Future Generation Computer Systems 89 (2018) 478–493

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A smart artificial bee colony algorithm with distance-fitness-based neighbor search and its application

Laizhong Cui [a], Kai Zhang [a], Genghui Li [a,b,*], Xizhao Wang [a], Shu Yang [a], Zhong Ming [a], Joshua Zhexue Huang [a], Nan Lu [a]

[a] *College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, PR China*
[b] *Department of Computer Science, City University of Hong Kong, Hong Kong*

HIGHLIGHTS

- A search strategy is designed for employed bee by utilizing the near-good-neighbors to generate offspring.
- A new selection probability is proposed for onlooker bee by considering both fitness and distance factors, in which each bee searches around far-good position of the current best solution.
- A search mechanism is presented for onlooker bee by exploiting the best solution among the neighbors of the selected position.
- A new variant of ABC is formed by combining above three proposed components, which outperforms some state-of-the-art ABC variants.

ARTICLE INFO

Article history:
Received 13 May 2018
Received in revised form 4 June 2018
Accepted 27 June 2018
Available online 10 July 2018

Keywords:
Artificial bee colony algorithm
Distance-fitness-based neighbor search
Global numerical optimization
Real life optimization problem

ABSTRACT

Artificial bee colony (ABC) is a kind of biologically-inspired optimization technology, which has been successfully used in various scientific and engineering fields. To further improve the performance of ABC, some neighborhood structures defined by topology, distance or fitness information have been used to design the novel search strategies. However, the distance and fitness information have the potential benefits by building the better neighborhood structure to balance the exploration and exploitation ability. Therefore, this paper proposes a new ABC variant with distance-fitness-based neighbor search mechanism (called DFnABC). To be specific, the employed bee exploits the information of a near-good-neighbor that not only has good fitness value but also is close to its own position to focus on the local exploitation around itself. Moreover, the selectable exploration scope of the employed bee decreases gradually with the process of the evolution and the search direction is guided by a randomly selected leader from the top Q solutions. In addition, each onlooker bee firstly selects a food source position that not only has high quality but also is far away from the current best position to search for the purpose of paying more attention to global exploration among the search space. Furthermore, the best neighbor's information of the selected food source position is used to generate the candidate solution. Through the comparison of DFnABC and some other state-of-the-art ABC variants on 22 benchmark functions, 28 CEC2013 test functions and 5 real life optimization problems, the experimental results show that DFnABC is better than or at least comparable to the competitors on majority of test functions and real life problems.

1. Introduction

With the continuous development of society and technology, a variety of global optimization problems (GOPs) [1] and some complex real life optimization problems (ROPs) [2–5] have been arisen in diverse scientific and engineering fields. The traditional optimization methods are difficult or even impracticable to be used to solve these problems. With the purpose of dealing with these problems, some evolutionary algorithms (EAs) have been proposed, such as Particle Swarm Optimization (PSO) [6], Ant Colony Optimization (ACO) [7], Genetic Algorithm (GA) [8,9], Differential Evolution (DE) [10], Artificial Bee Colony (ABC) algorithm [11,12] and so on. Due to their attractive advantages, *i.e.*, simple structure, easy to implement, good robustness, the study on the EAs has been attracting more and more attention of researchers and it has been triumphantly used to solve various kinds of optimization problems.

In this paper, we focus on ABC algorithm, which is firstly proposed by Karaboga [11] through simulating the intelligent foraging

* Corresponding author at: Department of Computer Science, City University of Hong Kong, Hong Kong.
E-mail address: genghuili2-c@my.cityu.edu.hk (G.H. Li).

https://doi.org/10.1016/j.future.2018.06.054
0167-739X/

behavior of real bee colony. Although ABC has shown excellent global optimization ability, it also faces the challenge of slow convergence as other EAs. The main reason for this shortcoming explained in many research works is that its solution search equation does well in exploration, but poorly in exploitation [13,14]. Therefore, many improved ABC variants have been proposed by inventing new solution search equations. Specially, utilizing the neighborhood structure to design search mechanism is a promising way to enhance the performance of ABC. To be specific. Gao et al. (BABC) [13] propose the fitness-based neighborhood search mechanism to exploit the potential information of the good individuals, where the individuals used in the search equation are proportionally selected based on its fitness. The better the fitness is, the larger the selection probability is. Kiran and Babalik (iABC) [15] propose a neighbor selection mechanism for onlooker bee. In iABC, the individuals whose fitness are less than the average fitness of the colony are regarded as the neighbors by all onlooker bees. Karaboga and Gorkemli (qABC) [16] use the best individual of the neighbor to guide the search of onlooker bee, in which the neighborhood structure is defined by Euclidean distance. Besides, NILABC [17] also uses neighborhood structure based on Euclidean distance to improve the performance of ABC.

However, the different definitions of neighborhood structure are able to lead to the different information communication among the population so as to affect the search performance of ABC. Specifically, on the one hand, if the neighborhood structure only considers the fitness value, it is conducive to fast convergence through preferentially exploiting the information of the good individuals but it may be easy to cause premature convergence when solving multimodal functions. On the other hand, the distance-based neighborhood structure is beneficial to explore the landscape and is able to keep diversity but it may hamper to converge. Based on these considerations, in this paper, we propose two new neighborhood structures that are defined by distance and fitness information to achieve a balance between the exploration and exploitation ability. Moreover, the usage of distance information aims at keeping the exploration ability, while the utilization of fitness information aims at promoting to converge. To crystallize this idea, two novel search equations based on the different neighborhood structures are designed for the employed bee and onlooker bee, respectively. And a new selection mechanism is designed for the onlooker bee. Concretely, in employed bee phase, each employed bee exploits the information of a near-good-neighbor to guide its search. The near-good-neighbor denotes that a food source position is close to the position of the corresponding employed bee and possesses the good fitness value (*i.e.*, function value). By this way, employed bees are able to fully explore the local areas among the landscape. Moreover, the selectable explore scope of employed bees gradually decreases with the increment of the number of iterations, and the search direction is guided by a randomly selected leader from the top Q solutions. In onlooker bee phase, each onlooker bee firstly chooses a far-good food source position by the probability model, in which the selection probability is positively proportional to the fitness value and inversely proportional to the distance between the selected solution and the current best solutions. The far-good food source position means that a food source position is far away from the current best position and has good quality. Furthermore, a new search mechanism is employed by onlooker bees to generate the candidate solution, which exploits the information of the best neighbor of the selected food source. By this way, not only the current best area but also some other promising local areas are also explored by onlooker bees, and the exploration and exploitation can be effectively balanced. Overall, we propose two new search strategies respectively used by employed bee and onlooker bee, and put forward a selection mechanism for onlooker bee, which are embedded into the original ABC to form a new variant of ABC, named DFnABC. To evaluate the performance of DFnABC, some extensive experiments are conducted on 22 common benchmark functions [18], 28 CEC2013 test functions [19] and 5 real life problems [2–5]. The comparison results with other ABC variants, (*i.e.*, qABC [16], BABC [13], CABC [18], GABC [20], MABC [21]) validate the effectiveness and efficiency of DFnABC in terms of solution quality, robustness and convergence speed.

