ORIGINAL PAPER

A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm

Dervis Karaboga · Bahriye Basturk

Abstract Swarm intelligence is a research branch that models the population of interacting agents or swarms that are able to self-organize. An ant colony, a flock of birds or an immune system is a typical example of a swarm system. Bees' swarming around their hive is another example of swarm intelligence. Artificial Bee Colony (ABC) Algorithm is an optimization algorithm based on the intelligent behaviour of honey bee swarm. In this work, ABC algorithm is used for optimizing multivariable functions and the results produced by ABC, Genetic Algorithm (GA), Particle Swarm Algorithm (PSO) and Particle Swarm Inspired Evolutionary Algorithm (PS-EA) have been compared. The results showed that ABC outperforms the other algorithms.

Keywords Swarm intelligence · Artificial bee colony · Particle swarm optimization · Genetic algorithm · Particle swarm inspired evolutionary algorithm · Numerical function optimization

1 Introduction

Several modern heuristic algorithms have been developed for solving combinatorial and numeric optimization problems [1]. These algorithms can be classified into different groups depending on the criteria being considered, such as population based, iterative based, stochastic, deterministic, etc. While an algorithm working with a set of solutions and trying to improve them is called population based, the one using multiple iterations to approach the solution sought is named as iterative algorithm. If an algorithm employs a probabilistic rule for improving a solution then it is called probabilistic or stochastic. Another classification can be made depending on the nature

D. Karaboga · B. Basturk (✉)
Department of Computer Engineering, Erciyes University, Kayseri, Turkey
e-mail: bahriye@erciyes.edu.tr

D. Karaboga
e-mail: karaboga@erciyes.edu.tr

of phenomenon simulated by the algorithm. This type of classification mainly has two important groups of population based algorithms: evolutionary algorithms (EA) and swarm intelligence based algorithms. The most popular EA is Genetic Algorithm (GA). GA attempts to simulate the phenomenon of natural evolution. In natural evolution, each species searches for beneficial adaptations in an ever-changing environment. As species evolve, the new attributes are encoded in the chromosomes of individual members. This information does change by random mutation, but the real driving force behind the evolutionary development is the combination and exchange of chromosomal material during breeding. Although sporadic attempts have been made to incorporate these principles in optimization routines since the early 1960s, GAs were first established on a sound theoretical basis by Holland [2]. The term swarm is used in a general manner to refer to any restrained collection of interacting agents or individuals. The classical example of a swarm is bees swarming around their hive but the metaphor can easily be extended to other systems with a similar architecture. An ant colony can be thought of as a swarm whose individual agents are ants; a flock of birds is a swarm of birds. An immune system [3] is a swarm of cells and molecules while a crowd is a swarm of people. Particle swarm optimization (PSO) algorithm which simulates the social behaviour of bird flocking or fish schooling was introduced by Eberhart and Kennedy in 1995 [4]. PSO is a population based stochastic optimization technique and well adapted to the optimization of nonlinear functions in multidimensional space. PSO has received significant interest from researchers studying in different research areas and has been applied to several real-world problems [5]. An improved version of PSO algorithm is the Particle Swarm Inspired Evolutionary Algorithm (PS-EA) which is a hybrid model of EA and PSO [21]. It compensates the limitations of PSO when solving real-world problems. In order to avoid infeasible individuals resulted from faulty updates, PS-EA incorporates PSO with heuristics of EA in the population generator and mutation operator while retaining the workings of PSO.

