Saudi Journal of Biological Sciences (2017) 24, 695–702

King Saud University

Saudi Journal of Biological Sciences

www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

A novel comprehensive learning artificial bee colony optimizer for dynamic optimization biological problems

Weixing Su [a], Hanning Chen [a,*], Fang Liu [a], Na Lin [b], Shikai Jing [b,c], Xiaodan Liang [a], Wei Liu [d]

[a] School of Computer Science and Software, Tianjin Polytechnic University, Tianjin 300387, China
[b] Beijing Shenzhou Aerospace Software Technology Co. Ltd., Beijing 110000, China
[c] School of Mechanical Engineering, Beijing Institute of Technology, Beijing 110081, China
[d] College of Information and Technology, Jilin Normal University, Siping 136000, China

Received 23 October 2016; revised 23 December 2016; accepted 7 January 2017
Available online 21 February 2017

KEYWORDS

Artificial bee colony;
Dynamic optimization;
Powell's search;
Crossover operation;
Life-cycle

Abstract There are many dynamic optimization problems in the real world, whose convergence and searching ability is cautiously desired, obviously different from static optimization cases. This requires an optimization algorithm adaptively seek the changing optima over dynamic environments, instead of only finding the global optimal solution in the static environment. This paper proposes a novel comprehensive learning artificial bee colony optimizer (CLABC) for optimization in dynamic environments problems, which employs a pool of optimal foraging strategies to balance the exploration and exploitation tradeoff. The main motive of CLABC is to enrich artificial bee foraging behaviors in the ABC model by combining Powell's pattern search method, life-cycle, and crossover-based social learning strategy. The proposed CLABC is a more bee-colony-realistic model that the bee can reproduce and die dynamically throughout the foraging process and population size varies as the algorithm runs. The experiments for evaluating CLABC are conducted on the dynamic moving peak benchmarks. Furthermore, the proposed algorithm is applied to a real-world application of dynamic RFID network optimization. Statistical analysis of all these cases highlights the significant performance improvement due to the beneficial combination and demonstrates the performance superiority of the proposed algorithm.
© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

* Corresponding author.
E-mail address: perfect_chn@hotmail.com (H. Chen).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

1. Introduction

Many real-world optimization problems are subject to changing conditions over time, which can be identified as dynamic optimization problems (DOP) (Ha, 2016). In the DOP cases, changes may affect the object function, the problem instance,

http://dx.doi.org/10.1016/j.sjbs.2017.01.044
1319-562X © 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

or constraints, causing that the optimal solutions of such dynamic problem being considered may change over time (Branke, 2001). From this point of view, most of the real world problems have dynamic characteristics, where one or more elements of the under-lying model for a given problem may change over time. This requires optimization algorithms to not only find the global optimal solution under a specific environment but also to continuously track the changing optima over different dynamic environments.

In recent years, investigating swarm intelligence (SI) algorithms for DOPs has attracted a growing interest (Clerc and Kennedy, 2002), due to that SIs are intrinsically inspired from natural or biological evolution, which is always subject to an ever-changing environment, and hence SIs, with proper enhancements, have a potential to be good optimizers for DOPs. Artificial bee colony algorithm (ABC) is one of the most popular members of the family of swarm intelligence, which simulates the social foraging behavior of a honeybee swarm (Karaboga and Basturk, 2007a,b). Due to its simple arithmetic and good robustness, the ABC algorithm has been widely used in solving many numerical optimizations (Karaboga and Basturk, 2007a,b; Biswas et al., 2014) and engineering optimization problems (Karaboga et al., 2007). However, facing up complex dynamic problems, similar to other EAs, ABC algorithm suffers from the following drawbacks (Karaboga and Basturk, 2007a,b): (1) the solution search equation of ABC works well in global exploration but is poor in the exploitation process. (2) With the dimension increasing, the information exchange of each individual is limited in a random dimension, resulting in a slow convergence rate.

