A hybrid cooperative cuckoo search algorithm with particle swarm optimisation

Lijin Wang*

School of Computer Science and Technology,
Shandong University,
Jinan, China
and
College of Computer and Information Science,
Fujian Agriculture and Forestry University,
Fuzhou, China
Email: lijinwang@fafu.edu.cn

Yiwen Zhong

College of Computer and Information Science,
Fujian Agriculture and Forestry University,
Fuzhou, China
Email: yiwzhong@fafu.edu.cn

Yilong Yin*

School of Computer Science and Technology,
Shandong University,
Jinan, China
Email: ylyin@sdu.edu.cn
*Corresponding authors

Abstract: This paper proposes an improved hybrid cooperative algorithm that combines cooperative cuckoo search algorithm and particle swarm optimisation, called HCCSPSO. The cooperative co-evolutionary framework is applied to cuckoo search algorithm to implement dimensional cooperation. The particle swarm optimisation algorithm, viewed as a cooperative component, is embedded in the back of the cuckoo search algorithm. During iteration, the best solution obtained by the previous cooperative component is randomly embedded in the last one to avoid the pseudo-minima produced by the previous one, while the subcomponents of best solution from the last cooperative component are also randomly planted in the subcomponents of the previous one. The results of experimental simulations demonstrate the improvement in the efficiency and the effect of the cooperation strategy, and the promising of HCCSPSO.

Keywords: cuckoo search algorithm; particle swarm optimisation; cooperative co-evolutionary; dimensional cooperation; component cooperation.

Reference to this paper should be made as follows: Wang, L., Zhong, Y. and Yin, Y. (2015) 'A hybrid cooperative cuckoo search algorithm with particle swarm optimisation', *Int. J. Computing Science and Mathematics*, Vol. 6, No. 1, pp.18–29.

Biographical notes: Lijin Wang received his PhD from Beijing Forestry University in 2008. He is currently a Post-Doctoral Fellow at the School of Computer Science and Technology, Shandong University, Jinan, China. He is also currently an Associate Professor at the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, China. His current research interests include nature-inspired algorithm, machine learning, and medical image processing.

Yiwen Zhong is a Professor of Computer Science at the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, China. He received his PhD in Computer Science and Technology from Zhejiang University in 2005, Hangzhou, China. His current research interests include computational intelligence, data visualisation and bioinformatics.

Yilong Yin received his PhD in 2000 from Jilin University, China. From 2000 to 2002, he worked as a Post-Doctoral Fellow at the Department of Electronics Science and Engineering, Nanjing University, China. He is currently Director of MLA Group and Professor of the School of Computer Science and Technology, Shandong University, China. His research interests include machine learning, data mining, and biometrics.

1 Introduction

Cuckoo search (CS), a swarm intelligence-based nature-inspired algorithm, bases on the obligate brood parasitic behaviour of some cuckoo species in combination with the Lévy flights behaviour of some birds and fruit files (Yang and Deb, 2009, 2010). During iteration process, CS cooperatively uses Lévy flights random walk (LFRW) and biased/selective random walk (BSRW) to search new solutions. After each random walk, the greedy strategy selects a better solution from the current and new generated solutions according to their fitness.

CS has gained popularity due to its high efficiency. One of reasons for high efficiency may be that CS is a cooperative learner. Due to CS using multi-agents and sharing information among them, the individual cooperation is a basic form. In LFRW, the best individual is shared with others, while information exchange is happened between the individuals in items of the crossover operation in BSRW.

The component cooperation is another form which is between LFRW and BSRW. The population output from LFRW is continually optimised in BSRW, while the population from BSRW is also coped with by LFRW. Along the view of component cooperation, other evolutionary algorithms are viewed as cooperative components integrated into CS to improve the performance (Wang et al., 2011a; Ghodrati and Lotfi, 2012; Li and Yin, 2012; Srivastava et al., 2012; Babukartik and Dhavachelvan, 2012).

The third form is dimensional cooperation. To produce a solution vector for the objective function, a valid solution vector can only be formed by using information from dimension, e.g., DDICS (Wang et al., 2013c), Co2CS (Hu and Yin, 2013), and CCCS (Zheng and Zhou, 2013). Co2CS and CCCS used the framework of cooperative co-evolutionary (CC) proposed by Potter and de Jong (1994). However, these two literatures did not investigate in detail the possibility that the partitioning could lead to the introduction of pseudo-minima.

