J. Cent. South Univ. (2018) 25: 107−120
DOI: https://doi.org/10.1007/s11771-018-3721-z

Springer

An enhanced artificial bee colony optimizer and its application to multi-level threshold image segmentation

GAO Yang(高扬)[1], LI Xu(李旭)[2], DONG Ming(董明)[1], LI He-peng(李鹤鹏)[3]

1. Academy of Information Technology, Northeastern University, Shenyang 110018, China;
2. Benedictine University, Lisle, IL, US;
3. Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China

Abstract: A modified artificial bee colony optimizer (MABC) is proposed for image segmentation by using a pool of optimal foraging strategies to balance the exploration and exploitation tradeoff. The main idea of MABC is to enrich artificial bee foraging behaviors by combining local search and comprehensive learning using multi-dimensional PSO-based equation. With comprehensive learning, the bees incorporate the information of global best solution into the solution search equation to improve the exploration while the local search enables the bees deeply exploit around the promising area, which provides a proper balance between exploration and exploitation. The experimental results on comparing the MABC to several successful EA and SI algorithms on a set of benchmarks demonstrated the effectiveness of the proposed algorithm. Furthermore, we applied the MABC algorithm to image segmentation problem. Experimental results verify the effectiveness of the proposed algorithm.

Key words: artificial bee colony; local search; swarm intelligence; image segmentation

Cite this article as: GAO Yang, LI Xu, DONG Ming, LI He-peng. An enhanced artificial bee colony optimizer and its application on multilevel threshold image segmentation [J]. Journal of Central South University, 2018, 25(1): 107–120. DOI: https://doi.org/10.1007/s11771-018-3721-z.

1 Introduction

Image segmentation is considered useful method to separate objects from background that has distinct gray levels. Among existing segmentation techniques, multi-level threshold is a simple but effective tool and requires multiple threshold values to accomplish segmentation. This approach can be classified into optimal threshold methods [1–4] and property based threshold methods [5–7]. The first category searches for the optimal thresholds which make the threshold classes on the histogram reach the desired characteristics. The second category detects the thresholds by measuring some property of the histogram. Property-based threshold methods are fast and suitable for the case of multilevel threshold, while the number of thresholds is hard to be determined.

Several algorithms have been proposed in literatures for optimal threshold [8–12]. In Refs. [4, 8, 9], some novel methods, derived from optimizing an objective function for bi-level and multi-level threshold, were proposed. These methods suffer from a common drawback that the computational complexity raises exponentially when the problem is extended to multi-level threshold. Recently, swarm intelligence (SI) algorithms have been introduced to image segmentation [12–16]. Among them, artificial bee colony algorithm (ABC) is one popular member of the SI family [17]. Due to its

Foundation item: Projects(6177021519, 61503373) supported by National Natural Science Foundation of China; Project(N161705001) supported by Fundamental Research Funds for the Central University, China
Received date: 2016−04−25; **Accepted date:** 2017−12−01
Corresponding author: GAO Yang, Master; Tel: +86−13842023316; E-mail: gaoy@mail.neu.edu.cn; ORCID: 0000-0002-1858-6324

good robustness, the ABC has been widely employed to solve many engineering optimization problems [18–21]. Especially, Refs. [19, 20] have proposed and developed the novel and effective

ABC variants by using a hybridization of life-cycle and optimal search strategies have obtained significant performance improvement. However, when tackling complex problems, these ABCs still suffer from the drawbacks of poor exploitation [18].

Aiming to conquer above drawbacks to some extent, this paper presents a modified artificial bee colony algorithm (MABC) for image segmentation. In our proposed MABC model, the local search operation is activated when a bee finds promising area and the comprehensive learning is used to facilitate more information shared in bee colony. By this hybrid mechanism, the proposed MABC can be claimed very powerful due to the fact that the exploitation and exploration can be elaborately balanced.

2 Standard ABC algorithm

The recently introduced artificial bee colony (ABC) algorithm is motivated by intelligent social behaviors of three types of bees [17]. In ABC, there are three groups of bees: employed bees, onlookers and scouts. The employed bees explore the food sources and transmit related information to onlooker bees. The onlooker bees select good food sources, and these food sources with higher quality will have a bigger probability to be chosen. If a food source found by employed bee is exhausted, the corresponding employed bee will be transformed to a random scout. The detailed procedures are given as follows.

Step 1: Initialization

In initialization phase, a group of food sources representing possible solutions are generated randomly by the following equation:

$$x_{i,j} = x_j^{\min} + rand(0,1)(x_j^{\max} - x_j^{\min}) \tag{1}$$

where i=1, 2, ⋯, S_N; j=1, 2, ⋯, D; S_N is the population size (the number of solutions); D donates the number of variables, i.e. problem dimension; $x_j^{\min}$ and $x_j^{\max}$ represent the lower upper and upper bounds of the jth variable, respectively.

Step 2: Sending employed bees

In this phase, the neighbor food source (candidate solution) can be generated from the old food source of each employed bee in its memory using the following expression:

$$v_{i,j} = x_{i,j} + \varphi(x_{i,j} - x_{k,j}) \tag{2}$$

where x_k is a randomly selected individual as a neighbor bee and is different from current bee; $x_{i,j}$ is another randomly chosen index donating a random dimension; ϕ is a number randomly falling into [−1,1].

Step 3: Sending onlooker bees

In this phase, an onlooker bee selects a food source lying on the probability value linked with that corresponding food source; P_i can be defined as following expression:

$$P_i = \frac{\text{fitness}_i}{\sum_{j=1}^{S_N} \text{fitness}_j} \tag{3}$$

where fitness$_i$ donates the fitness value of the jth solution.

Step 4: Sending scout bees

In the scout bees' phase, once a food source cannot be ameliorated further during a predetermined cycle (defined as "limit" in ABC), the food source should be replaced with a new one while the employed bee associated with it subsequently becomes a scout. The new food source is generated randomly according to Eq. (1).

Those procedures from Step 2 to Step 4 will be carried out repetitively until the termination condition is met.

3 Modified artificial bee colony algorithm

3.1 Local search

The Powell's local search algorithm is an extension of basic pattern search method, and has a merit of tackling the non-differentiable objective functions without derivatives [22]. This algorithm searches the objective optima bi-directionally along each vector, alternately. Then the new point is donated as a linear combinational vector as a new member added to the search vector list. Accordingly, the most successful search vector with most contribution to the new direction is removed from this list. This process is iterated until no significant improvement is achieved. The detailed implantation of this algorithm can refer to Ref. [22].

3.2 Comprehensive learning based on multi-dimensional best-solution information

In the original ABC version, the search equation (i.e., Eq. (2)) is used to generate a new position by a random-dimension disturbance, whereas this approach is similar to a blind mutation operator. That is, this equation drives the old individual bee towards (or away from) its randomly-selected neighbor at a random dimension. This inevitably causes the inefficiency of information exchange at the individual-level and population-level because the useful information of elites is not utilized fully and the dimension of learning is not enough.