Several ABC variants have been developed to improve its optimization performance. One significant improvement is the introduction of PSO-based search equation (Zhu and Kwong, 2010), which allows a powerful global search in the early stage by incorporating the information of the *gbest* solution into ABC. Similarly, Banharnsakun et al. (2011) presented a modified search equation for the onlooker bees. In their method, the new candidate solutions are more likely to be close to the current best solution. Gao et al. (2013a, b) proposed an efficient and robust ABC variant based on modified search equation and orthogonal learning strategies, which demonstrated its high effectiveness and efficiency. Another interesting approach by (Gao et al., 2013a,b) is using the Powell's method as a local search tool to enhance the exploitation of the algorithm. In this method, ABC good at exploration ensures the search is less likely to be trapped in local optima while it enjoys the merits of fine local search by Powell's method. Hybridization of ABC with other operators has also been studied widely. For example, Kang et al. (2011) used the Rosenbrock's rotational direction method to implement the exploitation phase and proposed the Rosenbrock ABC algorithm. Coelho and Alotto (2011) developed a novel alternative search equation in which a parameter is responsible for the balance between the Gaussian and the uniform distribution.

Inspired by previous works, this paper presents a novel optimization algorithm called comprehensive learning artificial bee colony optimizer (CLABC), which synergizes the idea of extended life-cycle evolving model with a pool of local searching strategies (Liu, 2013). The main motive of CLABC is to enrich artificial bee foraging behaviors in ABC model by combining population initialization based on orthogonal Latin squares approach, Powell's pattern search method, life-cycle, and crossover-based social learning strategy, which contributes in the following aspects:

(1) The orthogonal Latin squares approach can be used for artificial bee colony initialization to cover the search space with balanced dispersion and neat comparability.
(2) The crossover operation, which helps bees exchange more information after the early stage of the algorithm. In this case, the neighbor bees with higher fitness can be chosen to crossover, which effectively enhances the global search ability.
(3) Powell's local search method enables the bee exploit around promising area while avoiding search stagnation.
(4) Life-cycle, which results in a dynamic population. This means that, the bee can reproduce and quit adaptively throughout the foraging process and the population size varies as the algorithm runs in the dynamic landscapes.

This work adopted the moving peaks benchmark (MPB) to illustrative the inherent adaptive mechanism in the proposed algorithm of surviving in a changing environment. The proposed CLABC has been compared with its classical counterpart, the classical ABC algorithm (Karaboga, 2005) over dynamic benchmarks with respect to the statistical performance measures of solution quality and convergence speed.

The rest of the paper is organized as follows. In Section 2, the proposed comprehensive learning artificial bee colony (CLABC) algorithm is given. Section 3 presents the experimental studies of the proposed CLABC and the other algorithms with descriptions of the involved benchmark functions, experimental settings, and experimental results. Finally, Section 4 outlines the conclusion.

2. Comprehensive learning artificial bee colony algorithm

The main procedures of CLABC, including orthogonal Latin squares population initialization, Powell's pattern search, life-cycle, and crossover-based social learning strategies, are details as follows.

2.1. Population initialization based on orthogonal Latin squares approach

The orthogonal Latin squares approach can be used for population initialization to cover the search space with balanced dispersion and neat comparability. Suppose a population consisting of N individuals (or food sources)

has to be initialized in the D-dimensional search space, a orthogonal table $L_N(t^D)$ is designed deliberately where N represents the number of initial solutions or table rows, t is the factor level of orthogonal table, D donates the dimension of search space or the number of orthogonal arrays. Generally, this approach has a merit of obtaining optimal space coverage by consuming comparatively less tests, which has been proved theoretically by relevant theorem of non-parametric statistics in (Math, 1975). The main steps of population initialization are listed in *Algorithm 1*.

Algorithm 1. Population initialization using orthogonal table

Step 1: Calculate the discrete point yij of each initial solution:

$$y_{ij} = lb_i + (j-1)(ub_i - lb_i)/(N-1),$$

where i=1,2,…,N, ,j=1,2,…,D, lb_i and ub_i are the minimum and maximum boundary values of each variable.

Step 2: Compute initial solutions according to the orthogonal Latin squares approach:

x_{ij}=y_{jk}, i=1,2…,N, ,j=1,2,…,D.

where k=(i+j-1) mod N;

And if k==0, k=N.

Step 3: Construct the orthogonal table $L_N(t^D)$ with N initial solutions;

Step 4: Population with N individuals is initialized as X_i= (x_{i1}, x_{i2},…,x_{iD}), i=1, 2, … , N.