The major contributions of this paper are summarized as follows:

(1) A novel search strategy is proposed for employed bee, which exploits one of its near-good-neighbors to generate candidate solution. Moreover, the selectable explore scope of employed bee decreases with the evolutionary process and the search direction is guided by a randomly selected good individual from the top Q solutions.
(2) A new selection mechanism is designed for onlooker bee, which also considers both fitness value and distance. To be specific, each onlooker bee searches around a far-good food source position of the current best solution. By this way, the promising areas can be explored fully.
(3) A new search mechanism is presented for onlooker bee, in which the best solution among the neighbors of the selected position is used to generate the candidate solution.

The remainder of this paper is organized as follows. Section 2 introduces the original ABC algorithm briefly. A review on the improved ABC variants is given in Section 3. The details of DFnABC are described in Section 4. The experimental results and corresponding analysis are given in Section 5. Finally, Section 6 concludes this paper.

2. The original ABC

ABC is a population-based optimization technology by simulating the intelligent foraging behavior of honey bee swarms, which belongs to a branch of the EAs. In ABC, a food source position denotes a possible solution of the optimization problem, and the nectar amount of each food source position is regarded as the quality/fitness of the corresponding solution. The population consists of SN food source positions, which are evolved by three types of bees, namely the employed bee, onlooker bee and scout bee. The first half of the population is considered as employed bees, which are mainly responsible for randomly searching the neighboring region of the corresponding parent food source position and share the information of their food source positions with onlooker bees. The second half consists of onlooker bees, which mainly take responsibility of searching better food source position around the good solutions, which are selected based on the quality information provided by employed bees. The last bee group includes scout bees. If a food source position is not improved by employed bees or onlooker bees after a preset number of times (*limit*) around a certain food source position, which will be abandoned by its employed bee, and this employed bee will become a scout bee to search a new food source position randomly in the whole search space. In the original ABC, there are four phases, *i.e.*, initialization phase, employed bee phase, onlooker bee phase and scout bee phase. After the initialization phase, ABC iteratively carries out employed bee phase, onlooker bee phase and scout bee phase until the termination condition is met. The details of each phase are described as follows.

Initialization phase: The initial population contains SN food source positions (*i.e.*, solutions), each of which is generated according to Eq. (1),

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

where $i = 1, 2, \ldots, SN, j = 1, 2, \ldots, D$. SN is the number of employed bees or onlooker bees; D is the dimensionality of the search space; x_j^{max} and x_j^{min} represents the upper bound and lower bound of the jth dimension, respectively. Moreover, the fitness value of each food source position is calculated as Eq. (2),

$$fit\ (x_i) = \begin{cases} \dfrac{1}{1+f\ (x_i)}, & if\ (f\ (x_i) \geq 0) \\ 1+|f\ (x_i)|\ , & otherwise \end{cases} \tag{2}$$

where $fit\ (x_i)$ represents the fitness value of the ith food source position x_i and $f\ (x_i)$ is the objective function value of the food source position x_i for the optimization problem.

Employed bee phase: Each employed bee generates a candidate food source position in the neighborhood of its own food source position according to Eq. (3),

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot \left(x_{i,j} - x_{r,j}\right) \tag{3}$$

where v_i is the ith candidate food source position and x_i is the ith food source position. x_r is randomly selected from the population, which is different from x_i. $\varphi_{i,j}$ is a uniformly distributed random number in the range of $[-1, 1]$ and j is randomly selected from $\{1, 2, \ldots, D\}$. If the fitness value of v_i is better than its parent x_i, x_i will be replaced by v_i, and the *counter* that records the number of the consecutive unsuccessful updates of the food source position x_i is reset to 0. Otherwise, x_i is kept to enter the next generation and *counter* is increased by 1.

Onlooker bee phase: After updating all food sources, employed bees will share the quality information of their food source positions with onlooker bees. Each onlooker bee selects a food source position to search according to Eq. (4). Obviously, the better the fitness value is, the bigger the selection probability is. And then it will further search the better food source in the neighborhood of the selected food source position by using Eq. (3). If a candidate food source position obtained by the onlooker bee is better than its parent food source position (the selected one based on Eq. (4)), the parent food source position will be replaced by the new one, and its *counter* is reset to 0. Otherwise, the old one is kept and *counter* is increased by 1.

$$p\ (x_i) = \frac{fit\ (x_i)}{\sum_{j=1}^{SN} fit\ (x_i)} \tag{4}$$

Scout bee phase: After onlooker bee phase, the food source with the highest *counter* value is selected. If its *counter* value is bigger than the *limit* value, the food source position with the highest *counter* value will be abandoned by its employed bee, and then this employed bee will become a scout bee to seek a new food source position randomly according to Eq. (1). After the new food source position is generated, its *counter* value is reset to 0, and the scout bee returns to be an employed bee.

Note that if the jth variable $v_{i,j}$ of the ith candidate food source position violates the boundary constraints in employed bee phase and onlooker bee phase, it will be reset according to Eq. (1). The detailed procedure of ABC is shown in Fig. 1.

3. The improved ABC variants

With the development of science and technology, many GOPs arisen in the engineering and science field are becoming so complicated that the performance of the EAs needs to be enhanced. Being attracted by the potential and advantage of ABC, a lot of improved research works about ABC have been proposed by researchers. A brief review of these works is given as follows.

(1) Invention of new solution search equations. Motivated by PSO, Zhu and Kwong [20] proposed an improved ABC algorithm (short for GABC), which combines the valuable information of the current best solution into their solution search equation to enhance the exploitation ability of ABC. The experimental data demonstrated that GABC outperforms the original ABC on most cases. Inspired by the mutation operator DE/best/1, Gao and Liu [21] presented an improved search equation ABC/best/1, which is based on that the bee searches only around the best solution of the previous iteration to improve the exploitation. Besides, a probability P is introduced to control the frequency of employing solution search equation of ABC and ABC/best/1. Gao et al. [22] designed two new search equations to generate candidate solutions (EABC) for employed bee and onlooker bee respectively to balance the exploration and the exploitation. Luo et al. [23] put forward a modified solution search equation for the onlooker bee, which exploits the best solution of the previous iteration to guide the search of new candidate solutions (named COABC). Shi et al. [24] proposed a novel update equation and an improved dimension-selection strategy for employed bees to strike a good balance between global search and local tuning abilities (called NSABC). Kiran and Findik [25] invented a modified solution search equation (dABC) by introducing the previous successful direction information into the original search equation. Babaoglu [26] put forward a distribution-based solution update rule for ABC (distABC) to overcome stagnation behavior. Liu et al. [27] combined the information of the best solution found by each bee and the current best solution found by the population into their search equation (GPSABC). In addition, Kiran et al. [28], Wang et al. [29], Gao et al. [30] and Alkin and Erdal [31] proposed some methods that use multiple search equations adaptively to enhance the comprehensive performance of ABC based on the principle that different search equations have distinct advantages and perform differently on different problems or at different stages on the same problem.