Inspired by the social learning in PSO model [23], a new learning strategy is employed in search equation of ABC (i.e., Eq. (2)). To learn fully from the best individual in current bee population, assume that individuals exchange information to other individuals in a full-dimension manner. Specifically, in the employed or onlooker stage, the foraging direction of a bee is governed by the information combination of its randomly-selected neighbor and the best individual in the population (i.e., *g*best). And this search equation is modified as follows:

$$x_{\mathrm{new}_i} = x_i + l_1(x_{g\mathrm{best},j} - x_{i,j}) + l_2(x_i - x_k) \tag{4}$$

where $x_{g\mathrm{best}}$ is the best member from current population; x_k is randomly chosen neighbor individual (note that k is different with i); l_1 and l_2 is a random number within the scope of [−1, 1].

According to Eq. (4), the *g*best term can drive the new candidate solution towards the global best solution, as well as the full-dimension learning can enhance the efficacy of information exchange.

3.3 Proposed algorithm

The balance between exploration of the search space and exploitation of potentially good solutions is considered a fundamental problem in population-based optimization algorithms. In practice, the exploration and exploitation contradict with each other. By using the local search and comprehensive learning, the proposed MABC will act as the main optimizer for searching the near-optimal position while the local search will make fine tune the best solutions obtained by the MABC in each iteration. The main steps of the proposed algorithm are given as the following processes. The following is the proposed MABC algorithm.

Step 1) Initialization.

Step 1.1) Randomly generate S_N food sources in the search space to form an initial population by Eq. (1).

Step 1.2) Evaluate the fitness of each bee.

Step 1.3) Set the maximum cycle (*LimitC*).

Step 2) Iteration=0.

Step 3) Employ bee phase. Loop over each food source.

Step 3.1) Generate a candidate solution V_i by Eq. (4) and evaluate $f(V_i)$.

Step 3.2) Greedy selection and memorize the better solution.

Step 4: Calculate the probability value P_i by Eq. (3).

Step 5: Onlooker bee phase.

Step 5.1) Generate a candidate solution V_i by Eq. (4) and evaluate $f(V_i)$.

Step 5.2) Greedy selection and memorize the better solution.

Step 6: Powell's search phase.

If mode (Iteration, T_p) ==0, randomly choose $m\in\{1, \cdots, S_\mathrm{N}\}$ that has to be different from the best one, X_best, and generate a new solution V_s by Eq. (4). Use the V_s as a starting point and generate a new solution V_m by Powell's method as illustrated in standard ABC algorithm.

Step 7) Iteration= Iteration +1.

Step 8) If the iteration is greater than *LimitC*, stop the procedure; otherwise, go to Step 3).

Step 9) Output the best solution achieved.

4 Benchmark test

In the experimental studies, according to the no free lunch (NFL) theorem [24], a suit of 15 benchmark functions are employed to fully evaluate the performance of the MABC algorithm without a biased conclusion towards some chosen problems [25–28]. The involved benchmark functions can be classified as basic continuous benchmarks (f_1–f_8), CEC2005 benchmarks (f_9–f_{15}). The formula for each basic benchmarks and CEC2005 test functions is shown in Tables 1 and 2. In order to compare the different algorithms fairly, the number of function evaluations (FEs) is adopted as a time measure substituting the number of iterations, due to the fact that the algorithms do differing amounts of work in their inner loops.

Table 1 Classical test suite

Name	Function	Limit
Sphere (f_1)	$f_1(x)=\sum_{i=1}^{D}x_i^2$	$x_i\in[-5.12,\ 5.12]^D$
Rosenbrock (f_2)	$f_2(x)=\sum_{i=1}^{D}100\times(x_{i+1}-x_i^2)^2+(1-x_i)^2$	$x_i\in[-2.048,\ 2.048]^D$
Quadric (f_3)	$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	$x_i\in[-65.536,\ 65.536]^D$
Sin (f_4)	$f_4(x)=\frac{\pi}{n}\left\{10\sin^2(\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2\left[1+10\sin^2(\pi x_{i+1})\right]+(x_n-1)^2\right\}$	$x_i\in[-10,\ 10]^D$
Rastrigrin (f_5)	$f_5(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)\right]+10$	$x_i\in[-5.12,\ 5.12]^D$
Schwefe (f_6)	$f_6(x)=418.9829D+\sum_{i=1}^{D}-x_i\sin(\sqrt{\lvert x_i\rvert})$	$x_i\in[-500,\ 500]^D$
Ackley (f_7)	$f_7(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right]+20+\mathrm{e}$	$x_i\in[-32,\ 32]^D$
Griewank (f_8)	$f_8(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	$x_i\in[-5.12,\ 5.12]^D$

Table 2 CEC 2005 test suite

Name	Function	Limit
Shifted sphere (f_9)	$f_9(x)=\sum_{i=1}^{D}z_i^2+f_{\text{bias1}},\ z=x-o$	$x_i\in[-100,\ 100]^D$
Shifted Schwefel's problem 1.2 (f_{10})	$f_{10}(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}z_j\right)^2+f_{\text{bias}},\ z=x-o$	$x_i\in[-100,\ 100]^D$
Shifted Rosenbrock's function (f_{11})	$f_{11}(x)=\sum_{i=1}^{D-1}\left[100(z_i^2-z_{i+1})^2+(z_i^2-1)^2\right]+f_{\text{bias}}$	$x_i\in[-100,\ 100]^D$
Shifted rotated Griewank without bounds (f_{12})	$f_{12}(x)=\sum_{i=1}^{D}\frac{z_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{z_i}{\sqrt{i}}\right)+1+f_{\text{bias}}$	No bounds
Shifted rotated Ackley (f_{13})	$f_{13}(x)=-\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}z_i^2}\right)-\exp\left[\sum_{i=1}^{D}\cos(2\pi z_i)\right]+$ $20+e+f_{\text{bias}},\ z=(x-o)*M$	$x_i\in[-32,\ 32]^D$
Shifted Rastrigin (f_{14})	$f_{14}(x)=\sum_{i=1}^{D}\left[z_i^2-10\cos(2\pi z_i)+10\right]+f_{\text{bias}},\ z=x-o$	$x_i\in[-5,\ 5]^D$
Shifted rotated Rastrigin (f_{15})	$f_7(x)=(1+0.1\times rand[-1,1])\sum_{i=1}^{D}x_i^2$	$x_i\in[-5,\ 5]^D$

4.1 Parameters settings for involved algorithms

Experiment was conducted to compare with original artificial bee colony algorithm (ABC) [18], canonical PSO with constriction factor (PSO) [23], genetic algorithm with elitism (EGA) [29] and covariance matrix adaptation evolution strategy (CMA-ES) [30]. All algorithms were run 30 times respectively on each benchmark and the maximum evaluation number (FEs) was set at 100000. For involved benchmarks, the dimensions are all set as 30. All the control parameters for the EA and SI algorithms are set to be default of their original literatures: initialization conditions of CMA-ES are the same as those in Ref. [30]; the number of offspring candidate solutions generated per time step is $\lambda=4\mu$, where μ is a adjustable parameter defined in Ref. [30]; the limit parameter of ABC is set to be $S_N\times D$, where D is the dimension of the problem and SN is the number of employed bees [18]. For canonical PSO, the learning rates c_1 and c_2 are both set as 2.05 and the constriction factor $\chi=0.729$ [23]. For EGA, crossover rate of 0.8, mutation rate of 0.01, and the global elite with a rate of 0.06 are adopted [29]. For the proposed MABC, the control parameter Tp can be empirically set as 90 in the experiments and other parameters can be referred the setting of original ABC [18].