Then the initial solution (i.e., X_i = (x_{i1}, x_{i2},…,x_{iD}), i = 1,2,…,N) obtained by *Algorithm 1* is endowed with promising balanced dispersion and neat comparability.

2.2. Powell's pattern search

Powell's search, namely Powell's conjugate gradient descent method, is an extension of basic pattern search method to speed up the convergence of complex nonlinear problems, in which one merit is that the function need not be differentiable, and no derivatives are taken. The method pursues the minimum of the function by a bi-directional search along each search vector, in turn. The new position can then be represented as a linear combination of the search vectors. The new displacement vector becomes a new search vector, and is added to the end of the search vector list. Coincidently, the search vector which contributed most to the new direction (i.e. the one which was most successful), is deleted from the search vector list. The algorithm iterates an arbitrary number of times until no significant improvement can be made. The Powell's method algorithm is given as follows.

Algorithm 2. Powell's search:

Step 1: Initialize the starting point X_1, independent vectors d_i =e_i (i=1,...,D), the tolerance for stop criteria ε , set $f(1)$=$f(X_1)$, X_c=X_1, k=1

Step 2: **While** (stopping criterion is not met, namely $|\Delta f| > \varepsilon$) **do**

Step 3: **For** i=1 to D **do**

If (k>=2) **then**

Step 4: $d_i = d_{i+1}$

End If

Step 5: $X_{i+1} = X_i + \lambda_i$, is determined by Minimizing $f(X_{i+1})$

End for

Step 6: $d_{i+1} = \sum_1^D \lambda_i^* d_i = X_{D+1} - X_D$, $X_c(k+1) = X_{D+1} + \lambda_k d_{i+1}$, $f(k+1) = f(X_c(k+1))$

Step 7: $k = k+1$, $X_1 = X_c(k)$, $\Delta f = f(k) - f(k-1)$

End While

2.3. Crossover operation

As described in canonical ABC (Karaboga and Basturk, 2007a,b), the search equation for generating the positional change is much like a blind mutation operator to search in a randomly selected dimension of a randomly chosen bee, which means that the information exchange is restricted to a narrow local scope. This causes that, as the dimension of solved problems grows exponentially, the algorithm using this information exchange based on single dimension of single bee will suffer from the following drawback of premature convergence at the early generations. On the other hand, the neighbor bee and dimension are both chosen randomly, which results in the good individuals with higher fitness may likely be abandoned.

To address this concerning issue, inspired by genetic algorithm, the crossover-based comprehensive learning is employed in the bee hive. To benefit from information about positions of good sources foraged by the employed and onlooker bees, a given number of elites form excellent employed bees are subjected to crossover operation. The underlying idea behind this method is to facilitate better search in complex dynamic search space as opposed to classical ABC that perturbs a single parameter. The steps for crossover operation phase are given as follows and the schematic diagram is illustrated in Fig. 1.

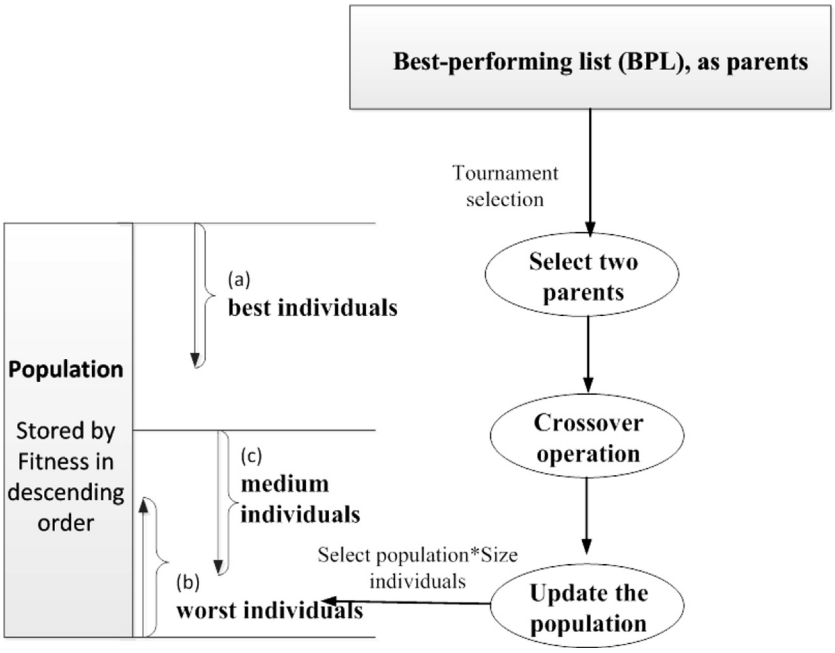

Fig. 1 The information exchange mechanism based on crossover operation.