(2) Combination with auxiliary techniques. The performance of ABC could be improved effectively by the assistance of auxiliary techniques. Kang et al. [32,33] presented two new ABC variants (called RABC and HABC), in which the original ABC is used to realize the exploration ability, and the rotational direction method and Hooke Jeeves pattern search technology are employed to focus on exploitation ability, respectively. Loubiere et al. [34] applied a sensitivity analysis method, *i.e.,* Morris' OAT method (One-At-Time), to select the dimensions with high influence on the objective result for preferential evolvement. Zhang et al. [35] presented two modified versions of ABC (namely GABC1 and GABC2) to enhance robustness and promote convergence speed, which are inspired by the concept of Grenade Explosion Method (GEM). Moreover, Gao et al. [18,36] employ the chaotic map and opposition-based learning method in the initial phase and the Powell's method as a local search tool to reinforce the performance of ABC. In addition, the memory-save technology has attracted much attention for improving the performance of ABC. For instance, Kiran and Babalik [15] used a memory board to save the solutions whose qualities are better than the average fitness value of the population, which is used to guide the population evolve. Banitalebi et al. [37] enhanced the exploitation capability of ABC though incorporating Estimation of distribution algorithm framework (EDA) with ABC (called EcABC). Li and Yang [38] introduced a new ABC variant named ABC with memory (ABCM), which memorizes the previous successful experiences of foraging behavior to guide the current foraging behavior. On the contrary, the short term tabu list (STTL) of tabu search was used by Bayraktar [39] to memorize the abandoned solution.

(3) Combination with other algorithms. Xiang et al. [40] proposed a well-known hybrid algorithm named hABCDE, which incorporates a modified ABC with a modified DE. Gao et al. [41] combined DE with gbest-guided ABC (GABC) by using an evaluation strategy with an attempt of utilizing more prior information of the previous search experience to speed up the convergence

Algorithm 1:The procedure of ABC

Initialization: Generate SN solutions that contain D variables according to Eq. (1)
While $FES < maxFES$
  for $i = 1$ to SN
    Generate a new solution v_i^g in the neighbourhood of x_i^g using Eq. (3)
    if $f(v_i^g) \leq f(x_i^g)$
      Replace x_i^g by v_i^g, $counter(i)=0$
    else
      $counter(i)= counter(i)+1$
    end if
  end for
  Calculate the probability P according to Eq. (4)
  for $i =1$ to SN
    Select a solution x_s^g from the population according to probability P
    Generate a new solution v_s^g in the neighbourhood of x_s^g using Eq. (3)
    if $f(v_s^g) \leq f(x_s^g)$
      Replace x_s^g by v_s^g, $counter(s)=0$
    else
      $counter(s)= counter(s)+1$
    end if
  end for
  $FES=FES+2SN$
  Select the solution $x_{\max}^g$ with max $counter$ value
  if $counter(\max)>limit$
    Replace $x_{\max}^g$ by a new solution generated according to Eq.(1)
    $FES=FES+1$, $counter(\max)=0$
  end if
end while

Output: The food source (solution) with the smallest objective value

Fig. 1. The pseudo-code of original ABC.

(called DGABC). Abraham et al. [42] invented a novel hybrid differential artificial bee colony algorithm (called HDABCA), which embeds DE strategy into standard ABC algorithm. Hsieh et al. [43] introduced PSO into ABC and proposed a new hybrid algorithm, named EABC-PGSVM. Tsai [44] integrated the artificial bee colony and bees algorithms into a hybrid ABC-BA algorithm, in which an agent can perform as an ABC agent in the ABC sub-swarm and/or a BA agent in the BA sub-swarm. Jadon et al. [45] put forward a hybridization of ABC and DE to develop a more efficient meta-heuristic algorithm (called HABCDE), which updates the bee's position through evolutionary operations of differential evolution (DE/best/1/bin) algorithmic process in onlooker bee phase. Ozturk et al. [46] embedded genetic operators such as crossover and swap into the neighborhood searching mechanism of ABC (GB-ABC). In addition, Li and Yin [47] proposed a hybrid algorithm (called DE/ABC) by combining differential evolution with artificial bee colony to solve parameter estimation for chaotic systems. Ding et al. [48] applied the local search mechanism of cuckoo search optimization (CS) to the onlooker bee phase of ABC in order to enhance its dedicated search. Besides, Chen et al. [49] integrated ABC with the annealing algorithm, Fister et al. [50] mixed ABC with the memetic search, and Tuba and Bacanin [51] combined ABC with the firefly algorithm, and so on.

4. The proposed algorithm

In this section, the proposed DFnABC algorithm is described in detail. Firstly, we give the motivations of our proposed algorithm. Secondly, the proposed algorithmic components are presented respectively. Finally, the complete proposed algorithm is shown for further explanation.

4.1. Motivations

Generally speaking, to find the best food source position, the employed bees take the responsibility of exploring the surrounding areas of the current position, while the onlooker bees pay attention to exploit promising regions. However, based on the search strategy of Eq. (3) in employed bee and onlooker bee phase, lots of employed bees explore the bad areas that results in the waste of resources and the onlooker bees search some local optimize areas that leads to the low precision results for solving the multimodal optimization problems. Thus, many improved ABC variants have been proposed by utilizing the different definitions of the neighborhood structure (*e.g.*, fitness value [13,15] and distance [16,17]) to improve the performance of ABC. Generally, these neighborhood structures are exploited in a simple way, which may not fully excavate the potential information hidden in the neighborhood, and even result in affecting the balance between exploration and exploitation ability of ABC. In order to further enhance the performance of optimization algorithm, the mixed information of distance and fitness has been exploited in different aspects of optimization [52,53] by different forms. For example, in context of global optimization, Biswas et al. [52] utilized the tournament selection method to select the parents by the modified affinity matrix (called mASDE), which takes into account the proximity of neighboring solutions and its relative gradient for generating a selection probability. In context of multimodal optimization, Biswas et al. [53] utilized the ratio of total reward T (as judged by the fitness value) with the distance E (Euclidean: that the forager needs to travel from its present position) to determine the onlooker bees select the food sources in perturbation strategy.

In this paper, we propose the effectively neighborhood structure by excavating the potential benefits both distance and fitness.

The usage of distance information aims at keeping the exploration, while the utilization of fitness information aims at promoting to converge. The differences of our work on distance-fitness trade-off are given as follows. First, compared with the single neighborhood structure [15–17], to making full use of the potential information about both distance and fitness, we build the effectively neighborhood search mechanism to better balance the exploration and exploitation ability. Second, compared with the combined neighborhood structure [52,53], the valuable information of fitness and distance are utilized not only in the parents selection probability but also in the search mechanism to adapt for the different responsibilities for different groups of bees. Third, the proposed selection probability model can not only improve the performance of ABC at a certain level, but also remarkably reduce the time-consuming process of roulette wheel method. The specific description of DFn-ABC is given as follows.

