Desheng LI

Anhui Science and Technology University

A hybrid cooperative quantum particle swarm optimizer with dynamic varying search area for function optimization

***Abstract**. This paper proposes a hybrid cooperative quantum particle swarm optimization (HCQPSO), hybridizing dynamic varying search area, cooperative evolution, simulated annealing and quantum particle swarm optimization (PSO) for function optimization. In the proposed HQCPSO, a technique of dynamic varying search area helps reduce the search spaces and populations of swarms, which could make the optimization more efficient. Simulated annealing is integrated in the position update to modify the trajectories of particles to avoid being trapped in the local optimum. To test the performance of HQCPSO, numerical experiments are conducted to compare the proposed algorithm with different variants of PSO. According to the experimental results, the proposed method performs better than other variants of PSO on benchmark test functions.*

***Streszczenie.** W artykule zaproponowano hybrydowy algorytm optymalizacji PSO. Porównanie z innymi, znanymi wariantami wykazało, że zastosowane w metodzie rozwiązania, pozwalają na efektywniejsze działanie proponowanego algorytmu PSO. Wyniki eksperymentalne potwierdziły powyższą tezę. (**Hybrydowy algorytm optymalizacji roju cząstek z dynamicznie zmiennym obszarem wyszukiwania w optymalizacji funkcji**).*

Keywords: Particle Swarm Optimization; Quantum-Behaved; Cooperative Evolution; Varying Search Area; Simulated Annealing.
Słowa kluczowe: optymalizacja roju cząstek, rozwój spółdzielczy, zmienny obszar poszukiwań, symulowane wyżarzanie, kształtowanie kwantowe.

Introduction

PSO, originally introduced by Kennedy and Eberhart [1], has become one of the most important swarm intelligence-based algorithms. The unique information diffusion and interaction mechanism of PSO enable it to solve many problems with good performance at low computational cost. Among the applications, function optimization has been often chosen to check the performance of them, because benchmark functions are not only well described in literature such as their properties, location and value of the optimal solution, but also have many different versions that can rove different capabilities of optimizer [2]. However, like all other intelligence algorithms, escaping from the local optimum and preventing premature conver-gences are two inevitable difficulties in implementation, especially as dimensionality increases, problems become more complex and the possibility for finding global optimum sharply decreases. This paper employs a hybrid mechanism to improve the performance of PSO for function optimization by dynamic varying search area, cooperative evolution, simulated annealing and Quantum PSO (QPSO). In the proposed approach, a technique of dynamic varying search area helps reduce the search areas and populations of swarms, which makes the algorithm more efficient. Also, simulated annealing is integrated in the position update to modify the trajectories of particles to avoid being trapped in the local optimum. The structure of the paper is organized as follows: a brief overview of QPSO and CQPSO are presented.

Quantum PSO (PSO) and Cooperative QPSO

In literature [3,4], Sun et al. proposed a Quantum Particle Swarm Optimization (QPSO), which discards the velocity vector of original PSO and consequently changed the updating strategy of particles' position to make the search more simple and efficient.

Cooperative Particle Swarm Optimization (CPSO) was proposed by Van den Bergh F. in [5], in which the high-dimension search space can be decompose into small scale ones. Compared to basic single swarm PSO, both robustness and precision are improved and guarantied.

The proposed algorithm: HCQPSO

As we known, complexity of optimization problem is not only relies heavily on the objective/constraint function, but also related with its search area. Simply speaking, subjected to the same objective/constraint function, the larger search area is, the harder it can find the solution [6]. Based on this idea, to vary the search area dynamically, or say it reduce, is necessary to accelerate the processing of algorithm. On the other hand, when the search area reduced, the populations of sub-swarms are unnecessary as big as previous. Given an optimization function:

(1) $$\min f(x)\ ,\ x=(x_1,x_2,...,x_{N_d})^T \in S =\subseteq R^{N_d}$$

where $S=[a_1,b_1]\times[a_2,b_2]\times[a_{N_d},b_{N_d}]$,the basic rationale of Dynamic Varying Search Area (DVSA) could be illustrated as the following description: Firstly, assume that N_p cooperative sub-swarms probe in the search space. When the minimal distances between optimal individuals of each sub-swarm reached a threshold, according to the maximum likelihood estimation, the hypothesis that the real optimal solution is in the area arounded by these particles was established. Then reduce the previous search area S to S', generate a new swarm with same sub-swarms on S', and decrease the popula-tions meanwhile. Finally, repeat the above procedures untill satisfy the end condition.

