifteen benchmark functions. Experimental results and statistical analyses demonstrate a significant improvement of the proposed algorithm.

Keywords: - Soft Computing; Particle swarm optimization; Orthogonal experimental design; Learning strategy; Global optimization;

1 Introduction

Global optimization problem extensively exists in real-world optimization fields, such as economic and engineering areas [1-3]. Since real-world problems become increasingly complex, superior techniques are always needed.

Particle swarm optimization (PSO), which was first proposed by Kennedy and Eberhart in 1995[4, 5], is a population-based stochastic optimization algorithm. The algorithm is inspired by the social choreography of birds flocking and fish schooling. PSO has been shown to be efficient and effective technique that it has been successfully applied to many practical areas, such as nonlinear constraint problems [6] and system design [7]. These optimization problems can be uniformly formulated as *D*-dimensional minimization problem as follows:

$$\text{minimize } f(x),\ x=[x_1,x_2,\ldots,x_D] \qquad (1)$$

Generally speaking, the advantages of PSO in addressing with global optimization problems include high convergence speed and easy implementation. However, it may easily be premature when solving complex problems. To improve the performance of PSO, many state-of-the-art PSO variants have been proposed [6, 8-16]. Qiao *et al*. [6] redefined personal best and global best to enhance the convergence rate of PSO. Jatmiko *et al*. [8] developed an modified PSO which follows a local gradient of the chemical concentration within a plume. Sabat *et al*. [11] integrated comprehensive learning strategy with hyper spherical manipulation to improve its exploration. A cooperatively coevolutionary algorithm based on the divide-and-conquer strategy has been incorporated into PSO [14], hence proposing two cooperative PSO models which offer remarkable results over the original PSO. Li and Yao [12] combined the cooperative coevolutionary model with random grouping and adaptive weighting schemes to develop a cooperatively coevolving PSO, which is promising on solving high-dimensional problem. What's more, Li and Yao adopted Cauchy and Gaussian distributions to sample new points and adaptively select the coevolving subcomponent sizes [13]. Liang *et al*. developed a comprehensive learning PSO (CLSPO) [9]. In CLSPO, instead of learning from the global and personal best information, a particle employs all the particles' best position to renew its velocity. CLPSO significantly enhances the performance of the original PSO on multimodal problems.

Orthogonal experimental design (OED) [17] is an efficient method to find out the best combination

levels of different factor without complete tests. More recently, several attempts try to incorporate the idea of OED into PSO, thus proposing orthogonal PSO (OPSO) [18] and orthogonal learning PSO (OLPSO) [19]. OPSO employs an "intelligent move mechanism" (IMM) to adjust a velocity for each particle, i.e., the cognitive learning and social learning components are operated by OED to determine the next move of particle. In OLPSO, the traditional PSO learning mechanism is replaced by an orthogonal learning strategy. The learning exemplar that constructed by OED is utilized to guide the exploration and exploitation of population. However, existing OED-based PSOs so depend on the operation of OED that increase the computation time of algorithm running. More exactly, OED operation generates a potential solution at the least cost of $2^{\lfloor \log_2(D+1) \rfloor}-1$ function evaluations (FEs). However, there is definite correlation between the performance of OED and PSO which also relies on computational time to evolve. Therefore, the motivation of this paper is to design a novel OED-based mechanism, which can fulfill OED's potential and maintain the convergence efficiency of PSO, to scale up PSO's performance for solving global optimization problems.

In this paper, the orthogonal permutation particle swarm optimizer (OPPSO) is proposed to improve PSO's performance on global optimization problems. An OED permutation strategy (OPS) is developed to update the relatively inferior individual and enhance the population's vigorousness. Instead of frequently running of the OED operation, the proposed method uses OED as a metabolic mechanism to take place of the losers with more superior ones. The losers with inferior fitness values are identified through a stochastic tournament. OPPSO also has a novel built-in learning paradigm, termed as switching learning strategy (SLS), which can prevent the solutions form falling into the local minimum and boost the global convergence ability.

The rest of the paper is organized as follows. Section 2 introduces and develops the OPPSO. Section 3 presents experimental setup, comparison results and discussion. Finally, concluding remarks are given in Section 4.

2. The proposed method

OPPSO integrates several novel strategies to enhance the search abilities and convergence speed of PSO. First, a brief description of OED [17]. Second, with the objective of improving diversity of the whole population, we adopt OED as metabolic mechanism to generate new position for the replacement of inferior particles, named as OED permutation in this paper. This strategy makes use of the advantages of orthogonal experiment to encourage the swarm to get rid of worse individuals while avoid extra consumption. Lastly, instead of using the traditional learning scheme, a switching learning strategy is proposed to accelerate the global convergence whilst prevent premature of algorithm.

2.1 OED technique

OED is an efficient method in identifying the best combination levels of different factor within reasonable experiments. We suppose an experiment that involves three factors which directly affect the outcomes, and each factor has three possible numbers. In order to seek the best combination of levels for a predefined objective function, we can use trail-and-error method to try every feasible combination, i.e., there are $3^3=27$ combinations of experimental designs. For an experiment with *N* factors and *Q* levels, the number of combinations is Q^N. Apparently, when the number of factors and levels are high, it is impossible to test all the combinations, which necessitates an efficient method. OED was introduced for this purpose[17]. It is capable of drawing a small but representative sample with the intention of reaching a dependable decision.