In light of the component cooperation, we propose a component-based cooperative learner, called HCCPSO, which considers CC-based CS as a component, and PSO as another component. The dimensional cooperation is still used in HCCSPSO, where the CC framework is applied to CS which is similar to Co2CS and CCCS. HCCSPSO also employs PSO which is embedded in the back of CC-based cuckoo search algorithm. Meanwhile, the bi-group strategy is used for two cooperative components in HCCSPSO. During iteration, the best population member, obtained by the CC cuckoo search algorithm, is randomly embedded in the population for PSO. In this case, HCCPSO can void the pseudo-minima produced by the CC algorithm. This is another motivation for this paper. In addition, the subcomponents of best solution from the last cooperative component are also randomly planted in the subcomponents of the previous one. We investigate HCCSPSO on ten benchmark functions, and the results demonstrate the improvement in the efficiency and the effect of the cooperation strategy, and the promising of HCCSPSO.

The remainder of this paper is organised as follows. Section 2 describes the CS algorithm, the PSO algorithm and the CC technique. Section 3 introduces the HCCSPSO algorithm. Section 4 reports the experimental results. Section 5 concludes this paper.

2 Preliminary

2.1 Cuckoo search

CS is a simple yet very promising population-based stochastic search technique, which is based on three idealised rules (Yang and Deb, 2009, 2010). The first rule is that one egg, laid by each cuckoo at each time, is dumped in randomly chosen nest. The second rule is that the nests with high-quality eggs will carry over to the next generation. The third rule is that the number of available host nests is fixed, and the host bird discovers the egg laid by a cuckoo with a probability $p_a \in [0, 1]$.

Generally speaking, when CS is used to solve an objective function $f(x)$ with the solution search space $[x_{j,\min}, x_{j,\max}]$, $j = 1, 2,\ldots,D$, a nest represents a candidate solution $X = (x_1,\ldots,x_D)$.

In the initialisation phase, CS initialises solutions that are randomly sampled from solution space by

$$x_{i,j,0} = x_{i,j,\min} + r \times \left(x_{i,j,\max} - x_{i,j,\min}\right), \quad i = 1, 2,\ldots, NP \tag{1}$$

where r represents a uniformly distributed random variable with the range [0, 1], and NP is the population size.

After initialisation, CS goes into iterative phase. At generation G ($G > 0$), LFRW is firstly employed to search new solutions around the best solution obtained so far, and can be formulated as follows.

$$X_{i,G+1} = X_{i,G} + \alpha_0 \frac{\phi \times u}{|v|^{1/\beta}} \left(X_{i,G} - X_{best} \right) \tag{2}$$

where α_0 is a scaling factor (generally, $\alpha_0 = 0.01$), $X_{i,G}$ means the i^{th} solution at generation G, X_{best} represents the best solution obtained so far, u and v are random numbers drawn from normal distribution with mean of 0 and standard deviation of 1, β is a constant and set to 1.5, and ϕ is formulated as follows.

$$\phi = \left(\frac{\Gamma(1+\beta) \times \sin(\pi \times \beta / 2)}{\Gamma\left(\left(\frac{1+\beta}{2} \right) \times \beta \times 2^{(\beta-1)/2} \right)} \right)^{1/\beta} \tag{3}$$

where Γ is *gamma function*.

After LFRW, CS selects the better solutions from the current and new generated solutions according to their fitness with the greedy strategy. In this case, CS can be easily trapped into a local optimum, therefore, BSRW is used to discover new solutions far enough away from the current best solution by far field randomisation. BSRW can be modelled as follows.

$$x_{i,j,G+1} = \begin{cases} x_{i,j,G} + r \times \left(x_{m,j,G} - x_{n,j,G} \right), & rand > p_a \\ x_{i,j,G}, & \text{otherwise} \end{cases} \tag{4}$$

where m and n are random indexes, r is a scaling factor with the range [0, 1], and p_a is a finding probability.

After BSRW, CS also uses the greedy strategy to select the better solutions from the current and new generated solutions according to their fitness. At the end of each iteration process, the best solution is updated.