4.2 Numerical results and comparison

4.2.1 Results on classical benchmarks

The means and stand deviations of the 30 run times of involved algorithms on classical test functions are listed in Table 3 where the best results are highlighted in bold. From Table 3, MABC and ABC obtain satisfactory results on the unimodal f_1, f_2, f_3 in terms of accuracy and convergence. MABC performs a little worse than ABC on these functions, but significantly better than other algorithms. f_5–f_8 are the most commonly used test multimodal functions, and an algorithm can be easily trapped in a local minimum. As expected, the MABC gets more favorable results than the compared algorithms on all these cases. The superior performance of MABC on these multimodal functions suggests that MABC is good at a fine-gained search. The performance improvement is mainly due to its Powell's search and improved search equation in MABC. That is, the ABC guided by so-far-best information will act as the main optimizer for exploration while the Powell's method aims to fine exploitation. From computation results on these classical functions, MABC performs most powerful on most test cases due to its using the proposed foraging strategies.

4.2.2 Results on CEC2005 benchmarks

Benchmarks f_9–f_{15} from CEC 2005 test bad are employed in this section and correlative computation results are presented in Table 4. From these results, it can be observed that MABC performs best on five out of the seven functions. ABC and CMA-ES achieve similar ranking, only worse than MABC. It is clearly visible and proven that MABC performs more powerful on CEC 2005

Table 3 Results obtained by all algorithms on classical test suite

Function		MABC	ABC	PSO	CMA-ES	GA
f_1	Mean	1.7198×10^{-11}	1.6191×10^{-11}	2.2059×10^{-4}	1.8626×10^{-6}	1.4077×10^{-3}
	Std	2.9544×10^{-11}	2.2532×10^{-11}	4.8068×10^{-4}	4.2273×10^{-4}	1.9295×10^{-3}
	Rank	2	**1**	4	3	5
f_2	Mean	5.7139×10^{-15}	1.0238×10^{-4}	2.1456×10^{-4}	6.2334×10^{-3}	5.3335×10^{-2}
	Std	1.2333×10^{-10}	2.0252×10^{-14}	1.0572×10^{-5}	2.1883×10^{-6}	5.0792×10^{1}
	Rank	**1**	2	3	4	5
f_3	Mean	1.4679×10^{-1}	3.0075×10^{-6}	1.7778×10^{-2}	1.4562×10^{-1}	2.5940×10^{-1}
	Std	1.4503×10^{-2}	1.9804×10^{-6}	5.1131×10^{-3}	1.4532×10^{-32}	3.8971×10^{-2}
	Rank	4	**1**	2	3	5
f_4	Mean	1.3043×10^{-10}	6.1442×10^{-3}	1.4789×10^{-3}	3.3235×10^{-3}	3.4971×10^{-2}
	Std	1.5945×10^{-10}	2.2358×10^{-2}	2.9159×10^{-4}	2.5874×10^{-3}	2.1107×10^{-3}
	Rank	**1**	4	2	3	5
f_5	Mean	1.7471×10^{-3}	6.6937×10^{-2}	8.9786×10^{-2}	4.1529×10^{-2}	3.0978×10^{-1}
	Std	5.3145×10^{-4}	4.9569×10^{-4}	7.2506×10^{-3}	7.0509×10^{-3}	2.9555×10^{-2}
	Rank	**1**	3	4	2	5
f_6	Mean	3.23811×10^{-5}	3.2986×10^{-5}	1.6616	4.3748×10^{-2}	3.0685
	Std	3.1251×10^{-2}	3.2325×10^{-2}	1.3884×10^{-1}	2.3449×10^{-2}	1.3645×10^{-1}
	Rank	**1**	2	4	3	5
f_7	Mean	3.6592×10^{-6}	3.0293×10^{-6}	1.0526×10^{-2}	1.6812×10^{-3}	2.0852×10^{-2}
	Std	2.8258×10^{-6}	2.6086×10^{-6}	7.2454×10^{-5}	7.7436×10^{-4}	8.6817×10^{-6}
	Rank	**1**	3	4	2	5
f_8	Mean	4.4537×10^{-10}	8.3839×10^{-8}	8.5416×10^{-5}	2.1706×10^{-7}	5.3409×10^{-4}
	Std	1.9980×10^{-10}	3.4183×10^{-7}	1.6413×10^{-5}	4.3498×10^{-7}	1.9193×10^{-4}
	Rank	**1**	2	4	3	5

Table 4 Results obtained by all algorithms on CEC05 benchmarks

Function		MABC	ABC	PSO	CMA-ES	EGA
f_9	Mean	-4.5832×10^{2}	-4.4132×10^{2}	3.4613×10^{1}	-4.3127×10^{2}	-3.5686×10^{2}
	Std	1.9472×10^{-14}	2.227×10^{-14}	5.8420×10^{2}	5.1713×10^{-14}	2.0814×10^{1}
	Rank	**1**	2	5	3	4
f_{10}	Mean	-4.5631×10^{1}	-4.3815×10^{1}	9.5247×10^{2}	-4.6012×10^{2}	1.4385×10^{4}
	Std	2.1225×10^{2}	2.3419×10^{2}	2.9622×10^{3}	2.2718×10^{-14}	5.4253×10^{3}
	Rank	2	3	4	**1**	5
f_{11}	Mean	4.2331×10^{2}	4.3429×10^{2}	2.6559×10^{7}	4.4678×10^{2}	4.0998×10^{4}
	Std	2.1056×10^{0}	1.9871×10^{0}	2.6716×10^{7}	1.8864×10^{0}	3.5523×10^{4}
	Rank	**1**	2	5	3	4
f_{12}	Mean	-1.9178×10^{2}	2.4681×10^{3}	6.4552×10^{3}	2.4392×10^{3}	3.2963×10^{3}
	Std	7.2376×10^{-3}	1.4429×10^{-2}	2.2961×10^{2}	1.3490×10^{-3}	4.6746×10^{2}
	Rank	**1**	3	5	2	4
f_{13}	Mean	-1.1831×10^{2}	-1.7658×10^{2}	-1.7397×10^{2}	-1.1896×10^{2}	-1.6903×10^{2}
	Std	1.3423×10^{-1}	6.4345×10^{23}	1.6532×10^{-1}	3.7856×10^{-2}	2.1242×10^{-2}
	Rank	2	3	4	**1**	5
f_{14}	Mean	-3.3021×10^{2}	-3.3021×10^{2}	-2.5985×10^{2}	-2.5849×10^{2}	-2.4729×10^{2}
	Std	1.2349×10^{-14}	3.3386×10^{-14}	3.3534×10^{1}	2.3651×10^{1}	3.5474×10^{0}
	Rank	**1**	**1**	4	3	5
f_{15}	Mean	-2.8243×10^{2}	-1.2346×10^{2}	-1.8931×10^{2}	-1.6559×10^{2}	-1.3717×10^{2}
	Std	1.1367×10^{1}	3.4544×10^{1}	2.5676×10^{1}	2.6985×10^{1}	3.9898×10^{1}
	Rank	**1**	5	2	3	4