4.2. Search strategy for employed bee

In the original ABC, for generating a candidate food source position (new solutions), each employed bee randomly chooses a food source position from the population to combine with its current food source position. This way causes that ABC does well in exploration but shows slow convergence rate. In order to improve the convergence rate and enhance the exploitation ability of ABC, a new neighborhood structure that is defined by distance and fitness information is introduced into employed bee phase. Concretely, a novel search strategy for employed bee is given as follows,

$$v_{i,j}^g = \begin{cases} x_{k,j}^g + \varphi_{i,j} \cdot \delta, & if\ x_{k,j}^g \le x_{pbest,j}^g \\ x_{k,j}^g - \varphi_{i,j} \cdot \delta, & otherwise \end{cases} \tag{5}$$

where $x_{k,j}^g$ represents the near-good-neighbor for the ith employed bee at the gth iteration, and δdenotes the explore scope of the ith employed bee. The leader x_{pbest}^g is randomly selected from the top Q solutions of the current population and $\varphi_{i,j}$ is a random real number in [0, 1]. The near-good-neighbor x_k^g for the ith employed bee is selected from the population by the roulette method based on the value of $P_{k,i}^e$, which is calculated as follows,

$$P_{k,i}^e = w \cdot d_{k,i,j}^g + (1-w) \cdot f_{k,i}^g \tag{6}$$

$$d_{k,i,j}^g = \frac{\max_{1\le k\le SN}\left\{\Delta_{k,i,j}^g\right\} - \Delta_{k,i,j}^g}{\max_{1\le k\le SN}\left\{\Delta_{k,i,j}^g\right\} - \min_{1\le k\le SN}\left\{\Delta_{k,i,j}^g\right\}} \tag{7}$$

$$\Delta_{k,i,j}^g = \left|x_{i,j}^g - x_{k,j}^g\right| \tag{8}$$

$$f_{k,i}^g = \frac{\Delta f_{k,i}^g - \min_{1\le k\le SN}\left\{\Delta f_{k,i}^g\right\}}{\max_{1\le k\le SN}\left\{\Delta f_{k,i}^g\right\} - \min_{1\le k\le SN}\left\{\Delta f_{k,i}^g\right\}} \tag{9}$$

$$\Delta f_{k,i}^g = f\left(x_i^g\right) - f\left(x_k^g\right) \tag{10}$$

$$w = \frac{1}{2} - a \cdot \sin\left(\left(\frac{g}{g_{\max}} - \frac{1}{2}\right) \cdot \pi\right), \quad g \le g_{\max} \tag{11}$$

In Eqs. (6)–(10), according to the value of $P_{k,i}^e$, the ith employed bee chooses the kth food source x_k^g as its near-good neighbor by roulette-wheel method. $d_{k,i,j}^g$ represents the normalized distance between x_i^g and x_k^g at the jth dimension in gth generation. $f_{k,j}^g$ denotes the normalized fitness difference between x_k^g and x_i^g in the gth generation. Note that in employed bee phase, we directly take objective function values as fitness. a is real number in $[-0.5, 0.5]$. g and $g_{\max}$ is the current iteration number and the maximum iteration number, respectively. w is a weighting factor, which is used to control the influence of fitness value ($f_{k,j}^g$) and distance ($d_{k,i,j}^g$). Its value is reduced with the increment of the number of iterations as shown in Eq. (11). To be specific, at the early stage, in much as possible, the value of w should be slightly larger so that the distance information pays a more important role than fitness information. In contrast, at the later stage, the smaller w will be beneficial for fitness information that takes a more important role than distance information so as to make the good solutions fully exploited to accelerate converge. Intuitively, the change curve of w is shown in Fig. 2.

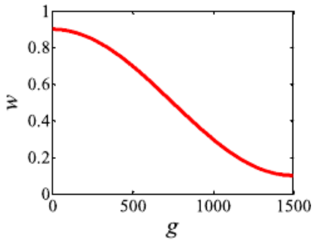


Fig. 2. The change curve of weighting factor w.

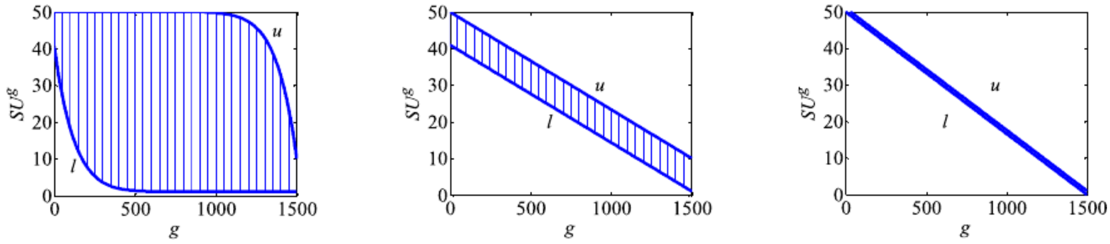
Moreover, in Eq. (5), δ is the explore scope of employed bee. In original ABC, the explore scope for the ith employed bee is the distance between x_i^g and the randomly selected position x_r^g at the selected dimension j (Eq. (3)). However, different evolutionary stages may need different explore scopes (δ) for improving the search efficiency. Generally, at the beginning of the evolutionary process, the large δ means a large perturbation, which is able to make the algorithm explore more new areas so as to keep the diversity of the population. As the population converges to some certain local regions, the smaller δ can provide the fine search ability. Thus, in this paper, the explore scope δ for the ith employed bee is randomly chosen from the subset SU^g of the scope set U^g. U^g is shown as follows.

$$U^g = \left\{U_{1,i}^g, U_{2,i}^g, \ldots, U_{SN,i}^g\right\},$$
$$U_{1,i}^g \le U_{2,i}^g \le \cdots \le U_{SN-1,i}^g \le U_{SN,i}^g \tag{12}$$
$$U_{r,i}^g = \left|x_{r,i}^g - r_{k,j}^g\right|, \quad r = 1, 2, \ldots, SN \tag{13}$$

where U^g is a set, which consists of the distances between the other positions and the near-good-neighbor x_k^g of the ith employed bee at the jth dimension. Moreover, the elements of set U^g are sorted in ascending order. Moreover, SU^g is a continuous subset of U^g, which is established as follows,

$$SU^g = \left\{U_{l,i}^g, U_{l+1,i}^g, \ldots, U_{u,i}^g\right\},$$
$$U_{1,i}^g \le U_{l,i}^g \le \cdots \le U_{u,i}^g \le U_{SN,i}^g \tag{14}$$
$$l = 1 + \left\lfloor SN \cdot 0.8\left(\left(g_{\max} - g\right)/g_{\max}\right)^{e^2}\right\rfloor \tag{15}$$
$$u = \left\lfloor SN \cdot \left(1 - 0.8\left(g/g_{\max}\right)^{e^2}\right)\right\rfloor \tag{16}$$

where l and u denote the initial index and end index of SU^g in U^g, respectively. Due to the selection randomness, the employed bees either choose the solution (x_k) which is close to their own current position and results in a small range search, or choose the solution (x_k) which is far away from current position and leads to a large range search. Actually, the employed bees should choose the appropriate scope of exploration at different stages. To select the proper explore scope for balancing exploration and exploitation, three kinds of mechanisms are proposed to control l and u, which are illustrated in Fig. 3. On the basis of our preliminary experiments, we choose the first one that is defined as Eqs. (15) and (16). In this way, at the early stages, SU^g includes the large distances (large index) and aims at providing large perturbation and exploring different areas. While at the later stages, SU^g contains the small order to make the current region searched by the employed bee as distances (small index) and aims at providing small perturbation



(a): Nonlinear decrease (b): Linear decrease($l < u$) (c): Linear decrease($l = u$)

Fig. 3. The change curve for different strategy of l and u ($SN = 50$).

and finely exploring the promising areas. In summary, to determine the explore scope δ of the ith employed bee, the procedures are: (1) Choose x_k^g as the near-good-neighbor of the ith employed bee. (2) Get the U^g set that includes the distances between the other positions and x_k^g. (3) Sort U^g in ascending order. (4) Use Eqs. (15) and (16) to control the lower index l and upper index u and select a continuous subset SU^g from U^g. (5) Randomly select an element subset SU^g as the explore scope of the ith employed bee.