Several approaches have been proposed to model the specific intelligent behaviours of honey bee swarms and applied for solving combinatorial type problems [6–14]. Tereshko and Loengarov considered a bee colony as a dynamical system gathering information from an environment and adjusting its behaviour in accordance to it. They established a robotic idea on the foraging behaviour of bees. Usually, all these robots are physically and functionally identical, so that any robot can be randomly replaced by the others. The swarm possesses a significant tolerance; the failure in a single agent does not stop the performance of the whole system. The individual robots, like insects, have limited capabilities and limited knowledge of the environment. On the other hand, the swarm develops collective intelligence. The experiments showed that insect-like robots are successful in real robotic tasks [6]. They also developed a minimal model of forage selection that leads to the emergence of collective intelligence which consists of three essential components: food sources, employed foragers and unemployed foragers. This model defines two leading modes of the behaviour: recruitment to a nectar source and abandonment of a source [7,8]. Teodorović suggested to use the bee swarm intelligence in the development of artificial systems aimed at solving complex problems in traffic and transportation [9,10]. Teodorović also proposed the Bee Colony Optimization Metaheuristic (BCO) which is capable of solving deterministic combinatorial problems, as well as combinatorial problems characterized by uncertainty [11]. Drias et al. introduced a new intelligent approach or meta-heuristic called Bees Swarm Optimization (BSO), which is inspired from the behaviour of

real bees and they adapted it to the features of the maximum weighted satisfiability (max-sat) problem [12]. Similarly, Benatchba et al. introduced a meta-heuristic to solve a 3-sat problem based on the process of bees' reproduction [13]. Wedde et al. presented a novel routing algorithm called BeeHive, which has been inspired by the communicative and evaluative methods and procedures of honey bees. In BeeHive algorithm, bee agents travel through network regions called foraging zones. On their way their information on the network state is delivered for updating the local routing tables [14].

The works given in previous paragraph include the combinatorial type of problems. There is only one numerical optimization algorithm in the literature based on intelligent behaviour of honey bee swarm [15]. In Ref. [15], Yang developed a virtual bee algorithm (VBA) to solve the numeric function optimizations. For the functions with two-parameters, a swarm of virtual bees is generated and the swarm is started to move randomly in the phase space. These bees interact when they find some target nectar corresponding to the encoded values of the function. The solution for the optimization problem can be obtained from the intensity of bee interactions [15]. For optimizing multivariable functions, Karaboga has described an artificial bee colony (ABC) algorithm [16] which is different from the virtual bee algorithm. Basturk and Karaboga compared the performance of ABC algorithm with the performance of GA in Ref. [17].

In this paper, we compared the performance of the ABC algorithm with that of GA, PSO and PS-EA on a set of multi-dimensional numerical optimization problems. These algorithms are chosen because they also are swarm intelligence and population based algorithms as the ABC algorithm. In Sect. 2, the proposed ABC algorithm is described, in Sect. 3 experiments and results are presented.

2 Proposed artificial bee colony (ABC) algorithm

In the ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. A bee waiting on the dance area for making decision to choose a food source, is called an onlooker and a bee going to the food source visited by itself previously is named an employed bee. A bee carrying out random search is called a scout. In the ABC algorithm, first half of the colony consists of employed artificial bees and the second half constitutes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source is exhausted by the employed and onlooker bees becomes a scout. The main steps of the algorithm are given below:

- Initialize.
- REPEAT.
  - (a) Place the employed bees on the food sources in the memory;
  - (b) Place the onlooker bees on the food sources in the memory;
  - (c) Send the scouts to the search area for discovering new food sources.
- UNTIL (requirements are met).

In the ABC algorithm, each cycle of the search consists of three steps: sending the employed bees onto the food sources and then measuring their nectar amounts; selecting of the food sources by the onlookers after sharing the information of employed

bees and determining the nectar amount of the foods; determining the scout bees and then sending them onto possible food sources. At the initialization stage, a set of food source positions are randomly selected by the bees and their nectar amounts are determined. Then, these bees come into the hive and share the nectar information of the sources with the bees waiting on the dance area within the hive. At the second stage, after sharing the information, every employed bee goes to the food source area visited by herself at the previous cycle since that food source exists in her memory, and then chooses a new food source by means of visual information in the neighbourhood of the present one. At the third stage, an onlooker prefers a food source area depending on the nectar information distributed by the employed bees on the dance area. As the nectar amount of a food source increases, the probability with which that food source is chosen by an onlooker increases, too. Hence, the dance of employed bees carrying higher nectar recruits the onlookers for the food source areas with higher nectar amount. After arriving at the selected area, she chooses a new food source in the neighbourhood of the one in the memory depending on visual information. Visual information is based on the comparison of food source positions. When the nectar of a food source is abandoned by the bees, a new food source is randomly determined by a scout bee and replaced with the abandoned one. In our model, at each cycle at most one scout goes outside for searching a new food source and the number of employed and onlooker bees were equal.

