2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science

An Improved Cooperative RDPSO Algorithm Based on Search History Decision

Ji Zhao[1,2]
1 School of IoT Engineering
WuXi City College Of Vocational Technology
Wuxi, China
2 School of IoT Engineering
Jiangnan University
Wuxi, China
e-mail:queenji97@sohu.com

Yi Fu
School of IoT Engineering
WuXi City College Of Vocational Technology
Wuxi, China
e-mail:fy_bright@yahoo.com

Juan Mei
School of IoT Engineering
WuXi City College Of Vocational Technology
Wuxi, China
e-mail:meijuanwx@gmail.com

***Abstract*—A novel cooperative RDPSO algorithm based on entire search history decision (CRDPSO) is reported. The proposed algorithm employs a binary space partitioning tree structure to memorize the positions and the fitness values of the evaluated solution. Benefiting from the space partitioning scheme, a fast fitness function approximation using the archive is obtained. The approximation is used to improve the mutation strategy in CRDPSO. The resultant mutation is adaptive and parameter-less. Meanwhile, the cooperation mechanism enhanced search ability helps to prevent premature convergence and improve optimization. The proposed algorithm is examined on 8 standard benchmark functions including multimodal and unimodal functions. The results show that CRDPSO outperforms four traditional algorithms.**

Keywords: random drift particle swarm optimization; cooperative evolution; search history; adaptive mutate;

I. Introduction

Particle Swarm Optimization (PSO) algorithm is an evolutionary optimization technique originally introduced by Kennedy and Eberhart in 1995[1]. Instead of using evolutionary operations, such as crossover and mutation, in other evolutionary algorithms, PSO relies on the exchange of information between individuals, which are volume-less particles of the population called swarm. Each particle flies in search space with a velocity, which is dynamically adjusted according to its own flying experience and its companions' flying experience.

Since 1995, many attempts have been made to improve the performance of the PSO. As far as the PSO itself concerned, however, it is not a global optimization algorithm, as has been demonstrated by F. van den Bergh[2]. To overcome this flaw, Jun Sun et al has proposed a variant of the PSO algorithm, called random drift particle swarm optimization (RDPSO)[3], which was inspired by free electron model of metal conductors in an external electric field[4]. It has been shown that RDPSO achieve the goal of effectively estimating the parameters of complex biochemical dynamic systems and obtain the solution of better quality than most of the global optimization method used for solution to the inverse problems[3]. Like other evolutionary algorithm, PSO as well as RDPSO, confront the problem of premature convergence and stagnate at local optimum, which results in great performance loss and sub-optimal solutions. In order to further improve RDPSO's performance on complex optimization problems, it is important to increase the diversity of particles. A novel cooperative random drift particle swarm optimization algorithm based on entire search history (CRDPSO) is proposed.

Several search algorithms[5-7] employ search history in the form of memory to adaptively guide the search strategies. However, they only use partial search histories – that is, only part of the information gained from the search is retained and the rest are discarded. Search history, including the performed operations, the positions of the evaluated solutions and the fitness values of the solutions, are valuable information to enhance the performance of a swarm intelligent algorithm (SI). Intuitively, it can be used to maintain diversity. It can also guide the search direction or suggest promising search regions of interest. In addition, when the same optimum reappears in the search history, it can warn that the search may have trapped in a local optimum. The non-revisiting genetic algorithm (cNrGA) is proposed by Yuen and Chow[8]. It is originally applied to genetic algorithm (GA) to prevent from solution re-evaluation. Meanwhile, the scheme also acts as a parameter-less mutation operation. The cNrGA is found to be more robust than GA. However, the adaptive mutation of cNrGA could not make gene fully explore due to sub-regions limitations. Moreover, because of the complexity of the GA algorithm, cNrGA has complex computation and slow convergence speed. Since all updated positions in CRDPSO are guaranteed to be novel (they are non-revisited before), faster convergence speed is expected. Due to the nature of the non-revisiting scheme, the non-revisited position self-switch local search and global search.