Algorithm 2. Crossover operation

Step 1. *Select elites to the best-performing list (BPL)*.A set of competent individuals from current food sources are selected to construct the best-performing list (BPL), the ones with higher fitness have larger probability to be selected.The size of BPL is equal with current population size. These individuals of BPL are regarded as elites. The selection operation tries to mimic the maturing phenomenon in nature, where the generated offspring will become more suitable to the environment by using these elites as parents.

Step 2. *Crossover operation*.To produce well-performing individuals, parents are selected from the BPL's elites only for crossover. To select parents effectively, the tournament selection scheme is used. Firstly, two enhanced elites are selected randomly, and their fitness values are compared to select the elites. The one with better fitness is viewed as parent. Then, another parent is selected in the same way. Two offspring are created by performing crossover on the selected parents. Here adopts a representative crossover method, namely arithmetic crossover, according to which, the offspring is produced by the following equation:

$$s_{new} = rand(0,1) \times parent1 + rand(0,1) \times parent2 \quad (1)$$

where S_{new} is newly produced offspring, parent1 and parent 2 are randomly selected from BPL.

Step 3. *Update with different selection schemes*.Not all current bees are replaced by the elites from BPL, we set a selecting rate CR to determine the replaced individuals. Assuming that population size is S, then the replaced individuals number is S^*CR. For the selected individual Sj, the newly produced offspring S_{new} is then compared with Sj, applying a greedy selection mechanism, in which the better one is remained. We can choose four selecting approaches: selecting the best individuals (i.e. S^*CR individuals), a medium level of individuals, the worst individuals, and randomly individuals.

2.4. Life-cycle model

This work assumes that the computational life-cycle model of bee colony has five major stages, namely the born, forage, reproduction, death, and migration. The bee state transition diagram is shown in Fig. 2.

The bees are born when they are initialized. Then they will forage for nutrient (nectar). We define the nutrient updating formula as:

$$N_i(t+1) = \begin{cases} N_i(t)+1 & if\ fit(X_i^{t+1}) < fit(X_i^t) \\ N_i(t)-1 & else \end{cases} \quad (2)$$

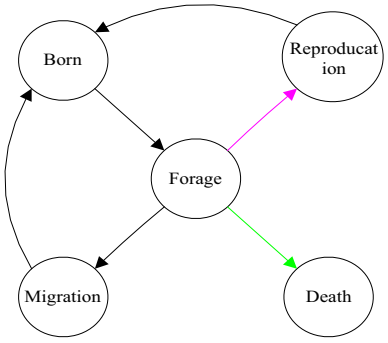

Fig. 2 Bee state transition in life-cycle model of CLABC.

where $fit(X_i^t)$ is the fitness of the i^{th} bee X_i at time t for a minimum problem, $N_i(t)$ is the nutrient obtained by the i^{th} bee X_i at time t. In initialization stage, nutrients of all bees are zero. For each X_i at onlooker bee phase, if the new position is better than the last one, it is regarded that the bee will gain nutrient from the environment and the nutrient is added by one. Otherwise, it loses nutrient in the foraging process and its nutrient is reduced by one. Then the information rate F_i^t deciding to reproduce or die for each bee X_i at time t is computed as:

$$H_i(t) = \frac{fit(X_i^t) - fit_{worst}^t}{fit_{best}^t - fit_{worst}^t} \tag{3}$$

$$F_i^t = \eta \frac{H_i(t)}{\sum_{j=1}^{S^t} H_j(t)} + (1-\eta)\frac{N_i(t)}{\sum_{j=1}^{S^t} N_j(t)}, \eta \subset [0,1] \tag{4}$$

where fit_{worst}^t and fit_{best}^t are the current worst and best fitness of the whole bee colony at time t.