We let $L_M(Q^N)$ represent an orthogonal array (OA) for N factors and Q levels, where $M=2^{\lceil \log_2(N+1) \rceil}$ is the number of combinations of levels to be tested. There are M rows in OA. For convenience, we let $L_M(Q^N)=[a_{i,j}]_{M\times N}$ in which the *j*th factor in the *i*th combination has level $a_{i,j}$ and $a_{i,j}\in\{1,2,\ldots,Q\}$. An orthogonal arrays $L_9(3^4)$ is given by (2).

$$L_9(3^4)=\begin{bmatrix}1&1&1&1\\1&2&2&2\\1&3&3&3\\2&1&2&3\\2&2&3&1\\2&3&1&2\\3&1&3&2\\3&2&1&3\\3&3&2&1\end{bmatrix} \quad (2)$$

Note that $L_9(3^4)$ is the well-balanced subset of the complete factorial combinations. In an OA, each column represents one factor and the values in the columns stand for the levels of the factor. The number of columns also limits the maximum factors of a problem. For example, the OA $L_9(3^4)$ can be applied

to a problem with at most four factors.

Table 1 presents the processes for construction of a two levels OA for N factors [19]. There are basic columns and non-basic columns in OA, the basic column number is $u=\log_2(M)$ and the non-basic column number is $(M-1-\log_2(M))$.

Table 1 Procedure for OA construction

Algorithm 1 - Construction of OA
Initialization
for $i \leftarrow 1:M$, do
for $k \leftarrow 1:u$, do
$j \leftarrow 2^{k-1}$
$OA[i][j]=\left(\left\lfloor (a-1)/2^{u-k} \right\rfloor\right) \bmod 2$
if $j>1$ then
for $s \leftarrow 1:(j-1)$, do
$OA[i][j+s]=(OA[i][s]+OA[i][j]) \bmod 2$
and for s
end if
end for k
end for i
Output OA

After evaluation of the M combinations, factor analysis (FA) is used to estimate the main effect [20]. FA is capable of identifying better level for each factor through evaluation of individual factors on objective function.

Let f_m denotes the function value of the mth combination, where $m=1,\ldots,M$. Define the main effect of the factor n with level q as S_{nq} where $n=1,\ldots,N$ and $q=1,2$. $q=1,2$ is given as follows:

$$S_{nq}=\sum_{m=1}^{M} f_m \times z_m \qquad (3)$$

where $z_m=1$ if the mth experimental test is with the level q of the factor n, otherwise $z_m=0$. For a minimization optimization, the level 1 of factor n makes a better contribution to the objective function than level 2 of factor n does while $S_{n1}<S_{n2}$, and vice versa. If $S_{n1}=S_{n2}$, levels 1 and 2 make the same contribution. With the best levels of each factor are determined, a new combination with better contribution can be easily obtained. The new one potentially approaches the best one in the 2^N combinations.

2.2 OED permutation

We employ OED method to establish a particle permutation strategy which acts as an incentive mechanism. The main motivation for OED permutation strategy (OPS) is based on the principle that PSO is a cooperatively search algorithm, where the communication and interaction among particles guide the whole population to evolve.

There are a *superior set* and an *inferior set* that with k particles each (both excluding the *gbest*) in OPS. OPS combines information of *gbest* and $pbest_i$ of the selected particle i from the *superior set* to produce a promising particle. Each of the D dimensions is regarded as a factor and hence there are D factors with two levels per factor. With an OA of D factors, we build an OA $L_M(2^{M-1})$ with M rows and $M-1$ columns according to Table 1, where $M=2^{\lceil \log_2(D+1) \rceil}$ and $D+1 \le M \le 2D$. In our approach, we regard each dimension as one factor. The detailed steps of OPS are presented in Table 2.

Table 2 Process of OED permutation strategy

Algorithm 2 - OED permutation strategy
if *permutation criterion* met then
Step 1: *superior set* $\leftarrow$ *pbest*($\lceil rand1 \times ps \rceil$), …, *pbest*($\lceil randk \times ps \rceil$); *inferior set* $\leftarrow$ *pbest*($\lceil rand1 \times ps \rceil$), …, *pbest*($\lceil randk \times ps \rceil$);
Step 2: Execute *tournament selection* on *superior set* to select the superior particle p_i;
Step 3: Generate M solutions X_j $(1 \le j \le M)$ on basis of OA $L_M(2^D)$ which is generated by *Algorithm* 1;
Step 4: Evaluate X_j $(1 \le j \le M)$ and record the best solution X_b;
Step 5: Compute the main effect S_{nq}, where $n=1,\ldots,D$ and $q=1,2$;
Step 6: Determine the best level for each factor and form the potential solution X_p; Compare $f(X_b)$ with $f(X_p)$ and output the best solution X_{OA};
Step 7: Execute *tournament selection* for *inferior set* to select the worse particle p_w;
Step 8: Verify that $f(X_{OA})$ is superior to *Fitness*(*pbest*(p_w)), if it is true, use X_{OA} to take place of p_w, otherwise go to step 10;
Step 9: Update *pbest*(p_w) and *gbest*;
Step 10: Initialize *permutation criterion*.
end if

In Table 2, *permutation criterion* is used to judge whether the OPS is executed or not. To ensure the population benefiting from OED operation and to minimize the computational time, OPS will not be initiated until the *gbest* is stagnant for a certain number of generations called the trigger gap T. The appropriate selection of T value is given in Section 2.5.