The rest part of the paper is organized as follows. Section 2 introduces the RDPSO. Section 3 describes the details of CRDPSO. Section 4 is dedicated to application over the benchmark function minimization. Section 5 reports the simulation results where the experimental environment and the experimental results are elaborated and discussed. Finally, conclusions are given in Section 6.

II. RANDOM DRIFT PARTICLE SWARM OPTIMIZATION

Trajectory analysis[9] demonstrated the fact that convergence of the PSO algorithm may be achieved if each particle converges to its local attractor p_i^k. The particle's directional movement toward is similar to the drift motion of an election in metal conductors in an external electric field. However, according to the free electron model [10], besides the directional motion caused by the electric field, the electron is also in thermal motion which appears to be random movement. The overall effect of the electron's movement is that the electron careens towards the location of the minimum potential energy. Therefore it is obvious that the movement of the electron is very similar to the process of finding the minimum solution of a minimization problem, if the position of the electron is considered as a candidate solution and the potential energy function as the objective function of the problem.

Inspired by the above facts, [3] assumes that the particle in PSO behaves like an electron moving in a metal conductor in an external electric field. Therefore the movement of the particle is the superposition of the thermal motion and the directional motion to p_i^k. Accordingly, the velocity of the particle can be represented by $V_{i,j}^{k+1} = V1_{i,j}^{k+1} + V2_{i,j}^{k+1}$, where $V1_{i,j}^{k+1}$ and $V2_{i,j}^{k+1}$ represents the velocities of the thermal motion and directional motion toward p_i^k., respectively. Therefore, the velocity of the particle is determined by the combination of $V1_{i,j}^{k+1}$ and $V2_{i,j}^{k+1}$. Next, an adaptive strategy is adopted to evaluate $\sigma_{i,j}^k$ by $\sigma_{i,j}^k = 2\alpha|C_j^k - X_{i,j}^k|$, where $C^k = (C_1^k, C_2^k, \ldots, C_D^k)$ is called mean best (*mbest*) position defined as the mean of the *pbest* positions of all the particles, i.e. $C_j^k = (\frac{1}{m})\sum_{i=1}^{m} p_{i,j}^k \; (1 \le j \le D)$. Therefore a novel set of update equations for the particle is obtained in RDPSO:

$$V_{i,j}^{k+1} = \alpha \,|\, C_j^k - X_{i,j}^k \,|\, \varphi_{i,j}^k + \beta(p_{i,j}^k - X_{i,j}^k) \qquad (1)$$

$$X_{i,j}^{k+1} = X_{i,j}^k + V_{i,j}^{k+1} \qquad (2)$$

where α is called compression-expansion coefficient and β is called acceleration coefficient. The PSO with Equations (1) and (2) is named as random drift particle swarm optimization (RDPSO).

III. COOPERATIVE RDPSO ALGORITHM BASED ON ENTIRE SEARCH HISTORY

A. Entire search history scheme

Definition 1: The Sub-region *r* of solution *x*

Suppose x is a solution in the search space S, i.e. $x \in S$, and S is partitioned into sub-region set $R = \cup_i r_i$ by a binary space portioning (BSP) tree T. The sub-region $r \in R$ is defined as the 'the sub-region of x' if $x \in r$ and r is represented by a leaf node of T.

The entire search history recorder scheme stores all visited solutions and its fitness $[e_i, f(e_i)]$ by a tree-structure archive, namely BSP tree. During iterations, the search space is being partitioned into set of regions R. In the BSP tree, a node represents a region of search space. Suppose a parent node P has two child nodes m and n. The child nodes linearly partition the sub-region of P into two overlapped sub-regions. The corresponding partitioning cuts along the k^{th} dimension where $k = \arg max|m(k) - n(k)|$. In this way, each previous solution generated by the RDPSO is recorded in a node of the tree. In the whole solution process the BSP tree records all solutions in the search space and analyzes the current solution by evolutionary search history of prior solutions. Since the tree construction depends on the sequence of solution set, the BSP tree is a random tree and its topology is different from trial to trial.