benchmarks than on basic benchmarks. This means that MABC with the proposed effective strategies is more competent in tackling complex problems.

5 Multilevel threshold for image segmentation by MABC

5.1 Kapur criterion

The Kapur multi-threshold entropy measure [31] has been popularly employed in determining whether the optimal threshold method can provide image segmentation with satisfactory results. It is aimed to find the optimal thresholds that can yield the maximum entropy. For multilevel threshold, Kaptur's entropy may be described as follows.

Consider an image containing N pixels of gray levels from 0 to L. $H(i)$ represents the number of the ith gray level pixel and $P(i)$ represents the probability of i. Then, we obtain:

$$H_i=\sum_i \frac{P(i)}{\omega_k}\ln\frac{P(i)}{\omega_k} \tag{5}$$

Assuming that there are M–1 thresholds $\{t_1, t_2, \cdots, t_{M-1}\}$ that divide the original image into M classes (C1 for $[0, t_1]$, C2 for $[t_1, t_2]$, and CM for $[t_{M-1}$, L]), the optimal thresholds $\{t_1, t_2, \cdots, t_{M-1}\}$ selected by the Kapurmethod are depicted as follows:

$$\{t_1^*, t_2^*, \cdots, t_{M-1}^*\}=\arg\max\left\{\sum_{i=1}^{M-1} H_i\right\} \tag{6}$$

Equation (6) is used as the objective function for the proposed MABC based procedure which is to be optimized (minimized). A close look into this equation will show that it is very similar to the expression for uniformity measure.

5.2 Experiment setup

The datasets involve a set of popular tested images used in previous studies [32], including avion.ppm, house.ppm, lena.ppm, peppers.ppm, safari04.ppm and hunter.pgm. The size of each involved image is 512×512. The proposed algorithm and compared algorithms are evaluated based on Kapur. The parameters of these algorithms including MABC, ABC, PSO, EGA and CMA_ES are set as described in Section 4.1. We strived to utilize the proposed algorithm to obtain multiple thresholds with larger fitness values and fast computation ability. The numbers of thresholds M-1 investigated in the experiments are 2, 3, 4, 5, 7, and 9. The population size is set to 20 and the maximum FE is set to 2000. All the experiments are repeated 30 times.

5.3 Experimental results of multilevel threshold

Table 5 gives the fitness, mean computation time, and optimal thresholds with M–1=2, 3, and 4 obtained by Kapur. From Table 5, we can see that Kapur takes too long computation time on these cases. From computation results in Table 6, it can be observed that population- based methods consume similar CPU time, which exhibits superior performance to pure Kapur. As can be seen form Table 7, the proposed MABC algorithm generally performs satisfactory fitness values with M–1=2, 3 and 4, and consumes less time than Kapur. This is mainly due to the fact that the comprehensive learning strategy using improved PSO-based search equation enables the proposed algorithm obtain faster convergence speed. Furthermore, the MABC-based algorithm achieves the best achievements among the population-based methods in most cases. Moreover, the differences between the MABC and the other algorithms are more evident as the segmentation level increases.

To further investigate the population-based methods over high-dimensional segmentation, we conduct these algorithms on image segment with M–1=5, 7 and 9. Table 8 gives the average fitness and standard deviation obtained by each population-based algorithm. From Table 8, it can be observed that MABC demonstrates the best performance and stability on these high-dimensional functions, which is more efficient than the conventional ABC and other classical population-based algorithms, which proves that the MABC-based algorithm is more suitable for resolving multilevel image segmentation problems.

6 Conclusions

In order to apply artificial bee colony algorithm to solve complex optimization problems efficiently, this paper proposes a modified artificial bee colony algorithm, namely MABC. The potential of the proposed MABC to balance the exploration and exploitation tradeoff is achieved by combining local search and comprehensive learning strategies. In MABC, each individual can be characterized by focused and deeper exploitation of the promising regions and wider exploration of other regions of the search space. The algorithm achieves this by employing local search to encourage fine exploitation when it enters the

Table 5 Objective values and thresholds by Kapur method

Image	M–1=2		M–1=3		M–1=4	
	Objective value	Optimal threshold	Objective value	Optimal threshold	Objective value	Optimal threshold
Avion	12.3225	70,171	15.6050	68,126,181	18.4232	66,105,142,183
House	12.5950	89,149	15.6664	70,123,171	18.5158	58,96,143,182
Lena	12.4577	98,163	15.4294	81,124,173	18.1234	63,96,134,173
Peppers	12.7457	73,142	15.7998	60,110,168	18.6505	57,102,144,193
Safari04	11.9905	75,141	15.0652	62,115,163	17.7978	50,85,122,160
Hunter	12.4885	91,177	15.7247	57,115,177	18.6384	43,91,132,178
Mean CPU time	1.24333	—	51.342	—	2325.472	—

Table 6 Mean CPU time of compared population-based methods on Kapur algorithm

Dimension	MABC	ABC	PSO	CMA-ES	EGA
2	0.12038	0.48146	0.5631	0.45652	1.55012
3	0.043328	0.64661	0.8669	0.47340	2.51913
4	0.046637	0.57266	1.2099	0.49143	3.43565
5	0.060344	0.73647	1.5804	0.48354	4.81492
7	0.046105	0.56812	2.6771	0.48564	7.04398
9	0.041402	0.59253	3.1064	0. 48209	8.98092

Table 7 Objective value and standard deviation by compared population-based methods on Kapur algorithm