2.2 PSO

PSO, proposed by Kennedy and Eberhart (1995) and Eberhart and Kennedy (1995), is a population-based stochastic optimisation technique. The population is made up of potential solutions, called particles, similar to birds in a flock. In general, when PSO is employed to solve an objective function $f(x)$ with the D-dimensional search space, each particle i has two vectors to maintain its state, namely, a position vector $X_i(x_{i1},\ldots,x_{iD})$ and a velocity vector $V_i(v_{i1},\ldots,v_{iD})$. During iteration, each particle updates its velocity by learning from its historically best position and the best position of the whole swarm obtained so far, and then updates its position according to its new updated velocity. The update rules in the original PSO are given as follows.

$$v_{id} = v_{id} + c_1 r_{1d} \left(pBest_{id} - x_{id} \right) + c_2 r_{2d} \left(gBest_d - x_{id} \right) \tag{5}$$

$$x_{id} = x_{id} + v_{id} \tag{6}$$

where $pBest_i(pBest_{i1},\ldots,pBest_{iD})$ is the historically best position of i^{th} particle, $gBest(gBest_1,\ldots,gBest_D)$ means the best position of the whole swarm obtained so far, c_1 and c_2 are acceleration parameters to learn the relative importance of $pBest_i$ and $gBest$, respectively, d stands the d^{th} dimension of search space, and r_{1d} and r_{2d} are uniformly distributed random numbers within rang between 0 and 1 for d^{th} dimension.

To ensure the velocity flying within a reasonable range, an inertia weight and a constriction factor are proposed by Shi and Eberhart (1998). Therefore, the velocity update rule is rewritten respectively as follows.

$$v_{id} = \omega v_{id} + c_1 r_{1d}\left(pBest_{id} - x_{id}\right) + c_2 r_{2d}\left(gBest_d - x_{id}\right) \tag{7}$$

where ω is an inertia weight.

Although, various improved PSOs have been developed, such as IPSO (Gao et al., 2012), CMQPSO (Mu et al., 2013), CAPSO (Dai et al., 2014), IDPSO (Wang and Wang, 2014), DNSPSO (Wang et al., 2013a), GOPSO (Wang et al., 2011b), AMPSO (Wang et al., 2013b), and so on, the PSO with inertia weight is employed in this paper, and ω decreases linearly from 0.9 to 0.4 during iteration in the experimental simulations.

2.3 CC technique

The CC (Potter and de Jong, 1994) technique, regarded as an approach to implementing the divide-and-conquer strategy, is a general framework for integrating evolution algorithms to solve the complex and large optimisation problems. Problem decomposition is the first and a critical step in the CC framework, where a large problem should be decomposed into subcomponents. In other words, a D-dimensional optimisation problem would be decomposed into m n-dimensional subcomponents where D equals that m multiplies n. The next step is subcomponent optimisation where each subcomponent would be separately optimised by the evolution algorithms. Subcomponents cooperation would happen when subcomponents being evaluated. More detailed description of CC framework can be found in the literatures (Potter and de Jong, 1994; Jansen and Wiegand, 2004; Van den Bergh and Engelbrecht, 2004; Shi et al., 2005; Yang et al., 2008).

3 HCCSPSO

In this section, we propose a hybrid cooperative cuckoo search algorithm with particle swarm optimisation, called HCCSPSO. The pseudo-codes of HCCSPSO are shown in Figure 1.

In Figure 1, HCCSPSO uses the CC framework to decompose an optimisation problem into several subcomponents which are separately optimised by the standard CS algorithm. In HCCSPSO, to evaluate each subcomponent, a simple method is to select the best solution of other subcomponents for constructing D-dimensional solution. In each subcomponent, HCCSPSO selects the better solutions from the current and new generated solutions according to their fitness with greedy strategy. For example, the solution vector $X = [x_1, x_2,\ldots,x_j,\ldots,x_D]$ can be decomposed into m $S_iX = [x_1, x_2,\ldots,x_j,\ldots,x_{D/m}]$ $(i = 1, 2,\ldots,m)$ according to the CC framework. Being evolved by CS, the new solution of i^{th} subcomponent is $N_iX = [x_1, x_2,\ldots x_j,\ldots x_{D/m}]$. To evaluate N_iX and

S_iX, the D-dimensional solutions are composed as Y_N = [$gbest_1$,…,$gbest_{i-1}$, N_iX, $gbest_{i+1}$,…,$gbest_m$] and Y_S = [$gbest_1$,…,$gbest_{i-1}$, S_iX, $gbest_{i+1}$,…,$gbest_m$]. If the fitness of the solution Y_N is better than that of the solution Y_S, the solution S_iX would be replaced with N_iX.