The node in the BSP tree linearly partition the adjacent sub-regions have certain overlap to each other, namely overlapped sub-regions. It creates a path for which particle could move from one sub-region to another with better fitness. The search for the overlapped sub-region r of a solution x is implemented as a tree node search. The search starts from examining the root node whilst r is initialized as the whole search space. Each time the search moves downwards, r is contracted along a specified direction until the search reaches leaf node. The sub-region contraction scheme guarantees that the resultant sub-region overlaps with all its adjacent sub-regions.

Consider the fitness values of the evaluated solutions are also recorded in the memory archive by the BSP tree; this archive can also be regarded as the approximation $\tilde{f}(x)$ of a fitness function $f(x)$. If the sub region of an evaluated solution e_i is r_i, the fitness value of an unseen solution x inside the sub-region r_i can be approximated as y_i i.e. $f(x) \approx \tilde{f}(x) = y_i = f(e_i)$. $\tilde{f}(x)$ is a step-wise function because of the fitness of all solutions in the sub-region of a BSP tree node is approximated to the same value, i.e., $\tilde{f}(a) = \tilde{f}(b) = y_i = f(e_i)$ for all $a, b, e_i \in r_i$. With a growing number of evaluated solutions and the solutions are recorded one by one, the approximation error of $\tilde{f}(x)$ monotonically decreases. So the approximate process can be thought of as a simple incremental learning method of machine learning.

B. An adaptive mutation mechanism

Definition 2: Neighborhood of a sub-region

The sub-region Y of node y is the neighborhood of the sub-region X of leaf node x if $X \subseteq Y$.

Every sub-region can be regarded as an optimum up to a certain neighborhood size, namely the number of neighbor points concerned. The neighborhood size points of a node decreases with its node depth as a result of the topology of the BSP tree. Therefor the neighborhood size is represented by the tree node depth difference. Suppose the sub-region Y of node y is the neighborhood of the sub-region X of node x, the neighborhood size of Y related to X is defined as the depth difference of y and x denoted by l; X is estimated as an optimal sub-region if the fitness value of x is the smallest amongst all evaluated solutions in Y.

When l is chosen to be the depth of the BSP tree, it is assumed that the approximated fitness functions consists of only one optimum. All particles are enforced to approach to the best found optimum. If l equals to zero, every sub-region is regarded as a local optimum. Then the particles are randomly mutated within its sub-region. In order to make the approximated fitness function locally guide the search, l is chosen to be 2 because l should be slightly larger than zero but much smaller than depth of the BSP tree.

Definition 3: Sub-region Distance

Suppose X and Y are the sub-regions of nodes x and y respectively. The sub-region M of node m is the smallest sub-region which contains both X and Y, i.e. $X, Y \subseteq M$, then the sub-region distances from X to Y is defined as depth difference between x and m.

After locating the optimal sub-regions of the approximated fitness function, the mutation direction of a sub-region X is defined as the direction pointing to its nearest optimal sub-region. Suppose N is the optimal sub-region set in S. The sub-region $Y \subseteq N$ is the nearest optimal sub-region of the sub-region X of node x if the distance from X to Y is the minimum amongst the distance from X to all optimal sub-regions, i.e. $Y = \text{argmin } Distance(X, A)$.

The nodes in sub-region mutation direction point to its nearest optimal neighborhood sub-region. In the adaptive mutation method, an individual x moves in the direction that the approximation fitness improvement of $\tilde{f}(x)$ at x is locally maximal. This can be done by assigning the mutation direction v as the direction pointing to the nearest historical optimum y of x, i.e., $v = y - x$. As $\tilde{f}(x)$ can be treated as a step function; its optimum in the form of an N-dimensional sub-region rather than an N-dimensional point. Also, the topology of the *BSP tree* represents the adjacencies of the sub-regions amongst the steps. Therefore the procedure to find the nearest optimum of $\boldsymbol{x}$ is equivalent to find the nearest optimal sub-region for *the sub-region of* $\boldsymbol{x}$ in $\tilde{f}(x)$. To balance the exploitative effect of this gradient descent like direction assignment, the mutation step size is randomly selected in the interval (0, 1), in which the mutant x' of x is a linear combination of x and y, $x' = x + \alpha v = (1-\alpha)\, x + \alpha y$. The mutation step size is randomly assigned within an adaptively adjusted range based on the search history, while its search direction is conducted by the approximated fitness function from the BSP tree. Thus the adaptive mutation mechanism is an adaptive, parameter-less and guided mutation operator. Moreover, it naturally avoids generating any out-of-bound particles. This is superior to some other methods that may generated out-of-bound solutions and need to define an extra repair operator to change the solutions back to valid ones.