In addition, the movement direction of the candidate solutions of x_k^g is guided by the leader x_{pbest}^g, which is randomly selected from the top Q positions of the current population. The parameter Q is set as follows,

$$Q = \max\left(\left\lfloor \frac{SN}{2}\left(1 - \frac{g}{g_{\max}}\right)\right\rfloor, 2\right) \tag{17}$$

where $\lfloor x \rfloor$ means the minimum integer that is larger than x. Intuitively, Q decreases gradually with the evolutionary process. At the early stages, Q is set to a large value (*e.g.*, $g = 0$, $Q = SN/2$), which means most of solutions can be regarded as the leader and a large number of search directions can be provided. While at the later stages, Q is set to a small value (*e.g.*, $g = g_{\max}$, $Q = 2$), which indicates only a small amount of solution with high quality can be treated as leader and only a few promising search directions can be chosen.

Overall, a new search mechanism is proposed for employed bee, which includes three core operators to deal with the following three core questions: (1) how to select the near-good-neighbor x_k^g; (2) how to select the explore scope δ; (3) how to determine the leader x_{pbest}^g. Moreover, the search behavior for the variable X_1 of the original search mechanism and our proposed search mechanism for employed bee are illustrated in Fig. 4(a) and (b), respectively, where the red point represents the global optimal position, and the blue box denotes the position of the ith employed bee. As shown in Fig. 4(a), in original ABC, the employed bee searches around itself, and the search direction and step (red arrow) are determined by the randomly selected food source position x_r. Fig. 4(b) indicates that the employed bee searches around its near-good-neighbor x_k, and its search direction and step are respectively determined by x_{pbest} and δ in our proposed method.

4.3. Selection mechanism for onlooker bee

The onlooker bees search around the good food source positions with high probability according to the feedback information provided by employed bees with the aims of finding the better candidate food source positions. The original selection mechanism (defined by Eq. (4)) only considers the fitness value, which may be conducive to the quick convergence through preferentially choosing promising areas, a new selection mechanism that takes x_{best}^g as the reference point is established as follows,

$$P\left(x_s^g\right) = \begin{cases} w \cdot \dfrac{\Delta_{s,best,j}^g}{\max_{1\le k\le SN}\left\{\Delta_{k,best,j}^g\right\}} \\ + (1-w)\cdot \dfrac{fit_s^g}{\max_{1\le k\le SN}\left\{fit_k^g\right\}}, & if \quad x_s \neq x_{best} \\ 1, & otherwise \end{cases} \tag{18}$$

$$\Delta_{s,best,j}^g = \left|x_{best,j}^g - x_{s,j}^g\right| \tag{19}$$

where $P\left(x_s^g\right)$ denotes an estimate value that the onlooker bee selects the sth food source position to search through wheel-roulette method; fit_s^g represents the fitness value of sth food source position according to Eq. (2); x_{best}^g is the best position of the population in the gth generation; w is the weighting factor, which is set the same as Eq. (11). As shown in Eq. (18), the positions that have better fitness and are far away from the current best position will be assigned a large selection probability. By this way, different promising areas are able to be fully explored by onlooker bee. But, compared with other areas, the one located by the current best solution is the most promising area. Thus, the selection probability of x_{best}^g is set to 1.

4.4. Search strategy for onlooker bee

After onlooker bee chooses a food source position, it will generate a candidate position for this selected position. The search strategy for onlooker bee is given as follows,

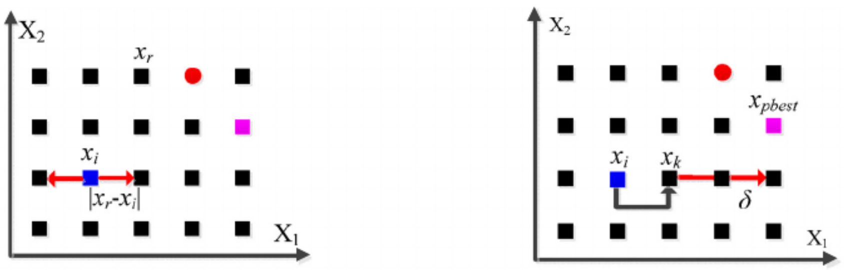
$$v_{s,j}^g = \begin{cases} x_{nbest,j}^g + \phi_{s,j}\left(x_{nbest,j}^g - x_{r1,j}^g\right), & if\ rand \le Sr \\ x_{nbest,j}^g + \phi_{s,j}\left(x_{nbest,j}^g - x_{r2,j}^g\right), & otherwise \end{cases} \tag{20}$$

where x_{nbest}^g represents the best food source position among the neighbors of the selected food source position x_s^g; $\phi_{s,j}$ is a random real number in $[-1, 1]$. x_{r1}^g is randomly selected from the neighbors of x_s^g and x_{r2}^g is randomly selected from the population. Sr is the mean success rate that the number of total successful searches ($Scount$) for all neighbors of x_s^g divided by their number of total searches ($Tcount$). Moreover, the neighborhood structure used in the onlooker bee phase is defined by the principle that if the distance ($\left|x_{s,j}^g - x_{k,j}^g\right|$) between x_s^g and $x_k^g (k = 1, 2, \ldots, SN)$ at the selected jth variable is less than the neighborhood radius R_s, x_k^g is a neighbor of x_s^g. The neighbor radius R_s is calculated as Eq. (21).

$$R_s = \frac{\sum_{k=1,k\neq s}^{SN}\left|x_{s,j}^g - x_{k,j}^g\right|}{SN-1} \tag{21}$$

As shown in Eq. (20), onlooker bees use the best neighbor to generate the candidate position. Moreover, the purpose of the ing the current good positions to search. But it is also easy to fall into the local optimal. Therefore, in order to explore different

usage of that if the better positions can be easily found in the neighborhood of x_s^s, (*i.e.*, Sr is large), the area located by the x_s^s is



(a): original mechanism (Eq. (3)) (b): Our proposed search mechanism (Eq. (5))

Fig. 4. The employed bee's search behavior. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a promising area, and the onlooker bee should conduct fine search in this area.

4.5. The complete proposed algorithm (DFnABC)

In this subsection, we put forward a novel ABC variant (named DFnABC) by combining the two novel search strategies and a new selection mechanism with the basic framework of ABC. The pseudo-code of the complete DFnABC is demonstrated in Fig. 5. As shown in Fig. 5, the essential differences between the ABC and DFnABC are the search strategy of employed bee (**lines 6–8**), the selection mechanism (**lines 17–18**) and search strategy (**lines 19–21**) of onlooker bee.

5. Experimental results

5.1. Benchmark test functions and experimental settings

In this paper, 22 well-known benchmark functions with low-dimension ($D = 30$), middle-dimension ($D = 50$) and high-dimension ($D = 100$) are firstly employed to evaluate the performance of DFnABC. These functions cover different types of problems and possess different characteristics, such as unimodal, multimodal, separable and non-separable (which are abbreviated as U, M, S and N in the category column of Table 1). Moreover, f_1-f_6 and f_8 is continuous function, f_7 is discontinuous step function, and f_9 is noisy-quartic function. Especially, f_{10} is the Rosenbrock function, which is unimodal for $D = 2$ and $D = 3$, but it may have many optimal solutions when $D>3$. f_{11}-f_{22} are multimodal, and the number of their local minima increases exponentially with the problem dimension. Generally speaking, the unimodal functions can be used to test the exploitation ability and the multimodal functions can be employed to demonstrate the exploration ability. The mathematical expression, category, search range, the global optimal value, and the "acceptable value" of each function are listed in columns 2, 3, 4, 5 and 6 of Table 1 respectively. When the objective function value of the best solution obtained by an algorithm in a run is less than the acceptable value, this run is regarded as a *successful run*.