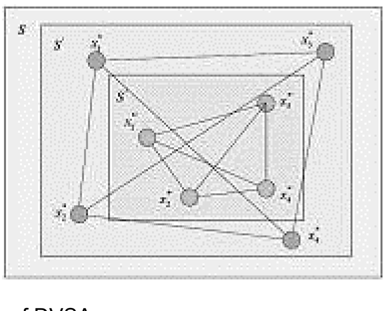

Fig. 1. A case of DVSA

Considering the vector x before the r -th reduce, where the i -th component x_i ranges over $[a_i^{r-1},b_i^{r-1}]$. Then x could be expressed as $x^{r-1}\in[a^{r-1},b^{r-1}]$.

Fig.1 examplifies the case that four cooperative sub-swarms reduce their search area. First, they probe the solution in S and get the best particles x_1^*,x_2^*,x_3^*,x_4^* which included in S'. So the search area becomes S'. The next time of reduce to S'' is the same procedure.

In this part, we will give the condition when the DVSA occures. Surpose there exist N_p sub-swarms, the best particles set found is writen

(2) $$x_b^{r-1} = \{x_b^{r-1,1}, x_b^{r-1,2}, ..., x_b^{r-1,N_p}\},$$
$$x_b^{r-1,p} = (x_{b1}^{r-1,p}, x_{b2}^{r-1,p}, ..., x_{bN_d}^{r-1,p}), p = 1, 2, ..., N_p$$

Now, let us consider the distance among them.

(3) $$D(x_b^{r-1,i}, x_b^{r-1,j}) = \left\| x_b^{r-1,i}, x_b^{r-1,j} \right\|_2$$

(4) $$D^{r-1} = \max_{x_b^{r-1,i}, x_b^{r-1,j} \in x_b^{r-1}} D(x_b^{r-1,i}, x_b^{r-1,j})$$

where $\|\cdot\|_2$ is the 2-norm on corresponding search area.

When D^{r-1} reached a small threshold, according to the maximum likelihood estimation, the hypothesis that the real optimal solution is in the area arounded by these particles was established. So the latter search can be performed around these particles. In light of this, we can give the condition of DVSA as shown in formula (5).

(5) $$D^{r-1} < \lambda \cdot \left\| a^{r-1} - b^{r-1} \right\|_2, \lambda \in \left(0, 1/N_p\right]$$

In other words, if the above equation is satisfied, then change the search area of the next generation of sub-swarms untill the DVSA occures again. λ can be a fix number, but more often, it is a paramater can be changed adaptively according to the results of evolution.

Let consider the search area after reduce. Note that after the r-th reduce, x_i ranges over $[a_i^r, b_i^r]$. Then the upper/Lower bounds are defined by the following equation:

(6) $$\begin{cases} a_i^r = \min\{x_{bi}^{r-1,p}\} - \xi \cdot (b_i^{r-1} - a_i^{r-1}) \\ b_i^r = \min\{x_{bi}^{r-1,p}\} + \xi \cdot (b_i^{r-1} - a_i^{r-1}) \end{cases}, \xi \in \left(0, 1\right]$$

To guarantee the new search area not larger than the previous area, the above equation should be modified as follows:

(7) $$a_i^r = \begin{cases} a_i^{r-1}, a_i^r < a_i^{r-1} \\ a_i^r, othewise \end{cases}, b_i^r = \begin{cases} b_i^{r-1}, b_i^r < b_i^{r-1} \\ b_i^r, othewise \end{cases}$$

The computational complexity also relies heavily on the scale of the population of the swarm/sub-swarm. In general, the more time about particle evaluation, the more computation takes place. Hence, under the permission of optimization performance, it is necessary to cut down the population of sub-swarms.

In this article, we will follow a traditional method called search granularity. Take the particle after the r-th reduce for instance, whose i-th component x_i ranges over $[a_i^r, b_i^r]$. The distance of this interval can be written as Eq.(8) which reflect the refined effort of search。If the distance among the solutions is small, we can say that search granularity is small, and vice versa. From the real experience, the bigger swarm, the less distance among the particles, and aslo lessen the search granularity.

(8) $$d_i^r = b_i^r - a_i^r$$

Furthermore, if it is asked that the search granularity on $[a_i^r, b_i^r]$ should be $1/N_{ik}$, the population scale of sub-swarm can be determined abey the below equation.

(9) $$N_i^r = \left\lfloor \prod_{k=1}^{N_d} N_{ik} \cdot d_i^r \right\rfloor$$

where $\lfloor \cdot \rfloor$ is the floor function. When the search area descreses, the population of the related sub-swarm also becomes small.