C. Cooperative Evolution

For high-dimensional problems, search algorithms which include stochastic algorithms suffer from the "curse of dimensionality" which refers to the phenomenon of scaling problems. Every dimension of a particle will affect its overall fitness. Therefore, even if dimensions values of some particles lie very close to the corresponding dimensions of the globally optimal solution, the particles still get lower fitness. The importance of cooperation operation is that it enables "good" dimensions information within the particle swarm to be preserved, and can prevent potentially useful information from being unnecessarily discarded. By performing cooperation one dimension at a time, each dimension is able to be evaluated, respectively, and save the most useful information to accelerate convergence. The cooperative evolution can improve the algorithm performance in the high dimension problems[11].

D. Cooperative RDPSO algorithm based on entire search history

Cooperative RDPSO algorithm based on entire search history is a kind of real coded evolutionary algorithm. The whole algorithm is mainly composed of four parts, namely population initialization, evolution, mutation and cooperative selection. CRDPSO algorithm focuses on enhancing adaptive mutation based on the evolution history and selecting the optimal particle through the dynamic coordination mechanism. Fig.1 summarizes the procedure of CRDPSO. Given a D-dimensional minimization problem $F(.)$ with search space R^D, the algorithm starts from initializing the current population of n particles $P=\{p_1, p_2, \ldots, p_n\}$. Meanwhile, the BSP tree T is initialized to consist of the root node. Then the population is evaluated and its fitness is recorded by T. Each particle in P is adaptively mutate, i.e. $p_i \xrightarrow{\text{mutate}} y_i$. Afterwards each particle p_i generates four particles $\{p_{i1}, \ldots, p_{i4}\}$ by Eq.(1) to Eq.(2) and dynamic cooperative evolve with the corresponding mutant y_i on each dimension, namely the ith particle dynamic collaborative evolutionary particle group is $\{y_i, p_{i1}, \ldots, p_{i4}\}$. The best permutation is selected to form a new particle for the new generation. The new offspring population needs to be recalculated the particle's fitness value and inserted into the tree T. The processes are repeated until the termination criterion is satisfied.

Algorithm: CRDPSO

Input: 1) a D-dimension minimization problem $F(.)$ 2) search space $S \subset R^D$ 3)the population size n

```
1. Initialize the current particle swarm $P=\{p_1,p_2,\ldots,p_n\}$
2. Evaluate $p_i$: $F_i = F(p_i)$
3. $pbest_i := p_j$
4. $gbest := p_j \in P$ where $j = \arg\min\{ F(p_i) \}$
5. Initialize BSP tree $T$ to which consists of root node only
6. for $i$:= 1 to $n$
7.     [$p_i$, $F(p_i)$] insert to $T$
```

```
8.  Next $i$
9.  While terminate criteria is not satisfied
10.     for $i$= 1,2,…,$n$
11.         $p_i \xrightarrow{\text{mutate}} y_i$
12.         $\{p_{i1}, \dots, p_{i4}\}$ updated from Eq.(1)-Eq.(2)
13.         $\{y_i, p_{i1}, \dots, p_{i4}\}$ create new next generation
        according to the dynamic coordination mechanism
14.     Next $i$
15.     for $i$= 1,2,…,$n$
16.         [$p_i$, $F(p_i)$] insert to $T$
17.     Next $i$
18.     for $i$= 1,2,…,$n$
19.         if $F(p_i)<F(pbest_i)$
20.             $pbest_i= p_i$
21.         Endif
22.     Next $i$
23.     $gbest = p_j \in P$ where $j$ = arg min{ $F(p_i)$ }
24. Loop
Output：the optimal solution gbest
```
Fig.1 The pseudo-code of CRDPSO

IV. Simulation Setup

A. Test Function

A set of standard test functions $F=\{f_1(x), f_2(x), \dots, f_8(x)\}$ is adopted to test for the proposed algorithm and verify its performance.The eight test functions are shown in table 1.