2.3 Switching learning strategy

In our SLS, the velocity formula is modified as

$$V_i^d(t+1) \leftarrow w(t) \times V_i^d(t) + c_1 \times rand_i^d \ldots \times (pbest_f^d(t) - X_i^d(t)), \text{ if } rand > Pr \quad (4\text{-a})$$

$$V_i^d(t+1) \leftarrow \chi \times [V_i^d(t) + c_2 \times rand_i^d \ldots \times (gbest^d(t) - X_i^d(t))], \text{ if } rand \le Pr \quad (4\text{-b})$$

where $pbest_f^d$ represents the corresponding dimension of the *f*th particle's personal best which can be any particle's *pbest* including its own, Pr represents the learning rate of particle i that decides the *i*th particle's flying trajectory, and χ is constriction factor.

Prior to the calculation of velocity, a random number is generated to decide which equation is adopted for the *i*th particle in the *t*th generation. Formula (4-a) is similar to comprehensive learning strategy [9], and (4-b) is proposed in this paper that aims at improving population's global convergence ability by making use of the *gbest* information. In (10-b), particle only learns from *gbest* experience with the adjustment of χ.

Hsieh *et al*. [10] proposed a search range sharing (SRS) strategy to explore other particles' better solutions. Inspired by this work, we introduced a SRS-like operation to the (4-b) in OPPSO, and hence we formed the global attraction strategy (GAS). In GAS, when a particle is updated by (4-b), the next step is to perturb it and place it in the boundary between $[G_{\min}, G_{\max}]$, i.e., restrict a particle to exploit the range that near the *gbest*. To improve GAS's robustness and increase the probability of finding potential solutions, we set G as

$$G \leftarrow 0.6 \times (1 + rand) \times gbest \quad (5)$$

Therefore, a particle searches the space near global best to enhance exploitation ability.

2.4 OPPSO

The detailed procedure of OPPSO is presented in Table 3. As described above, the original PSO is modified by incorporating the OPS and the SLS.

Let R denotes the number of execution of OED. The time complexity of OPPSO is $ps + ps \times G + (M+1) \times R$ function evaluations, where *ss* is swarm size, G is the number of total generations. The complexity of original PSO is $ps + ps \times G$ function evaluations. Thus, the time complexity of OPPSO does not obviously increase when comparing with PSO.

Table 3 Procedure of OPPSO algorithm

Algorithm 3 - OPPSO algorithm
Initialize swarm size *ss*, T, Pr, w, and w
Initialize $[V_{\min}, V_{\max}]$ and $[X_{\min}, X_{\max}]$
Randomly initialize positions of all particles $X = (X_1, X_2, \ldots, X_{ss})$ with size of *ss*
Randomly initialize the velocity $V = (V_1, V_2, \ldots, V_{ss})$
Evaluate fitness values $F^0 = (f(X_1), \ldots, f(X_{ss}))$
Set $pbest \leftarrow (X_1, X_2, \ldots, X_{ps})$, and set *gbest*
for $t \leftarrow 1$: *MaxGeneration*, do
  for $i \leftarrow 1$: *ss*, do
    if $rand > Pr$ then compute (4-a)
      $X_i(t+1) \leftarrow X_i(t) + V_i(t+1)$
    else compute (4-b)
      $X_i(t+1) \leftarrow X_i(t) + V_i(t+1)$
      Compute $G_{\min}$ and $G_{\max}$ using Eq.(5)
      Perturb $X_i(t+1)$ between $G_{\min}$ and $G_{\max}$
    end if
    Update $pbest_i(t+1)$ and $gbest(t+1)$
    if $gbest(t+1) != gbest(t)$ then $mark = 0$
    else $mark = mark + 1$ end if
    if $mark > T$ then Executive *Algorithm* 2 end if
  if termination not met then $i \leftarrow i+1$ end if
  end for *i*
  if termination not met then $t \leftarrow t+1$ end if
end for *t*
Output results

2.5 Adjusting T and Pr

The trigger gap T and learning rate Pr need to be predefined. In order to study the impact of T and Pr, five benchmark functions are employed to carry out empirical studies. For convenience, the correlations between T and Pr are ignored, thus we set one as a fixed number when the other one is investigated. The OPPSO is independently run 25 times on each of these functions and the average values are shown as follows. First, for $Pr = 0.1$, we tested different values of T from 5 to 60 generations, and the results are plotted in Fig. 1. The curves indicate that OPPSO with a value of T around 35 offers better results. Second, we set $T = 35$ and study the suitable rate Pr, as shown in Fig. 2. We

observe that the better results for the functions are obtained when Pr is about 0.07. Therefore, the combination of $T=35$ and $Pr=0.07$ are adopted in the next experiments.