After determining the populations of sub-swarms, how to generate the new swarm becomes vital in the next steps. In our previous research, an electoral mechanism was proposed to select the elites from each sub-swarm. In our algorithm, an electoral swarm is generated by the voting of primitive sub-swarms and also participates in evolution of swarm, whose candidate particles come from primitive sub-swarms with variable votes. In reverse, the number of selected particles could also impact the voting of the primitive sub-swarms, such as the total number of candidates and quota of selected ones. The selected candidates could share their components with best segments of position, which are then being composed into a new particle position to participate in the combining of positions. A new component of particle's position is also imported, i.e., P_{ed}^{best}, denoting the electoral best position composed by the dimensions of elected candidates. Recall that the local attractor in original quantum-behaved PSO can be written $\breve{P}_{id} = (c_1 r_1 P_{id} + c_2 r_2 P_{gd}) / (c_1 r_1 + c_2 r_2)$. Employing an electoral best position, it can be augmented into

(10)
$$\breve{P}_{id} = (c_1 r_1 P_{id}^{best} + c_2 r_2 \breve{P}_{gd}^{best} + c_3 r_3 P_{ed}^{best}) / (c_1 r_1 + c_2 r_2 + c_3 r_3) \cdot$$

So the new local attract position could be written as follows:

(11) $$\breve{P}_{id}' = \varphi \times P_{id}^{best} + \psi \times \breve{P}_{gd}^{best} + (1 - \varphi - \psi) \times P_{ed}^{best} \cdot$$

Due to the employment of this component, the particles in each sub-swarm therefore update their global best position by Eq.(12), which is the result associated with minimal fitness value of their local best positions and global best positions of electoral swarm.

(12) $$b(u, S_u.\breve{P}_{gd}^{best}) = argmin\, fitness(b(u, S_u.P_{id}^{best}), b(u, S_u. P_{ed}^{best})),\ 1 \le id \le s, 1 \le u \le k$$

Simulated annealing (SA) is a local search approach imported by Kirkpatrick et al., which begins with an initial given solution and moves around the neighbors to generate a new solution. SA belongs an optimization algorithm based on iterative solution strategy importing a probability function with Maxwell-Boltzmann distribution to search in the space. The physical foundation is the solid annealing process that first makes solid reach high temperature, then slows down the particles form the cooling for the position of the lowest energy. Hence, an optimization problem can be considered in a SA perspective, i.e., a minimum energy structure related to an optimal solution; a particle structure being a problem solution; a structure of energy as the objective function values; the terminal and initial temperature mapping for the control parameters of the search. The concrete phases of SA are shown in the following pseudocode: In this algorithm, as shown in Alg.1, we utilize the SA to refine the current position of each sub-swarm after composite of different dimensions like the literature.

```
Algorithm 1. Simulated annealing (SA)
Input: Cooling schedule with a reduction function $T(k) = t_0(\alpha)^k$, the current position of each sub-swarm Hid, starting temperature $t_0$, final temperature $t_{final}$, and maximum generation number $n_{max}$;
Output:;
Procedure:
 Initialize a random position $P_{id}$ of particle;
 Calculate the cooling function of $T(P_{id})$ and $T(H_{id})$.
  While $t > t_{final}$ Do
    While iter $\leqslant n_{max}$ Do
      Select $s$ as neighbors of $H_{id}$;
      $\sigma = T(P_{id}) - T(H_{id})$;
      If $\sigma < 0$ Then $H_{id} = s$;
      Else
        Generate $r \sim U(0; 1)$;
        If $r < e^{-\sigma/t}$ Then $H_{id} = P_{id}$;
        End If
      End If
    End While
    $t = T(k)$;
  End While
  Return $H_{id}$;
End
```

Based on the above methods, a variant of algorithm, HCQPSO is proposed with the below computing steps.

```
Algorithm 2. HCQPSO
Input: Control parameters and cooling function of SA.
Output: Solution;
Procedure:
 Set control parameters and initialize $N_p$ sub-swarms.
 Sub-swarms perform evolutional period independently.
 Select the optimal particles from different subswarms.
  Combine the components from different dimentions.
  Update all sub-swarms by SA according to Alg.1.
  Check the judgment conditions, if satisfied, then end.
  Check the condition of DVSA, if not satisfied, and then
    go to the second step.
  Calculate the reduced search area by Eq.(6,7).
  Adjust the population scale by Eq.(9).
  Generate a new swarm by electoral mechanism and
    go to the second step.
  Return the optimal solution.
End.
```