B. Algorithms Setting

The performance of proposed CRDPSO is compared with standard RDPSO, cNRGA and CLQPSO[12]. To CRDPSO and RDPSO the contraction expansion factor α and β are set as [3]. To CLQPSO the β decreases linearly from 1 to 0.5 and other factors are set as [12]. To cNrGA, the cross rate is uniform crossover and sets as r_x=0.5. The population size of all the algorithms is set to 100 and the maximum iteration number is set to 2000. The dimension of all the test functions is set as D = 30. Each test function independently runs 30 times, and the mean optimum and standard deviations of test functions are obtained after 30 times experiments.

V. Simulation Results

The performance of CRDPSO is compared with those of CLQPSO，RDPSO and cNrGA. Table 2 presents the means and standard deviations of the 30 runs of the four algorithms on the eight test functions. Seen from Table 2, although RDPSO and CRDPSO both get best perform for f_1, f_2 and f_6, CRDPSO is the most superior to other three algorithms for all test functions. CRDPSO has almost obtained the optimal solutions. The standard deviations of the optimal fitness for 30 independent trials of the algorithms are also listed in the table 2. The value in boldface indicates that the corresponding algorithm is the best amongst the test algorithms on a particular test function. It shows the standard deviation of CRDPSO is the best for all the test functions. This illustrates that for the vast majority of test functions, the stability of CRDPSO algorithm is the best. Therefore, from the detailed simulation results (mean and standard deviation), CRDPSO ranks first in all eight cases. CRDPSO obtained the highest average accuracy and it also is the most stable.

VI. Conclusion

Premature convergence and diversity are the most need to solve two problems by swarm intelligence optimization algorithms. Therefor a cooperative random drift particle swarm optimization algorithm based on search history decision (CRDPSO) is proposed. The two-dimensional space partitioning tree (BSP) is used to record the solutions and fitness values in the process of evolution. The continuous search space is divided into different overlapped sub-regions as a particle mutation range by the BSP tree. This makes the corresponding mutation is a kind of adaptive mutation and provides guidance for .global and local search of particles. The dynamic collaborative mechanism is introduced for better use of context information of each dimension, so as to make the most of any new information, which improves the performance of the variation of particle; prevents the algorithm from falling into the local convergence and prevents premature. Compared with other traditional algorithms, the experiment results on 8 standard testing functions show that the proposed algorithm has the best optimization ability, with enhancement in both convergence speed and precision those demonstrate the effectiveness of the CRDPSO.

Acknowledgments

The authors gratefully acknowledge financial support from “Qing LanProject” of WuXi City College of Vocational Technology and National Natural Science Foundation of China 61300149.

Reference

[1] Kennedy J, Eberhart R C.Particle swarm optimization[C]. Proc of IEEE IntConf on Neural Networks. Perth: IEEE Press, 1995, pp: 1942-1948.
[2] Van den Bergh F.: “An Analysis of Particle Swarm Optimizers”[D], PhD Thesis. University of Pretoria, Nov 2001.
[3] Sun J, Palade V, Cai Y, et al. Biochemical systems identification by a random drift particle swarm optimization approach[J]. Bmc Bioinformatics, 2014, 15(6):1-17.
[4] Omar MA: Elementary solid state physics: principles and applications. Addison-Wesley, 1993.\Box GEP, Hunter WG, MacGregor JF, Grjavec J: Some problems associated with the analysis of multiresponse data. Technometrics, 1973, 15: 33-51.
[5] Glover F, Laguna, M. Tabu Search[M], New York ,Springer,

2013 :3261-3362.