Figure 1 Pseudo-code of the HCCSPSO algorithm

Input: The number of dimension D, the number of subcomponent m, and other parameters of CS and PSO.
Output: the best solution *gbest*
Begin
1 Initialisation step:
1.1 **Initialize** two D-dimensional solution sets pop_1 and pop_2 with search space [$x_{j,min}$, $x_{j,max}$].
1.2 **Decompose** the D-dimensional solution set pop_1 into m n-dimensional *subpop*
Repeat
2 Dimensional Cooperation with the CC framework:
2.1 **For each** i m **Do**
Evolve the $subpop_i$ using CS algorithm
3 Exchange information in component cooperation:
4 Optimization by the PSO with inertia weight algorithm:
5 Exchange information in component cooperation:
Until stopping condition is true
End

HCCSPSO also uses the PSO algorithm laid at the behind of dimensional cooperation. PSO optimises the entire D-dimensional solutions. It is worthy pointing out that the PSO algorithm can be replaced with other evolutionary algorithms. Moreover, HCCSPSO employs the bi-group strategy. During iteration, the best solution of pop_1 is randomly planted in pop_2 optimised by PSO. In this case, HCCPSO can void the pseudo-minima produced by the previous cooperative component. In addition, exchange information happens after PSO. The subcomponents of best solution of pop_2 are randomly embedded in pop_1.

4 Experimental verifications

In this section, a suite of ten unconstrained single-objective benchmark functions (Suganthan et al., 2005) are used to validate the performance of HCCSPSO. These functions include separable functions F_1 and F_9, and non-separable functions F_2–F_8 and F_{10}. More detailed description of them can be found in literature (Suganthan et al., 2005).

For HCCSPSO, there are three control parameters, namely, the population size NP, the finding probability p_a, and the number of subcomponent m. For all experiments, unless a change is mentioned, the two population sizes NP are D which is the dimension of problem and set 30, p_a is 0.25, and m is 5. Moreover, in our experiments, each algorithm is used to optimise each benchmark function over 50 independent runs. The maximum number of function evaluations (MaxFEs) is 10,000 × D.

In our experimental studies, we select four performance criteria from the literature (Suganthan et al., 2005; Noman and Iba, 2008) to evaluate the performance of the algorithms.

- *Error:* Error is the function error value defined as $(f(x) - f(x^*))$, where x^* is the global optimum of the function, and x is the best solution obtained by the algorithm in a run. The minimum function error value that each algorithm can find is recorded in different runs, and the average and the standard deviation of the error value are calculated. The notation $AVG_{Er} \pm SD_{Er}$ is used in different tables.
- *Number of function evaluations (FES):* The number of function evaluations is also recorded in different runs when each algorithm archives the value to reach (VTR) suggested in Noman and Iba (2008) within MaxFEs. The average and standard deviation of *FES*, denoted $AVG_{Ev} \pm SD_{Ev}$ is used different tables.
- *Number of successful runs (SR):* The number of successful runs is recorded when the VTR is reached within MaxFEs.
- *Convergence graphs:* The convergence graphs show the average function error value performance of the total runs in respective experiments.

4.1 The effect of HCCSPSO

To show the performance of HCCSPSO, we focus on the solution accuracy and the convergence speed of it. The solution accuracy obtained by HCCSPSO is compared with the ones obtained by CS in Table 1. The best results are marked in boldface. The *t*-test results between HCCSPSO and CS is also given.

Table 1 Solution accuracy obtained by CS and HCCSPSO

Fun	*CS*	*HCCSPSO*	
	$AVG_{Er} \pm SD_{Er}$	$AVG_{Er} \pm SD_{Er}$	*t-test*
F_1	7.72e-30 ± 2.10e-29	0.00e+00 ± 0.00e+00	2.599‡
F_2	8.15e-03 ± 8.29e-03	7.81e-09 ± 1.60e-08	6.952‡
F_3	2.11e+06 ± 4.68e+05	8.79e+05 ± 5.15e+05	12.509‡
F_4	1.82e+03 ± 9.42e+02	2.96e+02 ± 4.20e+02	10.448‡
F_5	3.04e+03 ± 8.05e+02	6.39e+03 ± 1.97e+03	–11.131‡
F_6	2.43e+01 ± 2.55e+01	3.01e+00 ± 2.66e+00	5.872‡
F_7	1.03e-03 ± 2.51e-03	2.28e-02 ± 2.02e-02	–7.562‡
F_8	2.09e+01 ± 5.28e-02	2.07e+01 ± 1.36e-01	9.694‡
F_9	2.86e+01 ± 6.05e+00	1.80e+00 ± 2.15e+00	29.515‡
F_{10}	1.75e+02 ± 2.99e+01	1.40e+02 ± 4.60e+01	4.511‡