In this subsection, an analysis of the convergence of HCQPSO is provided. We discuss it from two perspectives, i.e., search area and population of swarms. Firstly, we analyze the varying of interval measure cased by two neighboring reduces. According to the policy of DVSA, it can be described as follows according Eq.(6,7):

$$b_i^r - a_i^r \le b_i^{r-1} - a_i^{r-1} \tag{13}$$

Without loss of generality, let

$$b_i^r - a_i^r = k_i^r \cdot (b_i^{r-1} - a_i^{r-1}), k_i^r \in (0,1], \tag{14}$$

$$\begin{aligned} b_i^r - a_i^r &= k_i^r \cdot (b_i^{r-1} - a_i^{r-1}) = k_i^r \cdot k_i^{r-1}(b_i^{r-2} - a_i^{r-2}) \\ &= \dots = K_i^r \cdot (b_i - a_i),\ K_i^r = \prod_{j=1}^{r} k_i^r \le \min\{k_i^1, k_i^2, \dots, k_i^r\} \end{aligned} \tag{15}$$

From the formula (14), we can see that when search area varies, the reduced area becomes the k_i^r times of oringin area. So when serveral genenrations of this procedure happens, the final area could be heavily reduced with the considerable promotion of efficiency. Secondly, in consideration of swarm populations, we have.

$$\begin{aligned} N_j^r &= \left\lfloor \prod_{i=1}^{N_d} N_{ji} \cdot d_i^r \right\rfloor = \left\lfloor \prod_{j=1}^{N_d} N_{ji} \cdot (b_i^r - a_i^r) \right\rfloor \\ &= \left\lfloor \prod_{j=1}^{N_d} N_{ji} \cdot K_i^r (b_i^r - a_i^r) \right\rfloor < \left\lfloor \prod_{j=1}^{N_d} N_{ji} \cdot (b_i - a_i) \right\rfloor \end{aligned} \tag{16}$$

The above inference shows that as the search area decreases, the related populations of swarms also be cut down with a certain rate.

Numerical results

To study the search behavior and its performance of HCQPSO with other versions of PSO, such as plain PSO, CPSO, and CQPSO, some typical benchmark functions are selected as the examples.

Rastrigin's function, is frequently used as a test function to test the performance of optimization algorithms as in Fig.2(a). Based on Sphere function, it uses cosine function to generate lots of local optimal points. It is a complex multimodal function, and optimization falls into the local optimum easily. Griewank function is a spin, inseparable variable-dimension multimode function. As the increase of its dimension, the scope of local optimum gets narrower so that searching global optimum becomes easy relatively. Therefore, for Griewank function, it is harder to get solution in low dimension than in high dimension. Michalewics function is a multi-modal function with parameter m which changes the steepness of valleys. The Lévy No. 8 function, as shown in Fig.2(b), has one global minimum and, approximately, 125 local minima.

Computational results of variants of PSO used in the paper is qualitatively ranked in Table 1. From Table 1, we can clearly get that the proposed HCQPSO algorithm performed greatly better than the plain PSO and QPSO. Also, compared to the basic Cooperative PSO (CPSO), the convergence property has been enhanced by the proposed techniques in the paper.

In Fig.3, the black cycles denote the distribution of particles of 2-d Rastrigin's function in QPSO under DVSA, while the red ones express that of HCQPSO with only two cooperative sub-swarms. It can be clearly seen that in HCQPSO, the search area in each generation of iteration is reduced dynamically into the potential rectangles along two red lines on horizontal/vertical directions. In addition, we can also find that the populations of the latter generations has been reduced obviously, which means the lower computational complexity meanwhile.

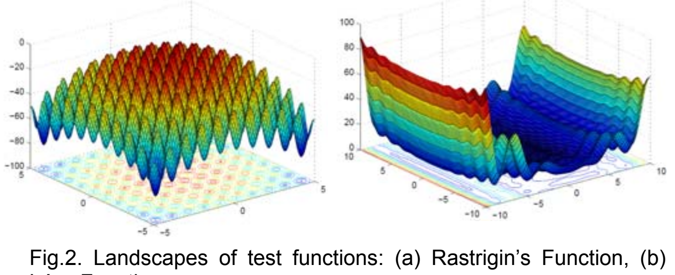
Fig.2. Landscapes of test functions: (a) Rastrigin's Function, (b) Lévy Function

Moreover, the convergence ability is also investigated in our experiment. Fig.4 illustrates the typical convergence of PSO, CPSO, QPSO, CQPSO and HCQPSO on the benchmark Michalewics function. From the figure, it can be seen that the varying curves of objective values using the HCQPSO descend much faster than using plain PSO and QPSO. In addition, the fitness values descent to lower level by using HCQPSO than CPSO due to the different mechanisms of simulated annealing and DVSA.