In the ABC algorithm, the position of a food source represents a possible solution of the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population. At the first step, the ABC generates a randomly distributed initial population $P(G = 0)$ of SN solutions (food source positions), where SN denotes the size of population. Each solution (food source) x_i ($i = 1, 2, \ldots, SN$) is a D-dimensional vector. Here, D is the number of optimization parameters. After initialization, the population of the positions (solutions) is subjected to repeated cycles, $C = 1, 2, \ldots, C_{\max}$, of the search processes of the employed bees, the onlooker bees and scout bees. An artificial employed or onlooker bee probabilistically produces a modification on the position (solution) in her memory for finding a new food source and tests the nectar amount (fitness value) of the new source (new solution). In case of real bees, the production of new food sources is based on a comparison process of food sources in a region depending on the information gathered, visually, by the bee. In our model, the production of a new food source position is also based on a comparison process of food source positions. However, in the model, the artificial bees do not use any information in comparison. They randomly select a food source position and produce a modification on the one existing in their memory as described in (2.2). Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. After all employed bees complete the search process, they share the nectar information of the food sources (solutions) and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. As in the case of the employed bee, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one.

An onlooker bee chooses a food source depending on the probability value associated with that food source, p_i, calculated by the following expression (2.1):

$$p_i = \frac{\text{fit}_i}{\sum\limits_{n=1}^{\text{SN}} \text{fit}_n},\tag{2.1}$$

where fit_i is the fitness value of the solution i evaluated by its employed bee, which is proportional to the nectar amount of the food source in the position i and SN is the number of food sources which is equal to the number of employed bees (BN). In this way, the employed bees exchange their information with the onlookers.

In order to produce a candidate food position from the old one, the ABC uses the following expression (2.2):

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}),\tag{2.2}$$

where $k \in \{1, 2, \ldots, \text{BN}\}$ and $j \in \{1, 2, \ldots, D\}$ are randomly chosen indexes. Although k is determined randomly, it has to be different from i. $\phi_{i,j}$ is a random number between $[-1, 1]$. It controls the production of a neighbour food source position around $x_{i,j}$ and the modification represents the comparison of the neighbour food positions visually by the bee. Equation 2.2 shows that as the difference between the parameters of the $x_{i,j}$ and $x_{k,j}$ decreases, the perturbation on the position $x_{i,j}$ decreases, too. Thus, as the search approaches to the optimum solution in the search space, the step length is adaptively reduced.

If a parameter produced by this operation exceeds its predetermined limit, the parameter can be set to an acceptable value. In this work, the value of the parameter exceeding its limit is set to its limit value.

The food source whose nectar is abandoned by the bees is replaced with a new food source by the scouts. In the ABC algorithm this is simulated by randomly producing a position and replacing it with the abandoned one. In the ABC algorithm, if a position cannot be improved further through a predetermined number of cycles called *limit* then that food source is assumed to be abandoned.

After each candidate source position $v_{i,j}$ is produced and then evaluated by the artificial bee, its performance is compared with that of $x_{i,j}$. If the new food has equal or better nectar than the old source, it is replaced with the old one in the memory. Otherwise, the old one is retained. In other words, a greedy selection mechanism is employed as the selection operation between the old and the current food sources.

ABC algorithm in fact employs four different selection processes: (1) a global selection process used by the artificial onlooker bees for discovering promising regions as described by (2.1), (2) a local selection process carried out in a region by the artificial employed bees and the onlookers depending on local information (in case of real bees, this information includes the colour, shape and fragrance of the flowers) (bees will not be able to identify the type of nectar source until they arrive at the right location and discriminate among sources *growing* there based on their scent) for determining a neighbour food source around the source in the memory as defined in (2.2), (3) a local selection process called greedy selection process carried out by all bees in that if the nectar amount of the candidate source is better than that of the present one, the bee forgets the present one and memorizes the candidate source. Otherwise, the bee keeps the present one in the memory. (4) a random selection process carried out by scouts.