Image	M–1	Objective value (standard deviation)				
		MABC	ABC	PSO	CMA-ES	EGA
Avion	2	1.4788×10	1.4788×10	1.4788×10	1.4763×10	1.4787×10
		0	0	0	2.7853×10^{-2}	4.1077×10^{-4}
		71	71	71	76.9428	70.7000
		173	173	173	166.8398	171.7000
	3	1.8791×10	1.8790×10	1.8790×10	1.8366×10	1.8775×10
		4.1095×10^{-4}	7.0527×10^{-4}	1.5756×10^{-3}	8.3349×10^{-1}	9.9384×10^{-3}
		69.0000	69.0000	68.8000	65.9486	68.6000
		126.7000	126.8000	126.9000	127.1134	127.7000
		183.0000	182.9000	183.0000	182.6826	183.7000
	4	2.2212×10	2.2191×10	2.2212×10	2.21010×10	2.2166×10
		5.3515×10^{-4}	1.9772×10^{-2}	6.5323×10^{-4}	5.4463×10^{-2}	1.8436×10^{-2}
		66.7000	68.0000	66.7000	64.8520	64.0000
		106.1000	98.4000	105.6000	103.3790	103.6000
		145.2000	149.0000	144.8000	139.7959	144.4000
		185.0000	181.4000	185.0000	179.9956	185.1000
House	2	1.5118×10	1.5118×10	1.5118×10	1.5104×10	1.5116×10
		1.7778×10^{-4}	2.2674×10^{-15}	2.2674×10^{-15}	1.8044×10^{-2}	1.1003×10^{-3}
		88.0000	88	88	85.8326	87.3000
		147.9000	148	148	152.1452	147.6000
	3	1.8853×10	1.8853×10	1.8853×10	1.8832×10	1.8848×10
		0	4.2836×10^{-4}	0	1.5877×10^{-2}	4.2420×10^{-3}
		72	72.6000	72	73.0233	73.3000
		122	122.8000	122	124.4326	123.4000
		174	174.6000	174	177.2193	175.3000
	4	2.2324×10	2.2309×10	2.2324×10	2.2193×10	2.2281×10
		8.6056×10^{-5}	9.2188×10^{-3}	5.6337×10^{-5}	1.7201×10^{-1}	1.8558×10^{-2}
		59.0000	60.5000	59.0000	62.2997	59.6000
		99.0000	99.9000	99.0000	100.9495	101.9000
		139.7000	140.7000	139.9000	137.2325	141.6000
		183.7000	183.1000	183.9000	181.2818	184.0000
Lena	2	1.4951×10	1.4951×10	1.4951×10	1.4943×10	1.4951×10
		2.2693×10^{-15}	2.2693×10^{-15}	2.2693×10^{-15}	1.2241×10^{-2}	6.9023×10^{-4}
		97	97	97	97.8584	97.7000
		164	164	164	167.7098	164.3000
	3	1.8565×10	1.8565×10	1.8565×10	1.8548×10	1.8558×10
		2.2694×10^{-15}	0	5.7544×10^{-4}	3.9026×10^{-2}	4.8480×10^{-3}
		82	82	82.1000	86.2773	83.0000
		126	126	126.2000	132.6633	128.0000
		175	175	175.3000	179.4721	175.1000
	4	2.1848×10	2.1823×10	2.1839×10	2.1747×10	2.1811×10
		5.6279×10^{-4}	1.7588×10^{-2}	9.8253×10^{-3}	7.2537×10^{-2}	1.5890×10^{-2}
		64.0000	72.1000	68.2000	78.3727	66.6000
		96.9000	109.0000	104.0000	110.8736	100.7000
		137.0000	140.8000	141.2000	147.5074	138.7000
		179.0000	177.8000	180.3000	182.6328	179.3000

to be continued

continued

Image	M–1	Objective value (standard deviation)				
		MABC	ABC	PSO	CMA-ES	EGA
Peppers	2	1.4163×10 2.0990×10^{-15} 75 147	1.4163×10 2.0989×10^{-15} 75 147	1.4163×10 2.0989×10^{-15} 75 147	1.4161×10 2.3541×10^{-3} 74.8766 149.3480	1.4163×10 4.5835×10^{-4} 75.0000 147.2000
	3	1.9015×10 2.2694×10^{-15} 61 113 165	1.9014×10 9.5451×10^{-4} 61.6000 113.5000 165.2000	1.9015×10 1.3614×10^{-4} 60.9000 112.8000 164.8000	1.8965×10 5.1873×10^{-2} 65.5961 117.3846 167.3244	1.9010×10 3.6360×10^{-3} 61.9000 112.7000 163.3000
	4	2.2483×10 1.6533×10^{-2} 56.7000 102.5000 145.7000 191.7000	2.2465×10 2.1955×10^{-2} 60.6000 107.2000 142.8000 188.8000	2.2488×10 0 58 105 148 194	2.2315×10 1.9638×10^{-1} 45.8509 89.9624 137.9354 177.9564	2.2444×10 2.3047×10^{-2} 56.5000 102.4000 144.5000 190.7000
Safari04	2	1.4387×10 2.2675×10^{-15} 76 142	1.4387×10 5.4502×10^{-5} 76.0000 141.9000	1.4387×10 2.2674×10^{-15} 76 142	1.4384×10 4.7229×10^{-3} 73.9177 136.6354	1.4386×10 3.3076×10^{-4} 75.8000 141.2000
	3	1.8124×10 5.2475×10^{-5} 62.9000 112.0000 161.0000	1.8124×10 3.8681×10^{-4} 62.7000 111.7000 161.0000	1.8124×10 9.7458×10^{-5} 62.5000 111.9000 161.0000	1.8069×10 5.4540×10^{-2} 60.7060 102.1697 149.8310	1.8111×10 8.2416×10^{-3} 63.5000 111.4000 160.3000
	4	2.1453×10 1.3760×10^{-3} 51.6000 87.4000 123.8000 161.0000	2.1421×10 3.4449×10^{-2} 51.6000 92.3000 123.3000 157.8000	2.1453×10 1.1111×10^{-3} 51.5000 87.3000 123.7000 161.0000	2.0807×10 7.7656×10^{-1} 51.8837 84.8792 120.7795 166.8326	2.1407×10 3.1659×10^{-2} 49.2000 84.0000 121.5000 160.3000
Hunter	2	1.4989×10 0 92 179	1.4989×10 0 92 179	1.4989×10 1.1776×10^{-5} 91.9000 179.0000	1.4889×10 9.5063×10^{-2} 85.8257 165.4642	1.4988×10 1.0474×10^{-3} 91.5000 178.7000
	3	1.8923×10 4.6933×10^{-4} 58.7000 116.9000 179.0000	1.8922×10 2.5452×10^{-3} 60.0000 119.2000 179.0000	1.8923×10 2.2693×10^{-15} 59 117 179	1.8664×10 2.7682×10^{-1} 56.7791 119.5081 180.1028	1.8905×10 1.1392×10^{-2} 59.7000 117.2000 179.1000
	4	2.2472×10 2.9198×10^{-3}	2.2442×10 2.4866×10^{-2} 41.3000 85.4000 131.8000 179.5000	2.2449×10 6.6593×10^{-2} 47.3000 92.8000 137.5000 183.1000	2.2250×10 1.3027×10^{-1} 53.3592 104.8610 150.0102 192.8985	2.2414×10 2.9961×10^{-2} 42.3000 90.8000 133.2000 180.0000