[6] Reynolds R G. An overview of cultural algorithms[J]. Advances in Evolutionary Computation, 1999.

[7] Farmer J D, Packard N H, Perelson A S. The immune system, adaptation, and machine learning[J]. Physica D: Nonlinear Phenomena, 1986, 22(1): 187-204.

[8] Yuen S Y, Chow C K, Continuous non-revisiting genetic algorithm[C], Proceedings of IEEE Conference Evolutionary Computation, 2009:1896-1903.

[9] R.G.Reynolds.An ov Clerc M, Kennedy J: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 2002, 6: 58-73.

[10] Matsubara Y, Kikuchi S, Sugimoto M, Tomita M: Parameter estimation for stiff equations of biosystems using radial basis function networks. BMC Bioinformatics, 2006, 7.

[11] Li Y, Jiao L, Shang R, et al. Dynamic-context cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation[J]. Information Sciences, 2015, 294: 408-422.

[12] Chen Wei, Zhou Di, Sun Jun, XU Wenbo. Improved quantum-behaved particle swarm optimization algorithm based on comprehensive learning strategy[J], Control and Decision, 2012, 27(5):719-723.

TABLE 1 TEST FUNCTIONS

		X	***x*₀**	***y*₀**			***X***	***x*₀**	***y*₀**
f_1	$\sum_{i=1}^{D} x_i^2$	[-100，100]^D	[0,…,0]	0	f_5	$\sum_{i>1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	[-5.12，5.12]^D	[0,…,0]	0
f_2	$\sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i>1}^{D} \lvert x_i\rvert$	[-10，10]^D	[0,…,0]	0	f_6	$-20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - exp\left(\frac{1}{D}\sum_{i=1}^{D} cos2\pi x_i\right)+20+e$	[-32，32]^D	[0,…,0]	0
f_3	$\sum_{i=1}^{D}\left\{\sum_{j=1}^{i} x_j\right\}^2$	[-100，100]^D	[0,…,0]	0	f_7	$\sum_{i=1}^{D-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[-29，31]^D	[0,…,0]	0
f_4	$\max_{i\in[1,D]} \lvert x_i\rvert$	[-10，10]^D	[0,…,0]	0	f_8	$\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} cos\frac{x_i}{\sqrt{i}} + 1$	[-600，600]^D	[0,…,0]	0

TABLE 2 THE OPTIMIZE RESULTS OF THE TEST FUNCTION

	***f*₁**	***f*₂**	***f*₃**	***f*₄**
RDPSO	**0**	**0**	1.1162E-02	1.55963E-10
	(0)	**(0)**	(1.2250E-02)	(1.64422E-10)
CRDPSO	**0**	**0**	**0**	**0**
	(0)	**(0)**	**(0)**	**(0)**
cNrGA	3.5001E-02	1.4545E-02	1.7262E+03	3.5697E-01
	(1.7732E-01)	(2.6594E-02)	(6.1711E+02)	(1.4813E-01)
CLQPSO	1.4081E-06	4.6353E-05	9.8525E+03	1.9765E+01
	(3.5814E-07)	(9.9336E-06)	(1.9354E+03)	(1.4157E+00)
	***f*₅**	***f*₆**	***f*₇**	***f*₈**
RDPSO	1.7756E+01	**8.8818E-16**	3.7207E+01	6.4840E-03
	(9.8049E+00)	**(1.0029E-31)**	(3.0608E+01)	(7.7636E-03)
CRDPSO	**0**	**8.8818E-16**	**5.1687E-10**	**0**
	(0)	**(1.0029E-31)**	**(2.6966E-09)**	**(0)**
cNrGA	6.7054E-01	1.8326E-02	8.6977E+01	9.6984E-02
	(8.4448E-01)	(4.4425E-02)	(8.4565E+01)	(8.5924E-02)
CLQPSO	**3.1481E-06**	6.1720E-04	4.1579E+01	2.0352E-05
	(1.5211E-06)	(1.2824E-04)	(1.0091E+01)	(7.6931E-06)