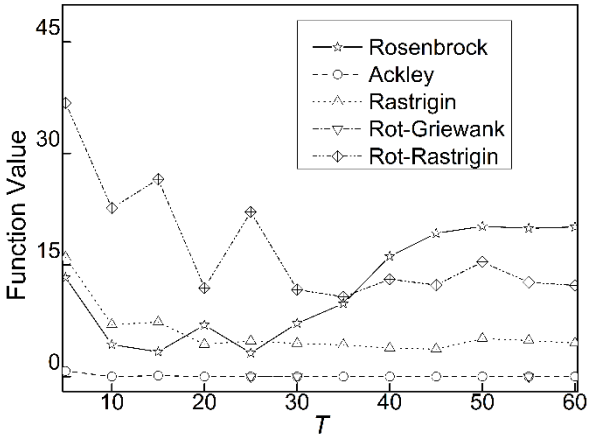

Fig. 1 OPPSO's performance with various T values when $Pr=0.1$

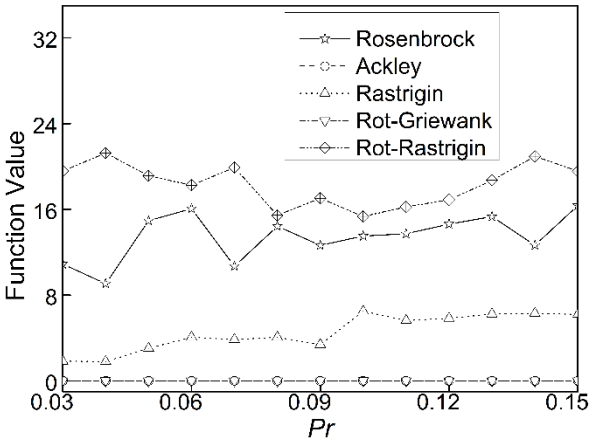

Fig. 2 OPPSO's performance with various Pr values when $T=35$

3. Experimental study and discussion

3.1 Benchmark functions

To test OPPSO's performance for various problems, fifteen benchmark functions including unimodal, multimodal, shifted and rotated problems [9, 21], are employed in our experiments. These functions, as listed in Table 4, are widely used for testing global optimization algorithms. From f_1 to f_9 are shifted unimodal and multimodal functions, and f_{10} to f_{17} are the rotated problems of f_2 to f_9. The rotated functions are constructed by left multiplying an orthogonal matrix [9].

Table 4 Test functions

Name	Benchmark function
Sphere	$f_1(x)=\sum_{i=1}^{D}x_i^2$
Rosenbrock	$f_2(x)=\sum_{i=1}^{D-1}(100(x_i^2-x_{i+1})^2+(x_i-1)^2)$
Ackley	$f_3(x)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})+20+e$ $-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))$
Griewank	$f_4(x)=\sum_{i=1}^{D}x_i^2/4000-\prod_{i=1}^{D}\cos(x_i/\sqrt{i})+1$
Rastrigin	$f_5(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$
Weierstrass	$f_6(x)=\sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right)$ $-D\sum_{k=0}^{k\max}(a^k\cos(2\pi b^k*0.5)),$ $a=0.5,\ b=3,\ k\max=20$
Schwefel	$f_7(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$
Quadric	$f_8(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2$
Rotated f_2	$f_9(y)=\sum_{i=1}^{D-1}(100(y_i^2-y_{i+1})^2+(y_i-1)^2),$ where $y=M*x$
Rotated f_3	$f_{10}(y)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}y_i^2})-$ $\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi y_i))$ $+20+e,$ where $y=M*x$
Rotated f_4	$f_{11}(y)=\sum_{i=1}^{D}y_i^2/4000-\prod_{i=1}^{D}\cos(y_i/\sqrt{i})$ $+1,$ where $y=M*x$
Rotated f_5	$f_{12}(y)=\sum_{i=1}^{D}(y_i^2-10\cos(2\pi y_i)+10),$ where $y=M*x$
Rotated f_6	$f_{13}(y)=\sum_{i=1}^{D}\sum_{k=0}^{20}\left(0.5^k\cos\left(2\pi 3^k(y_i+0.5)\right)\right)$ $-D\sum_{k=0}^{20}(0.5^k\cos(2\pi 3^k*0.5)),$ where $y=M*x$
Rotated f_7	$f_{14}(y)=\sum_{i=1}^{D}\lvert y_i\rvert+\prod_{i=1}^{D}\lvert y_i\rvert,$ where $y=M*x$
Rotated f_8	$f_{15}(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i}y_j)^2,$ where $y=M*x$

3.2 Algorithms for comparison and parameter settings

In order to verify the improvement of OPPSO, night state-of-the-art PSO variants are adopted to compare with OPPSO. For OPPSO, the parameters are $w\leftarrow 0.9\sim 0.4$, $\chi=0.7298$, $c_1=1.4945$ and $c_2=2.0478$, thus $\chi\times c_2\approx 1.4945$ [22], $k=3$. The algorithms adopted for comparison are shown in Table 5. The parameter settings of the benchmark algorithms are set as default values as recommended in the

corresponding references.

For fair comparison, experiments are carried out using the same swarm size of 30 and the maximum FEs is set to be 1.5×10^5. The dimension of test functions is set to be 30. Each algorithm is conducted 30 independent runs for each test function, and all the process data and final results are recorded for comparison.

Table 5 PSO variants for comparison

Algorithm		Reference
Abbr.	Full Name	
CPSO	Cooperative based PSO	[14]
CLPSO	Comprehensive learning PSO	[9]
FIPS	Fully informed particle swarm	[23]
OPSO	Orthogonal PSO	[18]
GPSO	Global topology based PSO	[22]
PSO-cf	PSO with constriction factor	[22]
UPSO	Unified PSO	[24]
OPPSO	The proposed method	—

3.3 Experimental results and discussion

3.3.1 Convergence characteristics and mean result

The convergence curves of different PSOs on representative functions in terms of mean best fitness over 150,000 FEs for 30 trials are presented in Fig. 3 - Fig. 6. From the figures, we observe that OPPSO offers better convergence rate for on most benchmark functions. For these functions, the curves of OPPSO decreased rapidly during the iterations, especially in the early period. However, CLPSO and CPSO provide faster convergence speed for function 5 followed by OPPSO. Overall, the proposed method has a strong and fast search ability for various problems, thus outperforms the compared algorithms on most functions in terms convergence rate.