The results of the experiments indicated that the proposed HCQPSO can lead to more efficiency and stability than plain PSO, QPSO, CPSO and CQPSO.

Table.1. Results of functions optimization in benchmark

Functions	PSO			QPSO			CQPSO			HCQPSO		
	Min	Max	Ave	Min	Max	Ave	Min	Max	Ave	Min	Max	Ave
Rastrigin's	6.12E-06	9.00E+00	4.67E+00	3.01E+00	8.90E+01	4.91E+01	**2.50E+01**	1.25E+02	7.52E+01	**2.50E+01**	**2.50E+01**	**2.50E+01**
Griewank	9.87E-03	1.50E-01	5.04E-02	9.12E-06	8.68E-02	3.86E-04	2.58E-06	4.95E-02	1.56E-02	**0**	8.04E-10	6.83E-11
Michalewics	-9.284715	-7.846484	-8.387755	-9.375576	-8.195449	-9.006959	-9.613477	-8.394369	-9.237160	**-9.660151**	**-9.660151**	**-9.660151**
Lévy	0.6663	4.6048	2.3496	0.1019	11.5187	2.2827	3.27E-05	2.94E-04	1.48E-04	**2.01E-05**	6.37E-04	2.38E-04

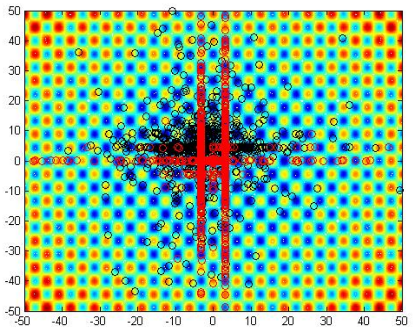

Fig.3. Particles in 2-d Rastrigin's Function in QPSO and HCQPSO under DVSA

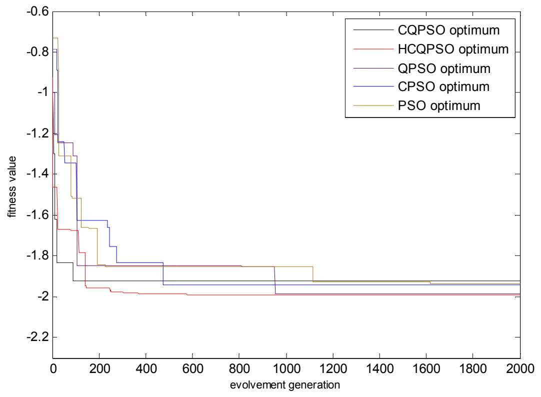

Fig.4. Evolution curves of HCQPSO and other PSO variants

Conclusion

This paper proposes a novel variant of QPSO called hybrid cooperative quantum particle swarm optimizer, and explores how to vary the search area dynamically in the view of space reduce. Moreover, simulated annealing is also integrated in the position update to avoid being trapped in the local optimum. Both the theoretical analysis and experimental results on benchmark functions show that this algorithm is more efficient than other versions of particle swarm optimizers.

Acknowledgments

This work was supported by the Talent Introduction Special Fund of Anhui Science and Technology University (No. ZRC2011304), and Science and Technology Plan Project of Chuzhou City (No. 201236).

REFERENCES

[1] Kennedy J., Eberhart R.C.: Particle swarm optimization, In: Proc. *IEEE International Conference on Neural Networks*, Perth, Australia, 1995, 1942-1948
[2] Lazinica A.: Particle Swarm Optimization, *IN-TECH*, 2009.
[3] Sun J., Feng B., Xu W.B.: Particle Swarm Optimization with Particles Having Quantum Behavior, *Proc. Congress on Evolutionary Computation*, 325(2004)
[4] Liu J., Sun J., Xu W.B.: Design IIR Digital Filters Using Quantum-Behaved Particle Swarm Optimization, *Advances in Natural Computation*, *LNCS,* 4222(2006), 637-640
[5] Van den Bergh F., Engelbrecht A.P.: A cooperative approach to particle swarm optimization, *IEEE Transactions on In Evolutionary Computation*, 8(2004), 1-15
[6] El Dor A., Clerc M., Siarry P.: A multi-swarm PSO using charged particles in a partitioned search space for continuous optimization[J]. *Computational Optimization and Applications*, 11(2011): 1-25

***Authors**: dr inż, Li Desheng, Anhui Science and Technology University, School of Science, ul. Donghua Road 9, 233100, Fengyang, Anhui, China, E-mail: ldsyy2006@126.com*.