Table 8 Objective value and standard deviation by compared population-based methods on Kapur algorithm

Image	M–1	Objective value (standard deviation)				
		MABC	ABC	PSO	CMA-ES	EGA
Avion	5	2.5320×10^{1}	2.5221×10^{1}	2.5320×10^{1}	2.4698×10^{1}	2.5206×10^{1}
		3.0374×10^{-4}	6.2487×10^{-2}	5.6043×10^{-4}	6.2887×10^{-1}	3.4271×10^{-2}
		60.0000	61.3000	59.9000	64.5832	57.9000
		89.3000	87.1000	89.1000	102.3949	92.8000
		123.5000	121.1000	123.2000	137.5997	125.0000
		155.6000	161.7000	155.3000	164.8053	155.9000
		187.3000	186.1000	187.2000	198.8899	188.2000
	7	3.1061×10	3.0740×10	3.1057×10	3.0458×10	3.0737×10
		6.7274×10^{-3}	1.1938×10^{-1}	9.4536×10^{-3}	3.5232×10^{-1}	1.2871×10^{-1}
		43.6000	47.6000	44.0000	39.3285	45.5000
		66.9000	63.8000	67.2000	63.7742	70.9000
		90.1000	95.5000	91.3000	89.9932	96.6000
		116.3000	114.7000	118.1000	116.6315	118.9000
		141.7000	147.4000	144.4000	142.9119	144.3000
		167.6000	173.8000	169.9000	169.8573	166.8000
		192.3000	198.8000	193.5000	200.1482	190.6000
	9	3.6244×10	3.5535×10	3.6257×10	3.4846×10	3.5698×10
		2.4223×10^{-2}	2.1094×10^{-1}	1.8195×10^{-2}	9.0016×10^{-1}	1.8886×10^{-1}
		42.2000	44.4000	41.8000	40.3983	42.1000
		64.4000	57.1000	63.8000	64.6187	63.9000
		85.5000	75.4000	85.6000	91.3059	84.3000
		106.2000	98.1000	105.5000	108.8207	100.6000
		127.3000	117.8000	125.9000	125.8186	123.5000
		147.9000	134.9000	146.1000	148.9948	142.4000
		167.7000	153.2000	166.2000	169.2357	161.4000
		186.5000	183.3000	186.0000	189.5198	181.5000
		204.9000	203.8000	204.6000	207.7511	202.8000
House	5	2.5443×10	2.5396×10	2.5440×10	2.4888×10	2.5350×10
		1.3814×10^{-3}	2.1435×10^{-2}	8.7192×10^{-3}	7.4379×10^{-1}	2.6642×10^{-2}
		55.4000	57.5000	55.0000	56.5953	56.0000
		89.0000	87.7000	88.3000	89.8251	90.4000
		122.1000	117.7000	121.3000	131.6330	121.1000
		154.7000	165.3000	154.4000	160.5719	154.9000
		189.4000	177.7000	189.4000	195.8798	189.7000
	7	3.1207×10	3.0864×10	3.1200×10	3.0399×10	3.0914×10
		1.1586×10^{-2}	1.2958×10^{-1}	1.5622×10^{-2}	6.6363×10^{-1}	6.9148×10^{-2}
		47.8000	43.0000	44.7000	34.5695	43.7000
		75.7000	66.0000	71.8000	60.0280	68.2000
		103.7000	88.7000	99.1000	86.2298	96.5000
		131.5000	121.6000	126.3000	114.9345	123.6000
		158.8000	155.9000	153.5000	145.4023	149.6000
		184.5000	158.8000	181.0000	175.8786	176.1000
		206.4000	200.9000	204.8000	205.5610	202.4000
	9	3.6579×10	3.5752×10	3.6587×10	3.4492×10	3.5792×10
		1.3848×10^{-2}	1.5598×10^{-1}	9.4458×10^{-3}	1.4665	1.6954×10^{-1}
		36.9000	39.6000	37.0000	32.5149	35.8000
		57.8000	62.0000	57.6000	58.6964	61.1000
		79.2000	88.9000	78.8000	82.0648	81.2000
		100.2000	113.9000	100.3000	103.7049	10.2000
		121.8000	132.1000	121.7000	130.1625	121.9000
		143.9000	148.1000	143.6000	151.2134	141.4000
		166.2000	177.8000	165.9000	170.4859	165.3000
		188.4000	191.5000	188.1000	187.9515	186.3000
		208.1000	210.9000	208.0000	213	205.6000