Note: ‡: the value of t with 98 degrees of freedom is significant at 0.05 level by a two-tailed t-test between HCCSPSO and CS

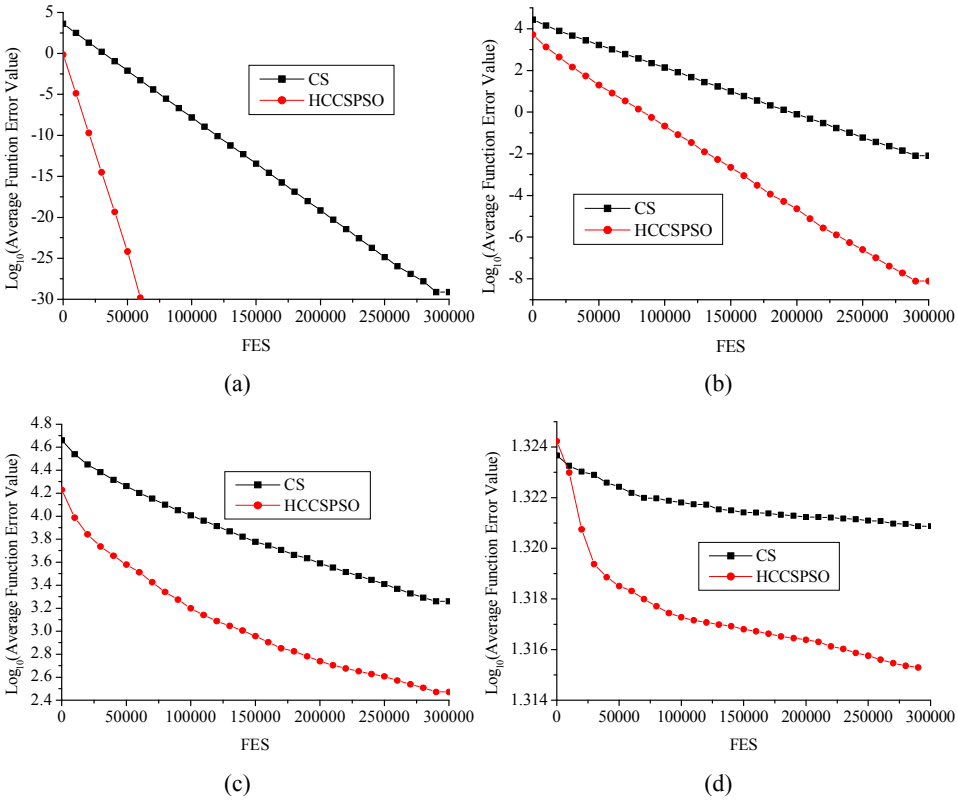
From Table 1, it is clearly that HCCPSO outperforms CS for the separable functions, and overall archived the better solution accuracy than CS on the non-separable functions. For example, HCCPSO does better than CS on F_1 and F_9, especially obtains the global optimum on F_1. Moreover, HCCPSO provides the solutions with higher accuracy to F_2, F_3, F_4, F_6, F_8, and F_{10}. However, HCCSPSO does not show an advantage on F_5 and F_7. This is perhaps because that there are not enough population sizes to provide the more diversity information for HCCSPSO to solve this kind of optimisation problems.

Table 2 lists the average number of function evaluations and the number of successful runs when CS and HCCPSO reach the VTR within MaxFEs. It is obviously that HCCPSO overall has a faster and steadier convergence to the VTR than CS. In terms of SR, HCCSPSO shares with CS on F_1, but HCCSPSO performs better than CS with the help of FES. In addition, HCCSPSO significantly outperforms CS for F_2, F_6 and F_9 by means of SR or FES. CS defeats HCCSPSO on F_7 by the aid of SR, however, in terms of FES, HCCSPSO converge faster than CS does at the beginning of iteration.