It is clear from the above explanation that there are three control parameters used in the basic ABC: The number of the food sources which is equal to the number of employed or onlooker bees (SN), the value of *limit* and the maximum cycle number (MCN).

In the case of honey bees, the recruitment rate represents a *measure* of how quickly the bee colony finds and exploits a newly discovered food source. Artificial recruiting could similarly represent the *measurement* of the speed with which the feasible solutions or the *good quality* solutions of the difficult optimization problems can be discovered. The survival and progress of the bee colony are dependent upon the rapid discovery and efficient utilization of the best food resources. Similarly, the successful solution of difficult engineering problems is connected to the relatively fast discovery of *good solutions* especially for the problems that need to be solved in real time. In a robust search process, exploration and exploitation processes must be carried out together. In the ABC algorithm, while onlookers and employed bees carry out the exploitation process in the search space, the scouts control the exploration process.

3 Experiments

3.1 Benchmark functions

A function is multimodal if it has two or more local optima. A function of variables is separable if it can be rewritten as a sum of functions of just one variable [18]. The separability is closely related to the concept of epistasis or interrelation among the variables of the function. The problem is even more difficult if the function is also multimodal. The search process must be able to avoid the regions around local minima in order to approximate, as far as possible, to the global optimum. The most complex case appears when the local optima are randomly distributed in the search space. The dimensionality of the search space is another important factor in the complexity of the problem [19]. A study of the dimensionality problem and its features was carried out by Friedman [20]. In order to compare the performance of the proposed ABC with PSO, PS-EA and GA, we used five classical benchmark functions as given in Ref. [21].

The first function is Griewank function whose value is 0 at its global minimum (0,0,…,0) (3.1). Initialization range for the function is $[-600,600]$. Griewank function has a product term that introduces interdependence among the variables. The aim is to overcome the failure of the techniques that optimize each variable independently. The optima of Griewank function are regularly distributed. Since the number of local optima increases with the dimensionality, this function is strongly multimodal. The multimodality disappears for sufficiently high dimensionalities ($n > 30$) and the problem seems unimodal.

$$f_1(\vec{x}) = \frac{1}{4,000}\left(\sum_{i=1}^{D}(x_i^2)\right) - \left(\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})\right) + 1. \tag{3.1}$$

The second function is Rastrigin function whose value is 0 at its global minimum (0,0,…,0) (3.2). Initialization range for the function is $[-15,15]$. This function is based on Sphere function with the addition of cosine modulation to produce many local

minima. Thus, the function is multimodal. The locations of the minima are regularly distributed. The difficult part about finding optimal solutions to this function is that an optimization algorithm easily can be trapped in a local optimum on its way towards the global optimum.

$$f_2(\vec{x}) = \sum_{i=1}^{D} (x_i^2 - 10 \cos(2\pi x_i) + 10). \tag{3.2}$$

The third function is Rosenbrock function whose value is 0 at its global minimum (1,1,...,1) (3.3). Initialization range for the function is [−15,15]. The global optimum is inside a long, narrow, parabolic shaped flat valley. Since it is difficult to converge the global optimum, the variables are strongly dependent, and the gradients generally do not point towards the optimum, this problem is repeatedly used to test the performance of the optimization algorithms.

$$f_3(\vec{x}) = \sum_{i=1}^{D} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2. \tag{3.3}$$

The forth function is Ackley function whose value is 0 at its global minimum (0,0,...,0) (3.4). Initialization range for the function is [−32.768,32.768]. Ackley has an exponential term that covers its surface with numerous local minima. The complexity of this function is moderated. An algorithm that only uses the gradient steepest descent will be trapped in a local optima, but any search strategy that analyses a wider region will be able to cross the valley among the optima and achieve better results. In order to obtain good results for this function, the search strategy must combine the exploratory and exploitative components efficiently.

$$f_4(\vec{x}) = 20 + e - 20e^{\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)} - e^{\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)} \tag{3.4}$$