To evaluate the performance of DFnABC, three evaluation metrics are used in our experiments. The detailed descriptions are given as follows.

(1) The mean and standard deviation of fitness value (mean/std): the mean fitness value implies the mean of the best objective function fitness values obtained by the algorithm when the stopping criterion is satisfied, and standard deviation signifies the stability of an algorithm. For the min-

(2) The average number of function evaluation (AVEN): AVEN denotes the mean number of function evaluations that are required to reach the acceptable value, which is adopted to evaluate the convergence speed. The smaller AVNE is, the faster the convergence speeds is. Note that AVEN will be only recorded for the successful runs. If an algorithm cannot find any solution whose objective function value is smaller than the acceptance value in all runs, AVEN will be denoted by "NAN".

(3) The successful rate (SR): the successful rate (SR %) of the 25 independent runs is utilized to evaluate the robustness or reliability of different algorithms. The greater the value of this metric is, the better the robustness/reliability is.

There are six parts in our experiments. ***Experiment* 1** validates the effectiveness of our proposed algorithmic components (*i.e.*, the two novel search strategies, a selection mechanism, the replaced base vector, the selectable exploration scope and the guided search direction) on 22 benchmark functions. ***Experiment* 2** evaluates the performance of DFnABC by the comparison with five outstanding ABC variants on 22 benchmark functions. ***Experiment* 3** demonstrates the performance of DFnABC by the comparison with five ABC variants on CEC2013 test functions. ***Experiment* 4** demonstrates the performance of DFnABC on five real life optimization problems. ***Experiment* 5** gives the comparisons of time complexity. ***Experiment* 6** proves the performance of DFnABC, based on the comparison of some other non-ABC methods and the state-of-the-art ABC variants on 22 benchmark functions.

5.2. Experiment 1: The effectiveness of the proposed algorithmic components

In order to clearly demonstrate that each of the proposed algorithmic components (*i.e.*, the two novel search strategies and a selection mechanism, the replaced base vector, the selectable exploration scope and the guided search direction) can make the contribution, the original ABC and the following seven ABC variants (DFnABC, DFnABCe, DFnABCo, DFnABCs, DFnABC1, DFnABC2 and DFnABC3) are used to discuss in this experiment on 22 benchmark functions with 30*D*.

(1) DFnABC denotes that ABC integrates all proposed algorithmic components.
(2) DFnABCe denotes that the search strategy (shown in Eq. (5)) of the employed bee of DFnABC is replaced by the original search equation (shown in Eq. (3)).
(3) DFnABCo denotes that the search strategy (shown in Eq. (20)) of the onlooker bee of DFnABC is replaced by the

imum optimization problem, the smaller the value of mean and standard deviation are, the higher quality/accuracy of the solution has.

original search equation (shown in Eq. (3)).

(4) DFnABCs denotes that the proposed selection mechanism (shown in Eq. (18)) of DFnABC is replaced by the original one (shown in Eq. (4)).

Table 1
Benchmark functions in experiments.

Name	Function	Category	Range	Min	Accept
Sphere	$f_1(x)=\sum_{i=1}^{D}x_i^2$	US	$[-100, 100]^D$	0	1×10^{-8}
Elliptic	$f_2(x)=\sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}}x_i^2$	UN	$[-100, 100]^D$	0	1×10^{-8}
SumSquare	$f_3(x)=\sum_{i=1}^{D}ix_i^2$	US	$[-10, 10]^D$	0	1×10^{-8}
sumPower	$f_4(x)=\sum_{i=1}^{D}\lvert x_i\rvert^{(i+1)}$	US	$[-1, 1]^D$	0	1×10^{-8}
Schwefel 2.22	$f_5(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	UN	$[-10, 10]^D$	0	1×10^{-8}
Schwefel 2.21	$f_6(x)=\max\left\{\lvert x_i\rvert,\quad 1\le i\le n\right\}$	UN	$[-100, 100]^D$	0	1×10^{-8}
Step	$f_7(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	US	$[-100, 100]^D$	0	1×10^{-8}
Exponential	$f_8(x)=\exp\left(0.5*\sum_{i=1}^{D}x_i\right)$	US	$[-10, 10]^D$	0	1×10^{-8}
Quartic	$f_9(x)=\sum_{i=1}^{D}ix_i^4+random[0, 1]$	US	$[-1.28, 1.28]^D$	0	1×10^{-1}
Rosenbrock	$f_{10}(x)=\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$	UN	$[-5, 10]^D$	0	1×10^{-1}
Rastrigin	$f_{11}(x)=\sum_{i=1}^{D}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	MS	$[-5.12, 5.12]^D$	0	1×10^{-8}
NCRastrigin	$f_{12}(x)=\sum_{i=1}^{D}\left[y_i^2-10\cos(2\pi y_i)+10\right]$ $y_i=\begin{cases}x_i & \lvert x_i\rvert<\frac{1}{2}\\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\ge\frac{1}{2}\end{cases}$	MS	$[-5.12, 5.12]^D$	0	1×10^{-8}
Griewank	$f_{13}(x)=1/4000\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	MN	$[-600, 600]^D$	0	1×10^{-8}
Schwefel2.26	$f_{14}(x)=418.98288727243380*D-\sum_{i=1}^{D}x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	MS	$[-500, 500]^D$	0	1×10^{-8}
Ackley	$f_{15}(x)=20+e-20\exp\ -0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\Bigg)$ $-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$	MN	$[-50, 50]^D$	0	1×10^{-8}
Penalized1	$f_{16}(x)=\frac{\pi}{D}\Big\{10\sin2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2\left[1+10\sin2\left(\pi y_{i+1}\right)\right]$ $+(y_D-1)^2\Big\}+\sum_{i=1}^{D}u(x_i, 10, 100, 4)$ $y_i=1+1/4(x_i+1),\ u_{y_i,a,k,m}=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a\le x_i\le a\\ k(-x_i-a)^m & x_i<a\end{cases}$	MN	$[-100, 100]^D$	0	1×10^{-8}
Penalized2	$f_{17}(x)=\frac{1}{10}\Big\{\sin2(\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2\left[1+\sin2\left(3\pi x_{i+1}\right)\right]$ $+(x_D-1)^2\left[1+\sin2\left(2\pi x_{i+1}\right)\right]\Big\}+\sum_{i=1}^{D}u(x_i, 5, 100, 4)$	MN	$[-100, 100]^D$	0	1×10^{-8}
Alpine	$f_{18}(x)=\sum_{i=1}^{D-1}\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\rvert$	MS	$[-10, 10]^D$	0	1×10^{-8}
Levy	$f_{19}(x)=\sum_{i=1}^{D-1}(x_i-1)^2\left[1+\sin2\left(3\pi x_{i+1}\right)\right]+\sin2(3\pi_1)$ $+\lvert x_D-1\rvert\left[1+\sin2(3\pi x_D)\right]$	MN	$[-10, 10]^D$	0	1×10^{-8}
Weierstrass	$f_{20}(x)=\sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]\right)-D\sum_{k=0}^{k\max}\Big[a^k$ $\cos\left(2\pi b^k0.5\right)\Big],\ a=0.5, b=3, k_{\max}=20$	MN	$[-1, 1]^D$	0	1×10^{-8}
Himmelblau	$f_{21}(x)=1/D\sum_{i=1}^{D}\left(x_i^4-16x_i^2+5x_i\right)$	MS	$[-5, 5]^D$	−78.33236	−78
Michalewicz	$f_{22}(x)=-\sum_{i=1}^{n}\sin(x_i)\sin20\left(\frac{i\times x_i^2}{\pi}\right)$	MS	$[0, \pi]^D$	−30, −50, −100	−D +1

(5) DFnABC1 denotes that the $x_{k,j}^g$ of Eq. (5) of DFnABC is removed.