Table 2 Average FES to reach VTR by CS and HCCSPSO

Fun	*CS*	*HCCSPSO*
	$AVG_{Ev} \pm SD_{Ev}$ *(SR)*	$AVG_{EV} \pm SD_{EV}$ *(SR)*
F_1	9,3154.0 ± 2,410.3 (50)	2,1998.0 ± 715.6 (50)
F_2	-	229,594.0 ± 20,476.8 (50)
F_6	-	180,197.0 ± 43,830.9 (15)
F_7	154,552.0 ± 25,342.7 (50)	56,650.0 ± 12,298.4 (22)
F_9	-	65,452.0 ± 10,970.0 (16)

Figure 2 Convergence graphs of CS and HCCSPSO for (a) F_1, (b) F_2, (c) F_4, and (d) F_8 (see online version for colours)



To provide more information about the convergence process of HCCSPSO, Figure 2 shows the convergence process curve of F_1, F_2, F_4, and F_8 optimised by CS and HCCSPSO. It is apparent that HCCSPSO perform better than CS in terms of final solution and convergence speed.

From the above discussion, HCCSPSO with cooperative strategy overall outperforms CS without cooperative strategy in terms of solution accuracy and convergence speed. It can be concluded that improvement in the efficiency and the effect of cooperative strategy, and the promising of HCCSPSO.

4.2 Comparison with other improved CS algorithms

In this section, CSPSO (Wang et al., 2011a) and Co2CS (Hu and Yin, 2013) are chosen to compare with HCCSPSO. The former combined CS with PSO, and the latter employed the CC framework. In addition, HCCSPSO is also compared with OLCS (Li and Yin, 2012) which used component cooperation with CS and orthogonal experiment design method. Table 3 lists the results where the results of the compared algorithms are directly taken from the literature (Hu and Yin, 2013). In addition, 'rank' denotes the ranking of the corresponding algorithm.

From Table 3, there is no specific algorithm to achieve the best solution for all optimisation problems. For example, in terms of the solution accuracy, CSPSO brings solutions with highest accuracy to F_2 and F_4. OLCS does best in optimising F_5 and F_7. Co2CS is good at F_3 and F_8. HCCSPSO yields the global optimum on F_1, and bring solutions with higher accuracy to F_6, F_9, and F_{10}.

Table 3 also ranks the algorithms on performance in terms of solution accuracy. It can be observed that HCCSPSO offers the best overall performance, followed by Co2CS, CSPSO, and OLCS with the help of the final rank.

Table 3 Solution accuracy obtained by CSPSO, Co2CS, OLCS, and HCCSPSO

Fun	*CSPSO*		*CO2CS*	
	$AVG_{Er} \pm SD_{Er}$	*Rank*	$AVG_{Er} \pm SD_{Er}$	*Rank*
F_1	2.65e-28 ± 2.67e-28	3	4.10e-30 ± 2.86e-29	2
F_2	1.41e-11 ± 2.68e-10	1	1.26e-09 ± 2.11e-09	2
F_3	8.01e+05 ± 6.49e+05	2	6.63e+05 ± 3.40e+05	1
F_4	5.93e+01 ± 4.39e+01	1	3.93e+02 ± 1.15e+03	3
F_5	3.25e+03 ± 9.33e+02	2	7.64e+03 ± 1.89e+03	4
F_6	6.56e+00 ± 1.78e+01	3	5.24e+00 ± 3.22e+00	2
F_7	2.22e-02 ± 1.20e-15	3	1.56e-02 ± 1.19e-02	2
F_8	2.09e+01 ± 5.62e-02	3	2.06e+01 ± 2.06e-01	1
F_9	1.57e+02 ± 2.21e+01	4	9.68e+00 ± 3.20e+00	2
F_{10}	2.52e+02 ± 5.86e+01	4	1.68e+02 ± 5.03e+01	3
Ave. R	2.6		2.2	
Final. R	3		2	

Table 3 Solution accuracy obtained by CSPSO, Co2CS, OLCS, and HCCSPSO (continued)