The fifth function is Schwefel function whose value is 0 at its global minimum (420.9867, 420.9867,..., 420.9867) (3.5). Initialization range for the function is [−500, 500]. The surface of Schwefel function is composed of a great number of peaks and valleys. The function has a second best minimum far from the global minimum where many search algorithms are trapped. Moreover, the global minimum is near the bounds of the domain.

$$f_5(\vec{x}) = D * 418.9829 + \sum_{i=1}^{D} -x_i \sin(\sqrt{|x_i|}). \tag{3.5}$$

3.2 Settings for algorithms

Common control parameters of the algorithms are population size and the number of maximum generation. In the experiments, maximum number of generations were 500, 750 and 1,000 for the dimensions 10, 20 and 30, respectively; and the population size was 125 as in Ref. [21]. Other control parameters of the algorithms and the schemes used in Ref. [21] are presented below; and also the values of these control parameters employed for GA, PSO and PS-EA in Ref. [21] are presented.

3.2.1 GA settings

The used GA scheme presented in Ref. [21] is as follows: single point uniform cross-over with the rate of 0.95, random selection mechanism, gaussian mutation with the rate of 0.1 and linear ranking fitness function are employed. A child chromosome is added to the population in child production scheme.

3.2.2 PSO settings

PSO equations are given with the expressions in (3.6) and (3.7).

$$\vec{v}(t+1) = \omega\vec{v}(t) + \phi_1 \text{rand}(0,1)(\vec{p}(t) - \vec{x}(t))$$
$$+ \phi_2 \text{rand}(0,1)(\vec{g}(t) - \vec{x}(t)), \tag{3.6}$$

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t+1), \tag{3.7}$$

where ω is the additional inertia weight, which varies from from 0.9 to 0.7 linearly with the iterations. The learning factors, ϕ_1 and ϕ_2 are set to be 2. The upper and lower bounds for v, (v_{min}, v_{max}) are set to be the maximum upper and lower bounds of x, i.e. $(v_{min}, v_{max}) = (x_{min}, x_{max})$. If the sum of accelerations would cause the velocity on that dimension $v(t+1)$, to exceed v_{min} or v_{max}, then the velocity on that dimension $v(t+1)$, is limited to v_{min} or v_{max}, respectively [21].

3.2.3 PS-EA settings

Particle Swarm Inspired Evolutionary Algorithm (PS-EA) is a hybridized algorithm combining concepts of PSO and EA. The main module of PS-EA is the Self-updating Mechanism (SUM), which makes use of the Inheritance Probability Tree (PIT) to do the updating operation of each individual in the population. A Dynamic Inheritance Probability Adjuster (DIPA) is incorporated in SUM to dynamically adjust the inheritance probabilities in PIT based on the convergence rate or status of the algorithm in a particular iteration [21]. The purpose of setting the initial inheritance probabilities differently is to observe whether DIPA works properly to adjust the inheritance probabilities back to that ensures an effective search. An infeasible set of initial inheritance probabilities are used to test the performance of DIPA module of PS-EA [21].

3.2.4 ABC settings

ABC algorithm has a few control parameters: Maximum number of cycles (MCN) equals to the maximum number of generation and the colony size equals to the population size, i.e. 125, as in the study presented in Ref. [21] . The percentage of onlooker bees was 50% of the colony, the employed bees were 50% of the colony and the number of scout bees was selected as one. The increase in the number of scouts encourages the exploration as the increase of onlookers on a food source increases the exploitation. Each of the experiments was repeated 30 times with different random seeds. The mean function values of the best solutions found by the algorithms for different dimensions have been recorded. The mean and the standard deviations of the function values obtained by the ABC, GA, PSO, PS-EA are given in Table 1.