In the foraging process, if the bee X_i converts enough information rate F_i^t as:

$$F_i^t > \max(F_{reproduce}, F_{reproduce} + \frac{(S^t - S)}{F_{adapt}}) \tag{5}$$

it will reproduce an offspring by using best-so-far solution information in search equation of employed and onlooker bees steps:

$$xnew_{i,j} = x_{i,j} + \varphi(x_{best,j} - x_{i,j}) \tag{6}$$

where x_{new} is the new offspring, x_i is the ith bee, x_{best} is best individual of current colony, j is a randomly chosen indexes; ϕ is a random number in range $[-1,1]$.

If the bee enters bad environment, and its information rate drops to a certain threshold as:

$$F_i^t < \min(0, \frac{(S^t - S)}{F_{adapt}}) \tag{7}$$

It will die and be eliminated from the population. Here S is the initial population size and S^t is the current colony size, F_{split} and F_{adapt} are two control parameters used to adjust the bee reproduction and death criterions.

It should be noticed that the population size will increase by one if a bee reproduces and reduce by one if it dies. As a result, the population size dynamically varies in the foraging process. At the beginning of the foraging process, the bee will reproduce when its information rate is larger than $F_{\text{reproduce}}$. In the course of bee foraging, in order to avoid the population size becoming too large or too small, the reproduction and death criterions, namely Eq. (5) and Eq. (7), are delicately designed: if S^t is larger than S, for each F_{adapt} of their differences, the reproduce threshold value will increase by one; if S^t is smaller than S, for each F_{dapt} of their differences, the death threshold value will decrease by one. The strategy is also consistent with the natural law: if the population is too crowded, the competition between the individuals will increase and death becomes common; if the population is small, the individuals are easier to survive and reproduce.

When the nutrient of a bee is less than zero, but it has not died yet, it could migrate with a probability as a scout bee. A random number is generated and if the number is less than migration probability P_e, it will migrate and move to a randomly produced position. Then nutrient of this bee will be reset to zero.

Table 1 Parameters of the CLABC.

$CLABC = (S, D, F_{split}, F_{adapt}, CR, Tp, \eta)$	
S	Population size
D	Dimensions of optimization problem
F_{split}	Reproduction and death criterion
F_{adapt}	Control parameter to adjust reproduction and death criterion
CR	Selection rate
Tp	Parameter to activate Powell's search
	Inertia coefficient

In summary, in order to facilitate the below presentation and test formulation, we define a unified parameters for CLABC model in Table 1.

3. Benchmark test

3.1. Dynamic test function

The moving peak benchmark (MPB) problem has been widely used as a dynamic benchmark problem in the literature (Morrison and De Jong, 1999). Within the MPB, the optima can be varied by three features, i.e., the location, height, and width of the peaks, which can be defined as follows:

$$Z = -\max_{i=1,\ldots,N}\left\{H_i - R_i\sqrt{(X-X_i)^2 + (Y-Y_i)^2}\right\} \tag{8}$$

where N is the number of peak in the environment. H_i indicates the ith peak height, R_i is the slope control variable, and (X_i, Y_i) represents the coordinate of its center. The initialization of parameters is shown in Table 2.

But in our experimental studies, the height and range of slope for each peak are set to be constant, and only the positions of the peaks are changing. In this case, x_{step} and y_{step} are the step sizes in x and y direction respectively. And for each step i, X_{i+1} and Y_{i+1} are calculated as

$$x_{step} = -A_x \times x_{step}(1 - x_{step}) \tag{9}$$

$$y_{step} = -A_y \times y_{step}(1 - y_{step}) \tag{10}$$

$$X_{step} = -X_i \times x_{step} Dx_i \tag{11}$$

$$Y_{i+1} = -Y_i \times y_{step} Dy_i \tag{12}$$

Table 2 Parameter settings.

Parameter	Value
N	15
H_1	0
R_1	0
X_1	0.5
Y_1	0.5
H_i	[1,10]
R_i	[8,20]
X_i	[−1, 1]
Y_i	[−1,1]
i	2,...,N

where A_x and A_y are both a constant, respectively. D_{xi} and D_{yi} can be assigned 1 or -1 with probability 0.5, respectively. An example of the dynamic environment, generated with MPB, in four steps, is illustrated in Fig. 3.