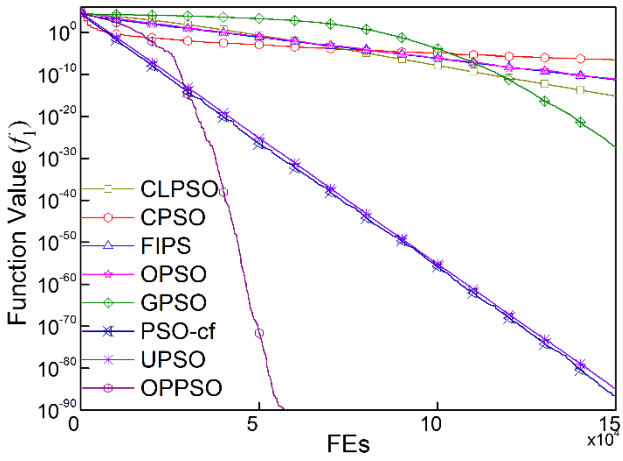

Fig. 3 Convergence curves on f_1

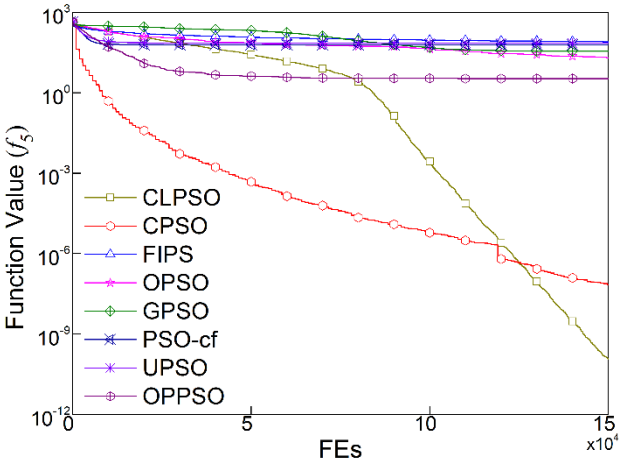

Fig. 4 Convergence curves on f_5

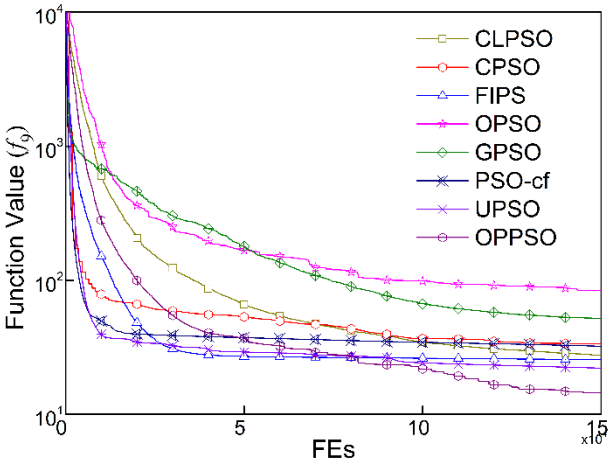

Fig. 5 Convergence curves on f_9

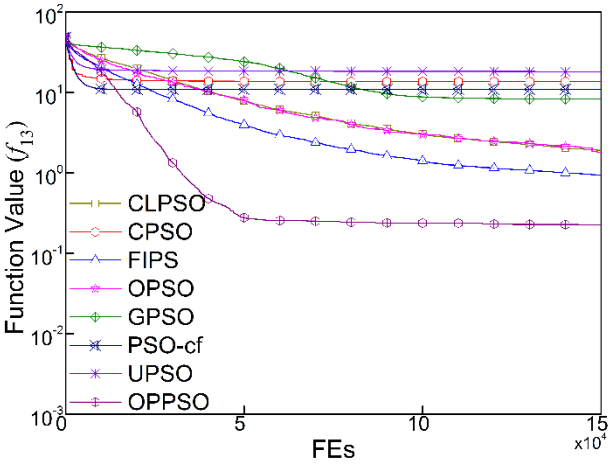

Fig. 6 Convergence curves on f_{13}

Table 6 presents the mean fitness values of the final solutions. The best means among the eight algorithms are highlighted in bold. From the results it can be

observed that the mean result of OPPSO performed better than the other PSO methods for the functions 1, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15, i.e., 13 out of the 15 functions. What is more, OPPSO significantly enhanced the performance on functions 6, 7, 8, 11, 14 and 15. CLPSO and CPSO outperform OPPSO on functions 5, which indicate their superior performance in solving multimodal functions. It can be observed that OPPSO outperforms the compared algorithms on most of the benchmark functions.