Fun	*OLCS*		*HCCSPSO*	
	$AVG_{Er} \pm SD_{Er}$	*Rank*	$AVG_{Er} \pm SD_{Er}$	*Rank*
F_1	2.41e-26 ± 6.21e-26	4	0.00e+00 ± 0.00e+00	1
F_2	5.70e-02 ± 4.79e-02	4	7.81e-09 ± 1.60e-08	3
F_3	2.57e+06 ± 7.13e+05	4	8.79e+05 ± 5.15e+05	3
F_4	2.37e+03 ± 1.23e+03	4	2.96e+02 ± 4.20e+02	2
F_5	2.44e+03 ± 7.31e+02	1	6.39e+03 ± 1.97e+03	3
F_6	2.45e+01 ± 1.99e+01	4	3.01e+00 ± 2.66e+00	1
F_7	4.72E-04 ± 1.13E-03	1	2.28e-02 ± 2.02e-02	4
F_8	2.09e+01 ± 5.31e-02	3	2.07e+01 ± 1.36e-01	2
F_9	3.54e+01 ± 6.64e+00	3	1.80e+00 ± 2.15e+00	1
F_{10}	1.54e+02 ± 3.74e+01	2	1.40e+02 ± 4.60e+01	1
Ave. R	3.0		2.1	
Final. R	4		1	

5 Conclusions and future work

In this paper, by combing with the PSO algorithm, we proposed HCCSPSO aiming at voiding the pseudo-minima produced by CC-based CS. In addition, HCCSPSO used the bi-group strategy. During iteration, the first population was optimised by CC-based CS, and the best solution was randomly embedded in the second population evolved by PSO. The subcomponents of best solution obtained by PSO were also randomly planted in the subcomponents of first population.

We investigated HCCSPSO on ten benchmark functions proposed in the CEC2005 special session on real-parameter optimisation. The results showed that the improvement in the efficiency and the effect of the cooperation strategy, and the promising of HCCSPSO. Additionally, we plan to apply HCCSPSO to more benchmark functions including some real-world optimisation problems for further examinations.

Acknowledgements

This work was supported by the NSFC Joint Fund with Guangdong of China under Key Project U1201258, the Shandong Natural Science Funds for Distinguished Young Scholar under Grant No. JQ201316 and the Natural Science Foundation of Fujian Province of China under Grant No. 2013J01216.

References

Babukartik, R.G. and Dhavachelvan, P. (2012) 'Hybrid algorithm using the advantage of ACO and cuckoo search for job scheduling', *International Journal of Information Technology Convergence and Services*, Vol. 2, No. 4, pp.25–34.

Dai, Y.T., Liu, L.Q., Li, Y. and Song, J.Y. (2014) 'An improved particle swarm optimisation based on cellular automata', *Int. J. of Computing Science and Mathematics*, Vol. 5, No. 1, pp.94–106.

Eberhart, R.C. and Kennedy, J. (1995) 'A new optimizer using particle swarm theory', *Proc. of the 6th International Symposium on Micro Machine and Human Science*, IEEE Press, pp.39–43.

Gao, Y.X., Wang, Y.M. and Pei, Z.L. (2012) 'An improved particle swarm optimization for solving generalized travelling salesman problem', *Int. J. of Computing Science and Mathematics*, Vol. 3, No. 4, pp.385–393.

Ghodrati, A. and Lotfi, S. (2012) 'A hybrid CS/PSO algorithm for global optimization', *Proc. of ACIIDS 2012*, Part III, LNAI, Berlin, Heidelberg, Springer, Vol. 7198, pp.89–98.

Hu, X.X. and Yin, Y.L. (2013) 'Cooperative co-evolutionary cuckoo search algorithm for continuous function optimization problems', *P.R. & A.I.*, Vol. 26, No. 11, pp.1041–1049.

Jansen, T. and Wiegand, R.P. (2004) 'The cooperative coevolutionary (1+1) EA', *Evolutionary Computation*, Vol. 12, No. 4, pp.405–434.

Kennedy, J. and Eberhart, R.C. (1995) 'Particle swarm optimization', *Proc. of the IEEE International Conference on Neural Networks*, IEEE Press, pp.1942–1948.

Li, X.T. and Yin, M.H. (2012) 'Parameter estimation for chaotic systems using the cuckoo search algorithm with an orthogonal learning method', *Chin. Phys. B*, Vol. 21, No. 5, pp.050507-1–050507-5.