The experiments based on MPB are designed to evaluate the adaptability of CLABC for various dynamic environments. Various environmental changes are used in our simulation studies, which are divided into three ranges:

(1) Range I – Slow level of environmental changes;
(2) Range II – Intermediate level of environmental changes;
(3) Range III – High level of environmental changes.

The level of changes is reflected by the frequency of changes in the environment, which is defined as a probability f. For the environmental changes classified in Range I, $f \in [0, 0.01]$, in Range II, $f \in [0.05, 0.2]$ and in Range III $f \in [0.3, 0.8]$. In our simulation studies, f indicates the probability of occurrence of environmental changes after each foraging step.

3.2. Performance evaluation criteria

3.2.1. Average best over a period (ABP)

The search ability of algorithms is one of the most important factors in optimization domain. In a dynamic environment, the optimum might be varying over time, which means that it is insufficient to only evaluate the fitness found after a certain number of generations.

There is an alternative as evaluation criteria, which averages over the best solution found at each step during a period between two environmental changes (Tang et al., 2006). It is concerned with an average of the best values, denoted by average best, found over a period T_i, where T_i denotes the ith period. This average best over a period is denoted as ABP, which represents the best fitness for a given period T_i. Let S_i be the first step of T_i, E_i be the last step. Thus, ABP is formulated as:

$$ABP_{Ti} = \frac{1}{E_i - S_i} \sum_{t=Si}^{Ei} f(t)^* \tag{13}$$

3.2.2. Accuracy

To obtain the accuracy of algorithm A in function F, firstly, we calculate accuracy in each step t,

$$Accuracy_{F,A}(t) = \frac{f_{F,A}(t) - V_{wF,A}(t)}{V_{oF,A}(t) - V_{wF,A}(t)} \tag{14}$$

Then, the accuracy as a whole is defined as:

$$Accuracy_{F,A}(t) = \frac{1}{N} \sum_{t=1}^{N} Accurancy_{F,A}(t) \tag{15}$$

where V_w and V_o are the worst and optimum value respectively, N is the number of steps.

3.2.3. Stability

Similar to the definition of accuracy, the stability is defined as follows:

$$Stability_{F,A}(t) = Accurancy_{F,A}(t) - Accurancy_{F,A}(t-1) \tag{16}$$

$$Stability_{F,A}(t) = \frac{1}{N} \sum_{t=1}^{N} \max\{0, Stability_{F,A}(t)\} \tag{17}$$

3.3. Parameters settings for the involved algorithms

With and without the orthogonal Latin squares population initialization strategy, two CLABC variants are tested, as seen in Table 3 with their experimental settings. To fully evaluate the performance of the proposed CLABC variants, the classical