Table 6 Optimization results

	f_1	f_2	f_3	f_4
CLPSO	7.54E-16	2.10E+01	9.61E-09	5.54E-11
CPSO	1.11E-07	1.54E+01	9.57E-05	2.81E-02
FIPS	5.68E-12	2.45E+01	4.99E-07	7.61E-05
OPSO	4.98E-15	**1.20E+01**	1.18E-07	4.59E-16
GPSO	1.07E-28	2.98E+01	5.78E-14	2.25E-02
PSO-cf	2.63E-86	1.85E+01	1.61E+00	3.17E-02
UPSO	1.25E-85	1.62E+01	6.21E-02	6.57E-04
OPPSO	**0.00E+00**	1.55E+01	**3.08E-15**	**0.00E+00**
	f_5	f_6	f_7	f_8
CLPSO	**6.44E-11**	8.61E-10	1.85E-10	1.34E+03
CPSO	1.01E-07	4.80E-03	8.29E-05	1.35E+03
FIPS	7.87E+01	1.85E-01	7.91E-08	3.55E+02
OPSO	1.74E+01	1.61E-04	8.01E-09	4.84E-14
GPSO	3.92E+01	1.05E-01	4.79E-19	2.43E+01
PSO-cf	6.22E+01	5.09E+00	3.72E-29	2.24E-09
UPSO	6.58E+01	9.74E+00	7.15E-50	5.66E-04
OPPSO	1.29E+00	**0.00E+00**	**1.58E-110**	**1.11E-66**
	f_9	f_{10}	f_{11}	f_{12}
CLPSO	2.76E+01	5.81E-05	4.58E-06	4.25E+01
CPSO	3.36E+01	2.33E+00	3.64E-02	1.03E+02
FIPS	2.56E+01	6.82E-07	2.13E-03	1.39E+02
OPSO	8.41E+01	2.53E-05	2.24E-08	**2.00E+01**
GPSO	5.17E+01	1.35E+00	1.45E-02	7.82E+01
PSO-cf	3.22E+01	2.23E+00	1.38E-02	8.08E+01
UPSO	2.21E+01	2.06E-01	3.86E-03	7.22E+01
OPPSO	**1.44E+01**	**3.32E-15**	**0.00E+00**	**2.00E+01**
	f_{13}	f_{14}	f_{15}	
CLPSO	1.92E+00	1.04E-03	1.99E+03	
CPSO	1.37E+01	8.16E+00	1.58E+03	
FIPS	9.47E-01	7.67E-07	2.76E+02	
OPSO	1.79E+00	4.31E-05	6.81E-03	
GPSO	8.27E+00	2.57E-01	3.24E+01	
PSO-cf	1.10E+01	3.76E-01	1.67E-08	
UPSO	1.82E+01	6.52E-02	1.95E-03	
OPPSO	**2.28E-01**	**6.27E-137**	**3.37E-80**	

3.3.2 Solution distribution and statistical test

The final solutions obtained by each algorithm over 30 trials on 15 tested functions are used as samples to justify the performance of all approaches. Fig. 7 - Fig. 10 present the box plots of the optimization results for several representative functions. From the figures, it can be observed that for most of the functions, the final solutions of OPPSO converged to the fine and narrow areas, which means OPPSO has robust performance for different problems and is capable of generating solutions with superior quality when comparing with the other algorithms.