Table 1 Results obtained by GA, PSO, PS-EA and ABC Algorithms

f	Alg. Dim	GA [21] Mean	SD	PSO [21] Mean	SD	PS-EA [21] Mean	SD	ABC1 Mean	SD	ABC2 Mean	SD
f_1	10	0.050228	0.029523	0.079393	0.033451	0.222366	0.0781	0.00087	0.002535	0.000329	0.0018
	20	1.0139	0.026966	0.030565	0.025419	0.59036	0.2030	2.01E-08	6.76E-08	0	0
	30	1.2342	0.11045	0.011151	0.014209	0.8211	0.1394	2.87E-09	8.45E-10	0	0
f_2	10	1.3928	0.76319	2.6559	1.3896	0.43404	0.2551	0	0	0	0
	20	6.0309	1.4537	12.059	3.3216	1.8135	0.2551	1.45E-08	5.06E-08	0	0
	30	10.4388	2.6386	32.476	6.9521	3.0527	0.9985	0.033874	0.181557	0	0
f_3	10	46.3184	33.8217	4.3713	2.3811	25.303	29.7964	0.034072	0.045553	0.012522	0.01263
	20	103.93	29.505	77.382	94.901	72.452	27.3441	0.13614	0.132013	0.014458	0.010933
	30	166.283	59.5102	402.54	633.65	98.407	35.5791	0.219626	0.152742	0.020121	0.021846
f_4	10	0.59267	0.22482	9.8499E-13	9.6202E-13	0.19209	0.1951	7.8E-11	1.16E-09	4.6E-11	5.4E-11
	20	0.92413	0.22599	1.1778E-6	1.5842E-6	0.32321	0.097353	1.6E-11	1.9E-11	0	1E-12
	30	1.0989	0.24956	1.4917E-6	1.8612E-6	0.3771	0.098762	3E-12	5E-12	0	0
f_5	10	1.9519	1.3044	161.87	144.16	0.32037	1.6185	1.27E-09	4E-12	1.27E-09	4E-12
	20	7.285	2.9971	543.07	360.22	1.4984	0.84612	19.83971	45.12342	0.000255	0
	30	13.5346	4.9534	990.77	581.14	3.272	1.6185	146.8568	82.3144	0.000382	1E-12

GA, PSO, PS-EA and ABC1 indicate the results that are obtained after 500, 750 and 1,000 cycles with a population having 125 individuals, while ABC2 indicates the results that are obtained after 1,000, 1,500 and 2,000 cycles
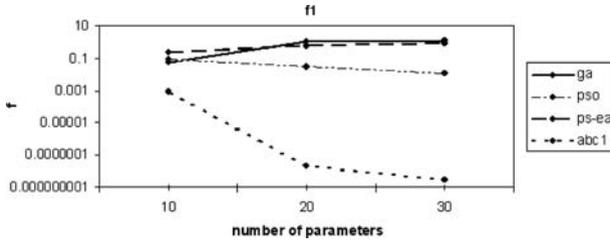
Fig. 1 The mean best values obtained for f_1 by GA, PSO, PS-EA and ABC1 after 500, 750 and 1,000 cycles for dimensions 10, 20 and 30
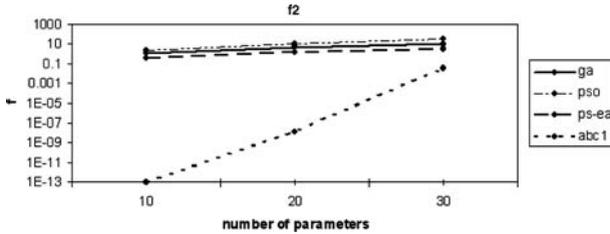

Fig. 2 The mean best values obtained for f_2 by GA, PSO, PS-EA and ABC1 after 500, 750 and 1,000 cycles for dimensions 10, 20 and 30
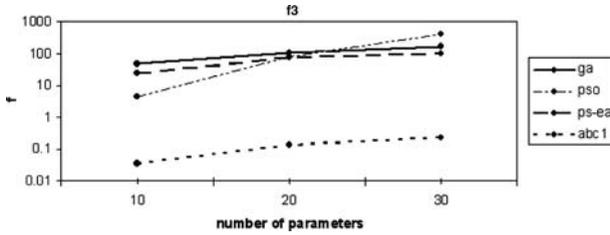

Fig. 3 The mean best values obtained for f_3 by GA, PSO, PS-EA and ABC1 after 500, 750 and 1,000 cycles for dimensions 10, 20 and 30