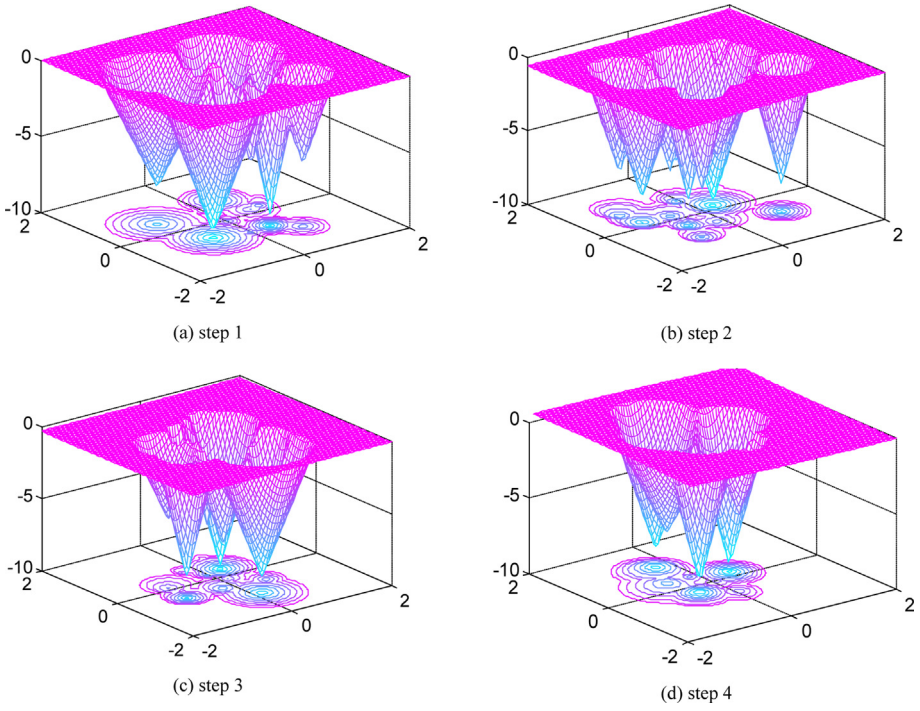

Fig. 3 A MPB example of environmental changes.

Table 3 Two versions of CLABC algorithms and their parameter settings.

Algorithm	Strategies	Parameters						
		S	D	η	CR	$F_{\text{reproduce}}$	F_{adapt}	Tp
CLABC-1	Life-cycle; crossover Powell's search	50	30	0.6	1	(Lb-Ub)-2	50	5
CLABC-2	Orthogonal Latin squares population initialization; life-cycle; crossover; Powell's search	50	30	0.6	1	(Lb-Ub)-1	50	5

artificial bee colony algorithm (ABC) (Karaboga and Basturk, 2007a,b) was used for comparison.

In all experiments in this section, the values of the initial population size used in each algorithm were chosen to be the same as 50. The number of function evaluations (FEs) was used as a measure criterion. All algorithms were terminated after 100,000 FEs. For the classical ABC algorithm, as referred to (Karaboga and Basturk, 2007a,b), the limit parameter of ABC is set to be $SN \times D$, where D is the dimension of the problem, and is set 30. SN is the total number of employed bees and onlooker bees, and can be set 50.

3.4. Results for the dynamic MPB functions

To investigate their performances in dynamical different-level environments, the frequency of changes with $f = (0.001, 0.01)$, $f = (0.05, 0.1)$ and $f = (0.5, 0.8)$ are selected, which fall into from Range I to Range III, respectively. Accordingly, the dynamic accuracy and stability results of CLABC-1, CLABC-2 and ABC are given in Table 4 and Table 5, respectively.

From the results, both CLABC-1 and CLABC-2 performed powerful when $f = 0.001$. However, CLABC-1 performed a little worse than ABC when $f = 0.01$, which belonged Range I. This means that the CLABC didn't exhibit significant advantage when the dynamic environmental varied slowly. When $f = 0.05$ and $f = 0.1$ in Ranges II, both CLABC-1 and CLABC-2 achieved satisfactory performance. In contrast, the performance of ABC decreased obviously, this is due to the CLABC can react to the dynamical environmental changes effectively. When $f = 0.5$ and $f = 0.8$ in Range III, it is not surprising that both the CLABC-1 and CLABC-2 still maintain high accuracy and stability, which can be interpreted that, in this case, the dynamical environment changes rapidly and the strategies of life-cycle and crossover operation play an important role in improving the ability of involved algorithm to search in dynamical functions.

The differences between the compared algorithms in terms of accuracy on these dynamic benchmarks suggest that the CLABC is better at a fine-gained search than its counterpart ABC. The key difference is mainly due to Powell's search and crossover operator in the CLABC. The ABC with crossover operator acts as the main optimizer for quickly searching the near-optimal areal while the Powell's method performs fine tuning the best solutions in a certain time interval. However, for the dynamic optimization cases, we need not only the high optimization accuracy and computation robustness, but also a faster solution speed. Fortunately, the life-cycle mode enables the population size and bee colony behaviors of CLABC can be dynamically adaptive to the complexity of the dynamic objective functions, which reduces the computational complexity of the optimization process.

Table 4 Accuracy comparison.

Freq	CLABC4	CLABC7	ABC	ABC & CLABC7	CLABC4 & CLABC7
0.001	0.301559	0.082557	0.427601	0.175878	1.705158
0.01	0.39025	0.220559	0.44092	0.049244	0.453024
0.1	0.395654	0.139605	0.418915	0.025615	0.909054
0.05	0.324041	0.120704	0.432145	0.143515	1.124204
0.5	0.396415	0.091319	0.360905	−0.03961	1.264104
0.8	0.395924	0.130154	0.337033	−0.06569	0.721053

Table 5 Stability comparison.