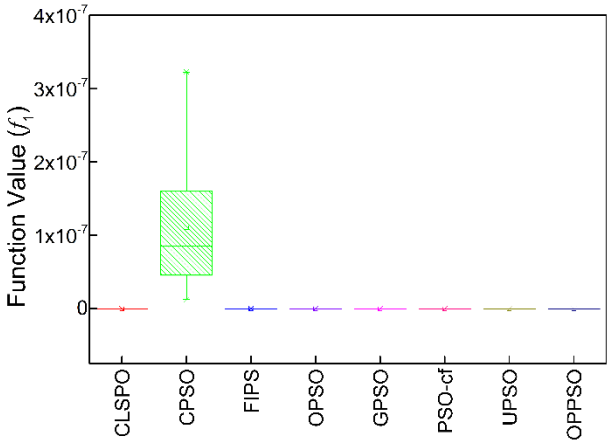

Fig. 7 Boxplot for the test results on f_1

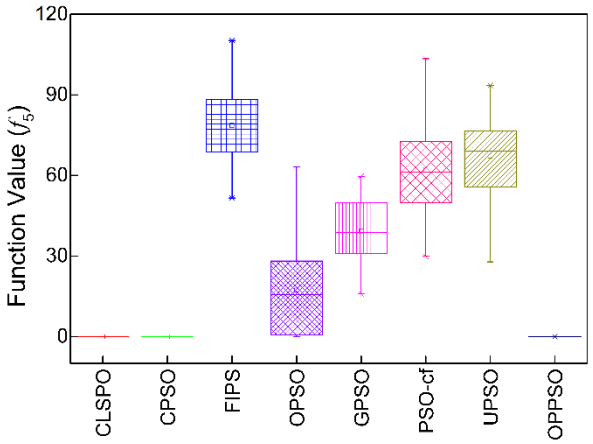

Fig. 8 Boxplot for the test results on f_5

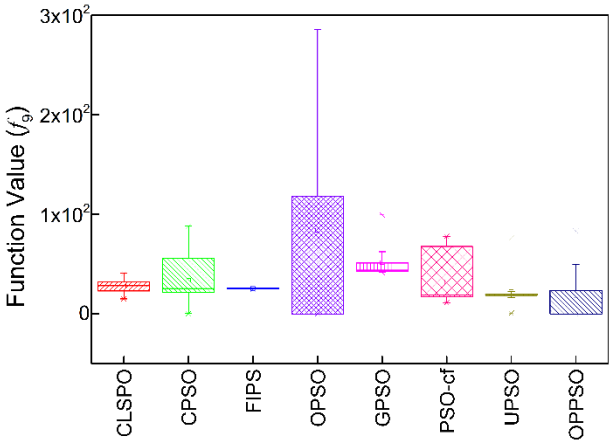

Fig. 9 Boxplot for the test results on f_9

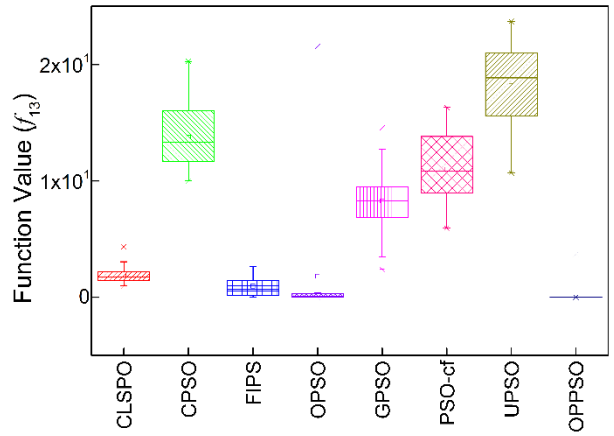

Fig. 10 Boxplot for the test results on f_{13}

In order to measure the statistical significance of results between OPPSO and the other algorithms, a set of one-tailed *t*-tests with 58 degrees of freedom at a 0.05 level of significance were carried out. In Table 7, the *t*-test results regarding A versus B are denoted as 's+', '+', '=', '-', 's-' when A is significantly better than, insignificantly better than, equal to, insignificantly worse than, and significantly worse than B respectively. For all the statistical analysis, the level of significance is set to be $\alpha = 0.05$.

From Table 7, OPPSO outperforms the other algorithms on functions 1, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15. The exception is function 5, where OPPSO was significantly worse than the first algorithms. For function 12, OPPSO obtained the same best results as OPSO, and they both are much better than the other variants. Although OPPSO numerically performed worse than OPSO on functions 2, the *t*-test results indicate that the difference is not statistically significant.

Table 7 *t*-test results for comparing OPPSO with other algorithms

	OPPSO vs.						
	CLPSO	CPSO	FIPS	OPSO	GPSO	PSO-cf	UPSO
f_1	s+	s+	s+	s+	s+	s+	s+
f_2	s+	s+	s+	-	s+	s+	s+
f_3	s+	s+	s+	s+	s+	s+	s+
f_4	s+	s+	s+	s+	s+	s+	s+
f_5	s-	s-	s+	s+	s+	s+	s+
f_6	s+	s+	s+	s+	s+	s+	s+
f_7	s+	s+	s+	s+	s+	s+	s+
f_8	s+	s+	s+	s+	s+	s+	s+
f_9	s+	s+	s+	s+	s+	s+	s+
f_{10}	s+	s+	s+	s+	s+	s+	s+
f_{11}	s+	s+	s+	s+	s+	s+	s+
f_{12}	s+	s+	s+	=	s+	s+	s+
f_{13}	s+	s+	s+	s+	s+	s+	s+
f_{14}	s+	s+	s+	s+	s+	s+	s+
f_{15}	s+	s+	s+	s+	s+	s+	s+

The final results of fitness values are employed to conduct Wilcoxon test to justify whether the proposed algorithm is significantly different from the comparing algorithms. The results are shown in Table 8. *R*+ and *R*- in Table 8 denote the sum of ranks for the functions where OPPSO outperforms and underperforms the compared algorithm. It can be observed that OPPSO significantly performs better than the other algorithms during the experiment.

Table 8 Results of Wilcoxon test

OPPSO vs.	*R*+	*R*-	*p*-value
CLPSO	111	9	0.003772
CPSO	105	15	0.010594
FIPS	120	0	0.000655
OPSO	106.5	13.5	0.011008
GPSO	120	0	0.000655
PSO-cf	120	0	0.000655
UPSO	120	0	0.000655

3.3.3 Discussion

Based on the experiments and comparisons, it can be concluded that the proposed strategies, OED permutation and switching learning strategy, greatly improve the performance of PSO and makes OPPSO surpass the compared PSO variants in terms of solution accuracy and convergence rate. OPPSO provides better adaptability for unimodal, multimodal, shifted and rotated problems due to the introduction of OPS and SLS.

In view of PSO's dimension-wise updating rule and OED's divide-and-conquer characteristic, we introduced OED as a population improvement mechanism to monitor the evolution and provide the

swarm with better individuals. In addition, the proposed SLS offers OPPSO better abilities of exploration and exploitation which guide the particle to search the promising region found by *gbest*, resulting in better convergence speed while maintain efficient exploration.

4. Conclusion

In this paper, we present an orthogonal permutation particle swarm optimizer that aiming at strengthening the performance of PSO on global optimization problems, such as unimodal, multimodal and rotated problems. The proposed OED permutation and switching learning strategies significantly improve the entire performance of the population and help individuals find the potential solutions easily. OPPSO makes full use of the particles' historical experience to enhance exploration and exploitation, e.g., substituting the superior offspring for the inferior particles and exploiting the promising space near the *gbest*.

Seven state-of-the-art PSO variants are used for comparisons, and comprehensive experiments have been carried out on 15 benchmark functions, including unimodal, multimodal and rotated problems. The experimental results verify that OPPSO statistically performs better than the other algorithms on most of the problems in terms of solution quality and convergence speed.