In order to show the performance of the ABC algorithm more clearly, the graphical representations of the results in Table 1 are reproduced in Figs. 1–5. Table 1 and the graphics indicate that while the performance of PS-EA algorithm deteriorates in optimizing difficult functions like Griewank and Ackley functions, the ABC algorithm shows better performance on them. However, PS-EA and GA outperform the ABC algorithm on only Schwefel function for dimension 20 and 30 by these settings. After obtaining results for 500, 750 and 1,000 cycles, the ABC algorithm was run for 1,000, 1,500 and 2,000 cycles instead of 500, 750 and 1,000 cycles for dimension 10, 20 and 30, respectively, to see whether the ABC algorithm can improve the solutions for this problem when the value of MCN is increased. The results produced in this case are shown in the ABC2 column of Table 1. As seen from the results in this column, the ABC algorithm can converge to the minimum of Schwefel function, too. In other words, this proves that the ABC algorithm has the ability of getting out of a local minimum in the search space and finding the global minimum. In the ABC algorithm, while the exploration process carried out by artificial scouts is good for global
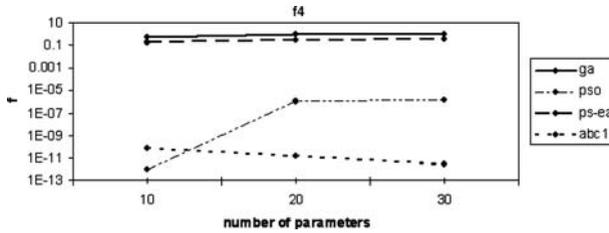
Fig. 4 The mean best values obtained for f_4 by GA, PSO, PS-EA and ABC1 after 500, 750 and 1,000 cycles for dimensions 10, 20 and 30
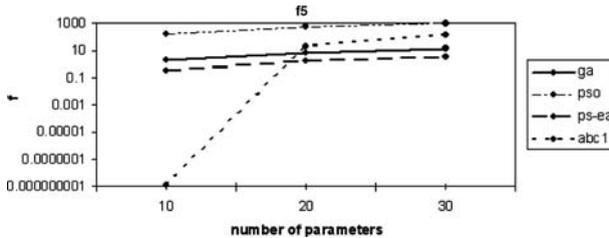

Fig. 5 The mean best values obtained for f_5 by GA, PSO, PS-EA and ABC1 after 500, 750 and 1,000 cycles for dimensions 10, 20 and 30
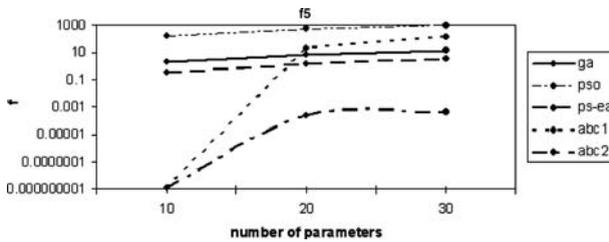

Fig. 6 The mean best values obtained for f_5 by GA, PSO, PS-EA, ABC1 after 500, 750 and 1,000 cycles and ABC2 after 1,000, 1,500 and 2,000 cycles

optimization, the exploitation process managed by artificial onlookers and employed bees is very efficient for local optimization. Therefore, the ABC algorithm is quite successful in optimizing multivariable and multimodal functions. In Fig. 6, the mean best values of GA, PSO, PS-EA, ABC1 and ABC2 are given graphically for f_5 when the value of MCN is set to 1,000, 1,500 and 2,000. A demo version of the Artificial Bee Colony Algorithm has been prepared and it is presented on its homepage [22].

4 Conclusion

This paper compared the performance of the ABC with that of GA, PSO and PS-EA which are also swarm intelligence and population based algorithms as the ABC algorithm. In order to demonstrate the performance of the ABC algorithm, PSO, PS-EA, GA and ABC algorithms were tested on five high dimensional numerical benchmark functions that have multimodality. From the simulation results it was concluded that

the proposed algorithm has the ability to get out of a local minimum and can be efficiently used for multivariable, multimodal function optimization. There are several issues which remain as the scopes for future studies such as the investigation of the control parameters' effect on the performance of the ABC algorithm and the convergence speed of the algorithm.