(6) DFnABC2 denotes that the explore scope δ of Eq. (5) of DFnABC is removed.

(7) DFnABC3 denotes that the leader x_{pbest}^g of Eq. (5) of DFnABC is removed.

Each algorithm conducts 25 times independent run on all 22 test functions with 30D. For a fair comparison, SN and $limit$ are respectively 50 and $SN\cdot D$, and the maximum number of function evaluations ($maxFES$) is utilized as the termination condition, which is set to $5000\cdot D$. Taking into account the limitations of space, all tables for the detailed experimental data were placed in the supplemental materials, which are marked as "S. Table". The experimental results are given in S. Tables 1–2 and the best results

the similar robustness in the most cases based on the AVEN metric. In addition, the convergence speed of DFnABC is faster than that of DFnABCe and DFnABCo in most cases but it is slower than that of DFnABCs in majority of cases. The reason for this phenomenon may be explained that DFnABCs uses the original selection mechanism (Eq. (4)) for onlooker bee, which only considers the fitness information so as to make the onlooker bee preferentially search around the good solutions. Therefore, the original selection mechanism is able to make the algorithm converge quickly, but it also causes that the algorithm may trap into the local optimal solution easily, such as on f_{11}. While our proposed selection mechanism (Eq. (18)) not only considers the fitness information but also exploits the distance information. Although it may hamper converge, it could be able to make the algorithms search various promising areas and effectively prevent the algorithms from falling into the local are marked with boldface.

As shown in S. Table 1, it can be clearly observed that DFnABC is better than DFnABCe, DFnABCo and DFnABCs regarding to the solution accuracy on the majority of the test functions based on the mean (std) metric. Moreover, all compared algorithms obtain optimal solution. Therefore, although the convergence speed of DFnABC is slower than that of DFnABCs, the robustness of DFnABC is better than DFnABCs in some cases, such as on f_{10} and f_{11}. Moreover, S. Table 1 also shows that DFnABCe, DFnABCo, DFnABCs and DFnABC are better than the original ABC in terms of the

Algorithm 2:The procedure of DFnABC

```
Generate SN solutions that contain D variables according to Eq.(1)
Set count=zeros(1,SN), Scount=zeros(1, SN), Tcount=zeros(1,SN)
while FES < maxFES
    for i =1 to SN
        Select a dimension j and calculate the probability P^e based on Eq. (6)
        Determine the near-good-neighbor x_k^g for x_i^g using roulette according to P^e
        Select the δ and x_pbest^g from the SU^g and top Q solutions respectively
        Generate a new solution v_i^g in the neighbourhood of x_k^g using Eq.(5)
        if f(v_i^g) ≤ f(x_i^g)
            x_i^g = v_i^g , counter(i)=0, Scount(i)= Scount(i)+1
        else
            counter(i)= counter(i)+1
        end if
        Tcount(i)= Tcount(i)+1
    end for
    for i =1 to SN
        Select a dimension j and calculate the probability P based on Eq. (18)
        Select a food source x_s^g for ith onlooker bee using roulette according to P
        Find the best food source x_nbest^g among the neighbours of x_s^g
        Calculate the mean success rate Sr of the neighbours of x_s^g
        Generate a new solution v_s^g using Eq.(20)
        if f(v_s^g) ≤ f(x_s^g)
            x_s^g = v_s^g , counter(s)=0, Scount(s)= Scount(s)+1
        else
            counter(i)= counter(i)+1
        end if
        Tcount(s)= Tcount(s)+1
    end for
    FES=FES+2SN
    Select the solution x_max^g with max counter value
    if counter(max) > limit
        Replace x_max^g by a new solution generated according to Eq.(1)
        FES=FES+1, counter(max)=0, Scount(max)=1, Tcount(max)=1;
    end if
end while
Output: The food source (solution) with the smallest objective value
```

Fig. 5. The pseudo-code of DFnABC.

solution quality and convergence speed. In summary, it can be concluded that all the proposed algorithmic components can bring an improvement to DFnABC. Moreover, the contribution can be further enhanced by combining these three proposed components.

From the results shown in S. Table 2, DFnABC outperforms better than or at least is competitive to DFnABC1, DFnABC2 and DFnABC3 in term of the solution accuracy on all test functions except for f_4, f_6, f_{10}, f_{15} and f_{18}. Moreover, DFnABC can obtain the second best results on f_{18} function based on the mean (std) and AVEN metric. In addition, the convergence speed of DFnABC is faster than that of DFnABC1 and DFnABC3 in most cases. Although the convergence speed of DFnABC is slower than that of DFnABC2 in majority of cases, the solution quality of DFnABC is better than or at least comparable to DFnABC2 on all benchmark functions apart from f_4 and f_{18}. Furthermore, DFnABC1, DFnABC2 and DFnABC3 are better than or at least comparable to DFnABCe in terms of results accuracy and convergence speed on most of functions. For the above phenomenon, the main reasons can be explained as follows. First, the employed bee can fully exploit a near-good-neighbor to accelerate the search of the local areas among the landscape and improve the exploitation quality so that DFnABC outperforms the employed bee will finely search the ambient region of the near-good-neighbor by choosing a small search range. Thus, the convergence speed of DFnABC is slower than that of DFnABC2 but the solution quality is better than that of DFnABC2. Overall, the experimental results indicate that each subtle change (*i.e.*, the replace base vector, the selectable exploration scope and the guided search direction) of the search strategy (Eq. (5)) of DFnABC can make the contribution for the improvement of the proposed algorithm.

In addition, we further provide an additional experiment to verify that these two different neighboring solution generation mechanisms (*i.e.*, Eqs. (5) and (20)) cooperatively used in DFnABC are better than the mechanism that only uses any one of them for employed bee and onlooker bee. To this end, DFnABC will be compared with two ABC variants (*i.e.*, Eqs. (5) and (20) are separately used in DFnABC for employed bee and onlooker bee simultaneously) on 22 test functions with 30*D*, named DFnABCN and DFnABCF respectively. The experimental results are given in S. Table 2. It can be seen that DFnABC is better than or at least comparable to DFnABCF in terms of mean, standard deviation and AVEN on all test functions expect for f_4 and f_{22}. With regard

DFnABC1 in most of cases. Second, the search direction guided by a randomly selected leader from the top Q solutions can speed up the convergence rate, which results in that the performance of DFnABC is better than DFnABC3. Third, the selectable exploration scope makes the employed bee search for the large region of the near-good-neighbor at the early stages. And then, at the later stages, to DFnABCN, DFnABC and DFnABCN show similar final solution quality, but DFnABC has better convergence rate on most of test functions based on AVEN. Therefore, the experimental results show that different neighbor search mechanisms used in employed bee and onlooker bee are better than the same mechanism employ in employed bee and onlooker bee.