Inspired by the efficient integration of OED in this work, we are aiming to study more intelligent strategies to improve the efficiency of information utilization for the population. Besides, applying the OPPSO to more real-world problems is also our interest.

Acknowledgements: This work was supported by the National Natural Science foundation of China (Grants No. 71171064 and 71271140).

Reference:

[1] F. Neri, Quantitative estimation of market sentiment: A discussion of two alternatives, *WSEAS Transactions on Systems*, Vol. 11, No. 12, 2012, pp. 691-702.

[2] L. Li, X. Chu, and L. Ma, Research on the Administrative Cost Control Model Based on the Principal-Agent Theory, *Proceedings of the 15th International Conference on Industrial Engineering and Engineering Management*, pp. 444-447, Zhengzhou, China, 2008.

[3] F. Neri, Empirical investigation of word-of-mouth phenomena in markets: A software agent approach, *WSEAS Transactions on Computers*, Vol. 4, No. 8, 2005, pp. 987-994.

[4] J. Kennedy, and R. Eberhart, Particle swarm optimization, *Proceedings of IEEE International Conference on Neural Networks Proceedings*, pp. 1942-1948, Piscataway, NJ, USA, 1995.

[5] R. Eberhart, and J. Kennedy, A new optimizer using particle swarm theory, *Proceedings of the sixth International Symposium of Micro Machine and Human Science*, pp. 39-43, Nagoya, Japan, 1995.

[6] B. Qiao, X. Chang, M. Cui *et al.*, Hybrid particle swarm algorithm for solving nonlinear constraint optimization problems, *WSEAS Transactions on Mathematics*, Vol. 12, No. 1, 2013, pp. 76-84.

[7] H.-C. Chen, D.-H. Guo, and H.-M. Feng, Fuzzy embedded mobile robot systems design through the evolutionary PSO learning algorithm, *WSEAS Transactions on Systems*, Vol. 10, No. 8, 2011, pp. 259-269.

[8] W. Jatmiko, P. Mursanto, B. Kusumoputro *et al.*, Modified PSO algorithm based on flow of wind for odor source localization problems in dynamic environments, *WSEAS Transactions on Systems*, Vol. 7, No. 2, 2008, pp. 106-113.

[9] J. J. Liang, A. K. Qin, P. N. Suganthan *et al.*, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 3, 2006, pp. 281-295.

[10] S. T. Hsieh, T. Y. Sun, C. C. Liu *et al.*, Efficient Population Utilization Strategy for Particle Swarm Optimizer, *IEEE Transactions on Systems Man and Cybernetics--Part B: Cybernetics*, Vol. 39, No. 2, 2009, pp. 444-456.

[11] S. L. Sabat, L. Ali, and S. K. Udgata, Integrated Learning Particle Swarm Optimizer for global optimization, *Applied Soft Computing*, Vol. 11, No. 1, 2011, pp. 574-584.

[12] X. D. Li, and X. Yao, Tackling High Dimensional Nonseparable Optimization Problems By Cooperatively Coevolving Particle Swarms, *IEEE Congress on Evolutionary Computation*, pp. 1546-1553, 2009.

[13] X. D. Li, and X. Yao, Cooperatively Coevolving Particle Swarms for Large Scale Optimization, *IEEE Transactions on Evolutionary Computation*, Vol. 16, No. 2, 2012, pp. 210-224.

[14] F. Van Den Bergh, and A. P. Engelbrecht, A cooperative approach to particle swarm optimization, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, 2004, pp. 225-239.

[15] X. Chu, Q. Lu, B. Niu *et al.*, Solving the distribution center location problem based on multi-swarm cooperative particle swarm

optimizer, *8th International Conference on Intelligent Computing Technology*, pp. 626-633, Huangshan, China, 2012.

[16] X. Chu, Q. Lu, and B. Niu, Improved PSO-based algorithm for the capacitated location problem of distribution center, *Computer Engineering and Applications*, Vol. 49, No. 7, 2013, pp. 16-19.

[17] D. C. Montgomery, Design and Analysis of Experiments, 3rd ed., New York: Wiley, 1991.

[18] S. Y. Ho, H. S. Lin, W. H. Liauh *et al.*, OPSO: Orthogonal particle swarm optimization and its application to task assignment problems, *IEEE Transactions on Systems Man and Cybernetics--Part A: Systems and Humans*, Vol. 38, No. 2, 2008, pp. 288-298.

[19] Z. H. Zhan, J. Zhang, Y. Li *et al.*, Orthogonal Learning Particle Swarm Optimization, *IEEE Transactions on Evolutionary Computation*, Vol. 15, No. 6, 2011, pp. 832-847.

[20] S. Y. Ho, L. S. Shu, and J. H. Chen, Intelligent evolutionary algorithms for large parameter optimization problems, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 6, 2004, pp. 522-541.

[21] X. Yao, Y. Liu, and G. M. Lin, Evolutionary programming made faster, *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 2, 1999, pp. 82-102.

[22] J. Kennedy, R. Eberhart, and Y. Shi, Swarm Intelligence: San Mateo, CA: Morgan Kaufmann, 2001.

[23] R. Mendes, J. Kennedy, and J. Neves, The fully informed particle swarm: Simpler, maybe better, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, 2004, pp. 204-210.

[24] K. E. Parsopoulos, and M. N. Vrahatis, UPSO: a unified particle swarm optimization scheme, *Proceedings of International Conference on Computational Methods in Sciences and Engineering*, pp. 868-873, Zeist, Netherlands, 2004.