References

1. Pham, D.T., Karaboga, D.: Intelligent Optimisation Techniques. Springer, London (2000)
2. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI (1975)
3. De Castro, L.N., Von Zuben, F.J.: Artificial Immune Systems. Part I. Basic Theory And Applications. Technical Report No. Rt Dca 01/99, Feec/Unicamp, Brazil (1999)
4. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE Service Center, Piscat away (1995)
5. Fukuyama, Y., Takayama, S., Nakanishi, Y., Yoshida, H.: A particle swarm optimization for reactive power and voltage control in electric power systems. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1523–1528. Orlando, Florida, USA (1999)
6. Tereshko, V.: Reaction-diffusion model of a honeybee colony's foraging behaviour. In: Schoenauer M. (ed.) Parallel Problem Solving from Nature VI. Lecture Notes in Computer Science, vol. 1917, pp. 807–816, Springer, Berlin (2000)
7. Tereshko, V., Lee, T.: How information mapping patterns determine foraging behaviour of a honey bee colony. Open Syst. Inf. Dyn. **9**, 181–193 (2002)
8. Tereshko, V., Loengarov, A.: Collective Decision-Making in Honey Bee Foraging Dynamics. Comput. Inf. Sys. J., **9**(3), 1–7 (2005)
9. Teodorović, D.: Transport Modeling By Multi-Agent Systems: A Swarm Intellgence Approach, Transport. Plan. Technol. **26**(4), 289–312 (2003)
10. Lucic, P., Teodorović, D.: Transportation Modeling: An Artificial Life Approach. ICTAI, pp. 216–223. Washington D.C. (2002)
11. Teodorović, D., Dell'Orco, M.: Bee colony optimization—a cooperative learning approach to complex transportation problems. In: Proceedings of the 10th EWGT Meeting, Poznan, 13–16 September 2005
12. Drias, H., Sadeg, S., Yahi, S.: Cooperative bees swarm for solving the maximum weighted satisfiability problem, computational intelligence and bioinspired Systems. In: Proceedings of the 8th International Workshop on Artificial Neural Networks, IWANN 2005, Vilanova i la Geltr, Barcelona, Spain, 8–10 June 2005
13. Benatchba, K., Admane, L., Koudil, M.: Using bees to solve a data-mining problem expressed as a max-sat one, artificial intelligence and knowledge engineering applications: a bioinspired approach. In: Proceedings of the First International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005, Las Palmas, Canary Islands, Spain, 15–18 June 2005
14. Wedde, H.F., Farooq, M., Zhang, Y.: BeeHive: an efficient fault-tolerant routing algorithm inspired by honey bee behavior, ant colony, optimization and swarm intelligence. In: Proceedings of the 4th International Workshop, ANTS 2004, Brussels, Belgium, 5–8 September 2004
15. Yang, X.S.: Engineering optimizations via nature-inspired virtual bee algorithms. Lecture Notes in Computer Science, pp. 317–323. Springer, GmbH (2005)
16. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005
17. Basturk, B., Karaboga, D.: An artificial bee colony (ABC) algorithm for numeric function optimization. In: Proceedings of the IEEE Swarm Intelligence Symposium 2006, Indianapolis, Indiana, USA, 12–14 May 2006
18. Hadley, G.: Nonlinear and Dynamics Programming. Addison Wesley, Reading, MA (1964)
19. Boyer, D.O., Martnez, C.H., Pedrajas, N.G.: Crossover Operator for Evolutionary Algorithms Based on Population Features. http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume24/ortizboyer05a-html/Ortiz-Boyer.html.

20. Friedman, J.H.: An overview of predictive learning and function approximation. From Statistics to Neural Networks, Theory and Pattern Recognition Applications, NATO ASI Series F, vol. 136, pp. 1–61. Springer, Berlin (1994)
21. Srinivasan, D., Seow, T.H.: Evolutionary Computation, CEC '03, 8–12 Dec. 2003, 4(2003), Canberra, Australia, pp. 2292–2297.
22. http://mf.erciyes.edu.tr/abc