Mu, L.L., Zhao, M.Z. and Zhang, C.Z. (2013) 'Quantum particle swarm optimisation based on chaotic mutation for automatic parameters determination of pulse coupled neural network', *Int. J. of Computing Science and Mathematics*, Vol. 4, No. 4, pp.354–362.

Noman, N, and Iba, H. (2008) 'Accelerating differential evolution using an adaptive local search', *IEEE Transaction on Evolutionary Computation*, Vol. 12, No. 1, pp.107–125.

Potter, M.A. and de Jong, K.A. (1994) 'A cooperative co-evolutionary approach to function optimization', *Proc. of the Third Parallel Problem Solving from Nature*, Springer-Verlag,, pp.249–257.

Shi, Y.H. and Eberhart, R.C. (1998) 'A modified particle swarm optimizer', *Proc. of IEEE World Congr. Comput. Intell.*, pp.69–73.

Shi, Y.J., Teng, H.F., and Li, Z.Q. (2005) 'Cooperative co-evolutionary differential evolution for function optimization', *Proceedings of the first International Conference on Natural Computation*, Lecture Notes in Computer Science 3611, Springer-Verlag, Berlin, Germany, pp.1080–1088.

Srivastava, P.R., Khandelwal, R., Khandelwal, S., Kumar, S., and Ranganatha, S.S. (2012) 'Automated test data generation using cuckoo search and tabu search (CSTS) algorithm', *Journal of Intelligent Systems*, Vol. 21, No. 2, pp.195–224.

Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., and Tiwari, S. (2005) *Problem Definitions and Evaluation Criteria for the CEC2005 Special Session on Real-Parameter Optimization*, Technical Report, Nanyang Technological University, Singapore.

Van den Bergh, F. and Engelbrecht, A.P. (2004) 'A cooperative approach to particle swarm optimization', *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, pp.225–239.

Wang, F., He, X.S., Luo, L.G. and Wang, Y. (2011a) 'Hybrid optimization algorithm of PSO and cuckoo search', *Proc. of the 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, IEEE Press, pp.1172–1175.

Wang, H., Wu, Z.J., Rahnamayan, S., Liu, Y. and Ventresca, M. (2011b) 'Enhancing particle swarm optimization using generalized opposition-based learning', *Information Sciences*, Vol. 181, No. 20, pp.4699–4714.

Wang, G.G., Guo, L.H., Duan, H., Liu, L., Wang, H.Q. and Wang, J.B. (2012) 'A hybrid meta-heuristic DE/CS algorithm for UCAV path planning', *Journal of Information & Computational Science*, Vol. 9, No. 16, pp.4811–4818.

Wang, H., Sun, H., Li, C.H., Rahnamayan, S. and Pan, J. (2013a) 'Diversity enhanced particle swarm optimization with neighborhood search', *Information Sciences*, Vol. 223, pp.119–135.

Wang, H., Wang, W.J. and Wu, Z.J. (2013b) 'Particle swarm optimization with adaptive mutation for multimodal optimization', *Applied Mathematics and Computation*, Vol. 221, pp.296–305.

Wang, L.J., Yin, Y.L. and Zhong, Y.W. (2013c) 'Cuckoo search algorithm with dimension by dimension improvement', *Ruan Jian Xue Bao/Journal of Software*, Vol. 24, No. 11, pp.2687–2698.

Wang, W.J. and Wang, H. (2014) 'An improved diversity-guided particle swarm optimisation for numerical optimisation', *Int. J. of Computing Science and Mathematics*, Vol. 5, No. 1, pp.16–26.

Yang, X.S. and Deb, S. (2009) 'Cuckoo search via Levy flight', *Proc. of World Congress on Nature & Biologically Inspired Computing (NaBIC2009)*, IEEE Press, pp.210–214.

Yang, X.S. and Deb, S. (2010) 'Engineering optimisation by cuckoo search', *International Journal of Mathematical Modeling and Numerical Optimisation*, Vol. 1, No. 4, pp.330–343.

Yang, Z.Y., Tang, K. and Yao, X. (2008) 'Large scale evolutionary optimization using cooperative co-evolution', *Information Sciences*, Vol. 178, No. 15, pp.2985–2999.

Zheng, H.Q. and Zhou, Y.Q. (2013) 'A cooperative co-evolutionary cuckoo search algorithm for optimization problem', *Journal of Applied Mathematics*, Article ID 912056, 9p, doi:10.1155/2013/912056.