Freq	CLABC4	CLABC7	ABC	ABC & CLABC7	CLABC4 & CLABC7
0.001	0.01396	0.03214	0.001029	−0.41856	−0.4366
0.01	0.009465	0.04096	0.002116	−0.3466	−0.4276
0.1	0.012598	0.049604	0.011733	−0.04099	−0.34194
0.05	0.009299	0.042643	0.006278	−0.12593	−0.38243
0.5	0.014894	0.042692	0.037387	0.693567	−0.0588
0.8	0.018464	0.049709	0.037448	0.460629	−0.10788

4. Conclusions

In order to improve classical artificial bee colony algorithm for complex dynamic optimization problems efficiently, this paper proposes a novel comprehensive learning artificial bee colony algorithm (CLABC) by combining several optimal foraging approaches, namely the orthogonal Latin squares population initialization, Powell's search, crossover operation, and life-cycle strategies. This paper substantially extends the previous work on the ABC algorithms that can be distinguished from it from three aspects: (1) to refine the local search behaviors when a bee finds promising area, the Powell's search is incorporated to emphasize the exploitation process. (2) to refine the bee-to-bee communication mechanism based on crossover that enhances the information exchange among bee colony, dealing with the global exploration. (3) to refine the bee life-cycle behaviors, which contains born, forage, reproduction, death, and migration states.

Acknowledgements

This research is partially supported by National Natural Science Foundation of China under Grant 61305082, 51378350, 5157518, 61602343 and 51607122.

References

Banharnsakun, A., Achalakul, T., Sirinaovakul, B., 2011. The best-so-far selection in artificial bee colony algorithm. Appl. Soft Comput. 11, 2888–2901.

Biswas, S., Das, S., Debchoudhury, S., Kundu, S., 2014. Co-evolving bee colonies by forager migration: A multi-swarm based Artificial Bee Colony algorithm for global search space. Appl. Math. Comput. 232, 216–234.

Branke, J., 2001. Evolutionary approaches to dynamic environments — updated survey, In: GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems, 27–30

Clerc, M., Kennedy, J., 2002. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. 6, 58–73.

Coelho, L.S., Alotto, P., 2011. Gaussian artificial bee colony algorithm approach applied to Loneys solenoid benchmark problem. IEEE Trans. Magn. 47, 1326–1329.

Gao, W.F., Liu, S.Y., Huang, L.L., 2013a. A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Trans. Cybern. 43, 1011–1024.

Gao, W.F., Liu, S.Y., Huang, L.L., 2013b. A novel artificial bee colony algorithm with Powell's method. Appl. Soft Comput. 13, 3763–3775.

Ha, J., 2016. Fitness dance video event analysis based on dynamic programming fusion multimoding. J. Mech. Eng. Res. Dev. 39, 270–277.

Kang, F., Li, J.J., Ma, Z.Y., 2011. Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf. Sci. 181, 3508–3531.

Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., Akay, B., Ozturk, C., 2007. Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In: International Conference on Modeling Decisions for Artificial Intelligence, 318–329.

Karaboga, D., Basturk, B., 2007. Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In: International Fuzzy Systems Association World Congress. 789–798.

Karaboga, D., Basturk, B., 2007b. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Global Optim. 39, 459–471.

Liu, Z.L., 2013. Environmental kuznets curve: a new computational test and modeling. IJACT 5, 551–558.

Math. Stat. Res. Group, 1975. Chinese Acad. Sci., Orthogonal Design (in Chinese). People Education Pub, Beijing, China.

Morrison, R.W., De Jong, K.A., 1999. A test problem generator for non – stationary environments. In: Proceedings of the 1999 IEEE Congress on Evolutionary Computation, IEEE Press, 2047–2053.

Tang, W.J., Wu, Q.H., Saunders, J.R., 2006. Bacterial foraging algorithm for dynamic environments, Evolutionary Computation. IEEE Congress on. IEEE, 1324–1330.

Zhu, G., Kwong, S., 2010. Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl. Math. Comput. 217, 3166–3173.