to be continued

continued

Image	M–1	Objective value (standard deviation)				
		MABC	ABC	PSO	CMA-ES	EGA
Lena	5	2.4958×10	2.4891×10	2.4958×10	2.4445×10	2.4825×10
		1.0276×10^{-4}	3.9236×10^{-2}	4.9381×10^{-4}	6.2124×10^{-1}	5.9339×10^{-2}
		63.2000	62.7000	62.9000	62.8437	62.6000
		94.2000	92.0000	94.0000	104.2379	96.0000
		128.0000	124.2000	127.7000	138.3126	127.8000
		163.0000	159.9000	162.8000	166.6784	159.8000
		194.0000	191.6000	194.0000	200.1172	191.8000
	7	3.0525×10	3.0192×10	3.0523×10	3.0014×10	3.0269×10
		1.4092×10^{-2}	1.3635×10^{-1}	1.5635×10^{-2}	1.7392×10^{-1}	1.0835×10^{-1}
		61.1000	68.1000	60.5000	55.8296	61.0000
		88.0000	82.0000	86.1000	76.1711	84.8000
		113.2000	122.2000	110.7000	100.9903	106.0000
		137.7000	134.9000	134.8000	124.6230	127.6000
		161.7000	159.8000	159.3000	155.3930	150.3000
		184.4000	183.2000	181.3000	185.0724	178.9000
		207.6000	199.3000	204.4000	208.8168	204.1000
	9	3.5610×10	3.5010×10	3.5593×10	3.4753×10	3.5098×10
		6.5984×10^{-2}	1.1778×10^{-1}	6.1620×10^{-2}	5.4634×10^{-1}	1.1050×10^{-1}
		54.6000	48.3000	53.8000	54.6064	52.9000
		74.5000	72.4000	73.4000	77.0530	73.0000
		94.7000	93.3000	93.0000	10.5115	92.9000
		115.8000	111.5000	113.9000	122.6892	112.8000
		136.7000	136.3000	134.2000	146.5324	134.1000
		157.7000	156.6000	154.8000	166.3013	152.5000
		177.2000	179.0000	174.9000	185.4754	172.5000
		197.0000	201.4000	195.3000	203.6046	193.8000
		216.7000	212.9000	216.1000	218.7176	213.5000
Peppers	5	2.5772×10	2.5688×10	2.5771×10	2.5437×10	2.5681×10
		4.5351×10^{-15}	6.4304×10^{-2}	3.3908×10^{-3}	3.1946×10^{-1}	3.6330×10^{-2}
		42	40.1000	41.8000	40.5265	42.1000
		77	76.1000	76.7000	71.0672	75.3000
		114	115.4000	113.3000	10.6263	111.8000
		154	152.7000	153.6000	136.5532	151.5000
		195	193.1000	195.0000	177.9534	189.9000
	7	3.1805×10	3.1367×10	3.1798×10	3.1249×10	3.1486×10
		1.8739×10^{-3}	8.5931×10^{-2}	2.7512^{-2}	5.1110×10^{-1}	1.1696×10^{-1}
		28.9000	33.6000	28.9000	32.2749	32.8000
		53.1000	61.9000	53.2000	63.7187	59.1000
		78.9000	92.3000	79.3000	88.3016	87.8000
		106.6000	119.3000	107.6000	116.1936	113.5000
		136.6000	145.6000	137.9000	140.5088	139.2000
		166.0000	175.2000	166.7000	168.7356	167.6000
		196.0000	198.8000	196.2000	193.8358	196.3000
	9	3.7063×10	3.6283×10	3.7062×10	3.5389×10	3.6501×10
		8.7697×10^{-3}	2.1931×10^{-1}	8.9762×10^{-3}	6.5672×10^{-1}	1.1960×10^{-1}
		27.5000	27.9000	27.4000	28.9289	23.9000
		49.4000	50.6000	49.1000	49.4507	48.2000
		71.3000	71.9000	70.6000	73.2390	69.8000
		91.7000	88.0000	91.7000	93.4893	90.6000
		112.5000	117.1000	113.2000	117.1072	111.2000
		134.3000	139.2000	135.0000	143.7611	135.5000
		155.9000	158.1000	156.1000	170.0757	155.3000
		176.9000	179.0000	177.2000	189.4210	178.3000
		198.6000	193.4000	199.0000	208.8491	200.5000

to be continued

continued

Image	M–1	Objective value (standard deviation)				
		MABC	ABC	PSO	CMA-ES	EGA
Safari04	5	2.4425×10	2.4278×10	2.4421×10	2.3760×10	2.4264×10
		8.8404×10^{-4}	4.9409×10^{-2}	1.2716×10^{-2}	5.4677×10^{-1}	4.6987×10^{-2}
		44.4000	42.7000	43.5000	46.2046	49.7000
		73.5000	67.9000	72.1000	78.7147	77.0000
		103.1000	100.7000	102.2000	112.6111	106.3000
		132.2000	129.2000	131.6000	143.8856	133.0000
		161.8000	155.7000	161.9000	165.4592	161.6000
	7	2.9744×10	2.9323×10	2.9775×10	2.8735×10	2.9431×10
		3.5518×10^{-2}	1.6447×10^{-1}	4.2420×10^{-3}	8.4573×10^{-1}	9.8657×10^{-2}
			37.2000	35.6000	44.2517	36.8000
			54.8000	55.8000	69.5757	58.1000
			82.1000	76.5000	86.8598	77.1000
			106.6000	97.7000	111.3488	97.6000
			120.4000	119.0000	132.8446	119.1000
			151.7000	140.2000	155.3272	139.5000
			169.2000	162.0000	178.8638	161.0000
	9	3.4530×10	3.3853×10	3.4540×10	3.3090×10	3.3969×10
		2.2465×10^{-2}	2.5449×10^{-1}	2.8748×10^{-2}	8.5686×10^{-1}	1.6485×10^{-1}
		33.1000	31.9000	31.8000	26.5241	28.6000
		51.4000	49.2000	49.5000	45.7286	46.3000
		69.6000	64.3000	67.6000	62.2029	63.7000
		87.3000	80.3000	85.9000	80.8269	82.6000
		105.9000	97.5000	104.3000	10.0816	100.1000
		124.0000	121.9000	122.8000	118.5414	119.6000
		142.3000	140.0000	142.0000	140.4589	138.0000
		161.0000	156.2000	161.0000	157.1801	155.7000
		175.9000	172.7000	176.0000	176.4394	172.2000
Hunter	5	2.5753×10	2.5648×10	2.5752×10	2.5336×10	2.5620×10
		4.7560×10^{-4}	4.1053×10^{-2}	4.7229×10^{-3}	3.0699×10^{-1}	3.2818×10^{-2}
		45.6000	45.8000	45.5000	49.9880	38.0000
		90.1000	82.3000	89.8000	88.7777	83.4000
		133.3000	127.5000	132.6000	128.8345	125.0000
		179.0000	169.4000	178.9000	171.0519	166.8000
		221.8000	211.9000	222.0000	206.6578	208.1000
	7	3.2041×10	3.1816×10	3.2028×10	3.1281×10	3.1717×10
		1.5440×10^{-2}	6.7630×10^{-2}	2.3392×10^{-2}	6.1339×10^{-1}	7.8053×10^{-2}
		34.0000	28.9000	32.2000	32.9020	30.1000
		68.8000	57.5000	65.5000	65.1084	64.3000
		103.4000	93.1000	99.1000	95.6527	95.2000
		138.4000	122.9000	133.2000	127.4247	130.0000
		174.1000	154.5000	166.8000	154.6755	160.0000
		198.5000	184.5000	193.3000	188.5151	190.9000
		226.6000	223.7000	225.0000	220.7071	224.0000
	9	3.7800×10	3.7186×10	3.7813×10	3.6777×10	3.7191×10
		2.8796×10^{-2}	1.6618×10^{-1}	1.0068×10^{-2}	6.3294×10^{-1}	1.4968×10^{-1}
		23.2000	23.6000	22.6000	28.3305	24.4000
		48.3000	47.2000	47.9000	51.6743	48.7000
		73.6000	73.6000	73.6000	74.6889	73.4000
		98.9000	100.9000	99.2000	100.4492	99.6000
		124.1000	130.6000	124.2000	124.5811	126.3000
		149.5000	158.6000	149.8000	155.0974	144.4000
		175.6000	183.2000	175.3000	179.1257	171.9000
		199.6000	205.6000	199.6000	201.2723	198.9000
		226.5000	227.4000	227.0000	228.0872	224.0000

promising region with high fitness, while enhance information sharing between excellent bees to improve the exploration when the individual finds difficulties during exploitation.

Finally, the MABC algorithm is applied in the real-world image segmentation problems. The correlative results obtained by MABC-based method on each image indicate a significant improvement compared to several other popular population-based methods. As an effective population-based method, the MABC algorithm can be incorporated to other popular threshold segmentation methods based on optimizing the fitness function.

References

[1] KITTLER J, ILLINGWORTH J. Minimum error threshold [J]. Pattern Recognition, 1986, 19: 41–47.

[2] PUN T. Entropic thresholding, a new approach [J]. Computer Graphics & Image Processing, 1981, 16(3): 210–239.

[3] OTSU N. A threshold selection method from gray-level histograms [J]. IEEE Transactions on Systems Man & Cybernetics, 2007, 9(1): 62–66.

[4] KAPUR J N, SAHOO P K, WONG A K C. A new method for gray-level picture thresholding using the entropy of the histogram [J]. Computer Vision Graphics & Image Processing, 1985, 29(3): 273–285.

[5] LIM Y W, SANG U L. On the color image segmentation algorithm based on the thresholding and the fuzzy c-means techniques [J]. Pattern Recognition, 1990, 23(9): 935–952.

[6] TSAI D M. A fast thresholding selection procedure for multimodal and unimodal histograms [J]. Pattern Recognition Letters, 1995, 16(6): 653–666.

[7] YIN P Y, CHEN L H. A new method for multilevel thresholding using symmetry and duality of the histogram [J]. Journal of Electronic Imaging, 1993, 2(4): 337–345.

[8] BRINK A D. Minimum spatial entropy threshold selection [J]. IEE Proceedings-Vision, Image and Signal Processing, 1995, 142(3): 128–132.

[9] CHENG H D, CHEN J R, LI J. Threshold selection based on fuzzy c-partition entropy approach [J]. Pattern Recognition, 1998, 31(7): 857–870.

[10] HUANG L K, WANG M J J. Image thresholding by minimizing the measures of fuzziness [J]. Pattern Recognition, 1995, 28(1): 41–51.

[11] CHANDER A, CHATTERJEE A, SIARRY P. A new social and momentum component adaptive PSO algorithm for image segmentation [J]. Expert Systems with Applications, 2011, 38(5): 4998–5004.

[12] MA L, HU K, ZHU Y, et al. A hybrid artificial bee colony optimizer by combining with life-cycle, Powell's search and crossover [J]. Applied Mathematics & Computation, 2015, 252: 133–154.

[13] GAO H, XU W, SUN J, et al. Multilevel thresholding for image segmentation through an improved quantum-behaved particle swarm algorithm [J]. IEEE Transactions on Instrumentation & Measurement, 2010, 59(4): 934–946.

[14] GHAMISI P, COUCEIRO M S, MARTINS F M L, et al. Multilevel image segmentation based on fractional-order Darwinian particle swarm optimization [J]. IEEE Transactions on Geoscience & Remote Sensing, 2014, 52(5): 2382–2394.

[15] CUEVAS E, ZALDIVAR D, PÉREZ-CISNEROS M. A novel multi-threshold segmentation approach based on differential evolution optimization [J]. Expert Systems with Applications, 2010, 37(7): 5265–5271.

[16] GAO H, KWONG S, YANG J, et al. Particle swarm optimization based on intermediate disturbance strategy algorithm and its application in multi-threshold image segmentation [J]. Information Sciences, 2013, 250(11): 82–112.

[17] KARABOGA D. An idea based on honey bee swarm for numerical optimization [R]. Technical report-tr06. Erciyes University, Computer Engineering Department, 2005.

[18] KARABOGA D, BASTURK B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm [J]. Journal of Global Optimization, 2007, 39(3): 459–471.

[19] MA L, HU K, ZHU Y, et al. A hybrid artificial bee colony optimizer by combining with life-cycle, Powell's search and crossover [J]. Applied Mathematics & Computation, 2015, 252: 133–154.

[20] MA L, HU K, ZHU Y, et al. Cooperative artificial bee colony algorithm for multi-objective RFID network planning [J]. Journal of Network & Computer Applications, 2014, 42: 143–162.

[21] MA L, ZHU Y, ZHANG D, et al. A hybrid approach to artificial bee colony algorithm [J]. Neural Computing & Applications, 2016, 27(2): 387–409.

[22] POWELL M J D. Restart procedures for the conjugate gradient method [J]. Mathematical Programming, 1977, 12(1): 241–254.

[23] SUMATHI S, HAMSAPRIYA T, SUREKHA P. Evolutionary intelligence: An introduction to theory and applications with Matlab [M]. Springer Science & Business Media, 2008.

[24] WOLPERT D H, MACREADY W G. No free lunch theorems for optimization [J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 67–82.

[25] MA L, HU K, ZHU Y, et al. Discrete and continuous optimization based on hierarchical artificial bee colony optimizer [J]. Journal of Applied Mathematics, 2014, 2014(1): 1–20.

[26] MA L, ZHU Y, LIU Y, et al. A novel bionic algorithm inspired by plant root foraging behaviors [J]. Applied Soft Computing, 2015, 37(C): 95–113.

[27] LIANG J J, QIN A K, SUGANTHAN P N, et al. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions [J]. IEEE Transactions on Evolutionary Computation, 2006, 10(3): 281–295.

[28] CLERC M, KENNEDY J. The particle swarm - explosion, stability, and convergence in a multidimensional complex space [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(1): 58–73.

[29] HANSEN N, OSTERMEIER A. Completely derandomized

self-adaptation in evolution strategies [J]. Evolutionary Computation, 2001, 9(2): 159–195.
[30] KAPUR J N, SAHOO P K, WONG A K C. A new method for gray- level picture thresholding using the entropy of the histogram [J]. Computer Vision Graphics & Image Processing, 1985, 29(3): 273–285.
[31] YIN P. Multilevel minimum cross entropy threshold selection based on particle swarm optimization [J]. Applied Mathematics & Computation, 2007, 184(2): 503–513.
[32] CAO L, BAO P, SHI Z. The strongest schema learning GA and its application to multilevel thresholding [J]. Image & Vision Computing, 2008, 26(5): 716–724.

(Edited by YANG Hua)

中文导读

增强性人工蜂群算法及在多阀值图像分割中的应用

摘要：提出了一种改进的人工蜂群算法来处理图像分割问题，具体采用一系列群体优化觅食策略来平衡开发和探测寻优模式。该算法的主要思想是将局部搜索策略和基于多维粒子群方程的复杂学习策略相结合，可丰富人工蜂群觅食行为模式。通过全局学习，蜂群把全局最优信息整合到搜索方程中以提高探测搜索能力，同时局部搜索使蜂群能更深层探索优势区域，最终取得开发和探索平衡。通过比较该改进蜂群算法和进化算法、群智能算法在一系列基准函数上的实验结果，表明本文所提出的算法的有效性。将改进蜂群算法应用于处理图像分割问题，实验结果也证明了该算法的有效性

关键词：人工蜂群算法；局部搜索；群体智能；图像分割