Table 2
Parameters setting of all compared algorithms.

Algorithm	Parameters setting
qABC	$SN = 50$, $limit = SN \cdot D$, $r = 1$
BABC	$SN = 50$, $limit = SN \cdot D$, $UR = 0.5$
GABC	$SN = 50$, $limit = SN \cdot D$, $C = 1.5$
CABC	$SN = 50$, $limit = SN \cdot D$
MABC	$SN = 50$, $limit = SN \cdot D$, $P = 0.7$
DFnABC	$SN = 50$, $limit = SN \cdot D$

5.3. Experiment 2: Comparison on 22 benchmark functions

In this subsection, we demonstrate the performance of our proposed method DFnABC by comparing with five state-of-the-art ABC variants (*i.e.*, qABC [16], BABC [13], GABC [20], CABC [18] and MABC [21]) on 22 test functions with 30D, 50D and 100D. For making a fair comparison, SN is set to 50, and $maxFES$ is set to 150000, 250000 and 500000 [54] for functions with 30D, 50D and 100D, respectively. The detailed parameter settings of all algorithms are given in Table 2, which are set the same as the original paper for the compared algorithms. Each algorithm conducts 25 independent runs on all test functions. In addition, the Wilcoxon's rank-sum test [55] that is a nonparametric statistic test for independent samples at 5% significance level is used to show the significant differences between DFnABC and other algorithms. The detailed experimental results are given in S. Tables 3–5 and the best results are marked with boldface. It is noted that the test results ("−", "=", "+") of Wilcoxon's rank-sum test respectively denote that the performance of DFnABC is better than, similar to and worse than that of the corresponding compared algorithms.

From the results of the 30D functions shown in S. Table 3, it is obvious that DFnABC consistently outperforms the compared algorithms in terms of solution accuracy and convergence rate on the majority of the test functions. To be specific, DFnABC performs significantly better than all competitors on unimodal functions f_1-f_3 and f_5. For sumPower function f_4, although DFnABC is beaten by BABC and MABC in terms of the solution quality, it has better convergence rate than all compared algorithms. Moreover, DFnABC outperforms or at least is competitive to all compared algorithms on functions f_7-f_{10}. However, with respect to Schwefel 2.21 f_6, DFnABC is beaten by all compared algorithms. Regarding to the multimodal functions f_{11}-f_{22}, DFnABC can obtain the global optimal solution on functions f_{11}-f_{14} and f_{20}-f_{22}. Moreover, DFnABC is superior to all competitors on Alpine function f_{18}. With respect to Ackley function f_{15}, DFnABC is only beaten by CABC and MABC, but it obtains the best convergence speed. For the remaining cases, DFnABC is better than or at least comparable to all compared algorithms. Overall, DFnABC outperforms qABC, BABC, GABC, CABC and MABC on 17, 8, 15, 8 and 8 cases out of 22 functions. In contrast, DFnABC is only beaten by qABC, BABC, GABC, CABC and MABC on 1, 2, 1, 2 and 3 functions respectively.

Furthermore, we also compare DFnABC with all compared algorithms on the 22 test functions with 50D and 100D to investigate the scalability of DFnABC. Regarding to the results of 50D functions shown in S. Table 4, it also clearly shows that DFnABC has better performance than all compared algorithms in terms of solution accuracy and convergence rate on most test functions. To be specific, DFnABC is better than qABC, BABC, GABC, CABC and MABC on 17, 9, 16, 8 and 9 out of 22 functions respectively, and DFnABC is only search space dimension. To show the performance of the compared algorithms at different evolution stages, the mean and standard deviation of the objective function error value $(f(X_{best}) - f(X^*))$ after $[0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0] \cdot maxFES$ function evaluations [19] for all functions with $D = 50$ are presented in Fig. 6. In Fig. 6, the rectangle and its height represent the mean and standard deviation of function error value, respectively. It clearly indicates that DFnABC always has better solution quality and convergence rate than that of the competitors in during the entire evolution process.

In addition, according to the Friedman test [56], the final ranking of all ABC variants for each benchmark function with 30D, 50D and 100D are shown in S. Table 6. The best results are marked in boldface. Obviously, the average ranking of DFnABC on all functions is better than that of other ABC variants. Overall, this experiment demonstrates that DFnABC is better than or at least competitive to the state-of-the-art ABC variants on these benchmark functions.

5.4. Experiment 3: Comparison on CEC2013 test problems

To demonstrate the performance of DFnABC on more complex problems, in this subsection, we compare DFnABC with five ABC variants (*i.e.*, GABC [20], qABC [16], dABC [25], EABC [22] and MEABC [57]) on 28 CEC2013 test problems with D = 30, 50 and 100, which are derived from the CEC2013 special session on real-parameter optimization [19]. In this experimental study, SN and $limit$ are respectively set to 50 and $SN \cdot D$, and other parameter settings of all competitors are set the same as the settings used in their original papers. According to the requirements of CEC2013 [19], the $maxFES$ for all functions is set to $10000 \cdot D$, and each compared algorithm independently conducts 51 runs on each function. The *mean* and *standard deviation* of the *function error values* $f(X_{best}) - f(X^*)$ are employed to evaluate the optimization performance, where X_{best} is the best solution found by the algorithm in each run and X^* is the true global optimal solution of the test function. Moreover, the Wilcoxon's rank-sum test with the 5% significant level is conducted on the experimental results to obtain the reliable statistic conclusion. The experimental results are given in S. Tables 7–9 for $D = 30$, 50 and 100, respectively. For the sake of clarity, the best results are highlighted in boldface. Note that if the *function error value* $f(X_{best}) - f(X^*)$ is smaller than 10^{-8}, it will be taken as zero [19].

In the case of $D = 30$, it can be seen from S. Table 7 that DFnABC is better than or at least comparable to all compared algorithms in terms of mean and standard deviation on most of test functions. To be specific, DFnABC outperforms GABC, qABC, dABC, EABC and MEABC on 12, 13, 14, 13 and 14 functions respectively. In contrast, DFnABC is only beaten by GABC, qABC, dABC, EABC and MEABC on 5, 6, 5, 6 and 5 functions. Therefore, we can conclude that DFnABC is better than or at least comparable to these state-of-the-art ABC variants on these complex functions with $D = 30$. In the case of $D = 50$, it can be seen from S. Table 8 that DFnABC outperforms or at least is competitive to the compared algorithms in term of solution accuracy in most cases. Concretely speaking, DFnABC is significantly better than GABC, qABC, dABC, EABC and MEABC on 14, 12, 15, 12 and 15 functions, respectively. In contrast, DFnABC is only beaten by GABC, qABC and dABC on 4, 6, 6, 5 and 5 functions. Therefore, DFnABC also performs better than the state-of-the-art ABC variants on these complex test functions with $D = 50$. In the case of $D = 100$, it can be seen from S. Table 9 that the results of 100D function is similar to those of 30D and 50D function. In detail, DFnABC is significantly superior to GABC, qABC, dABC, EABC beaten by qABC, BABC, GABC, CABC and MABC on 1, 3, 1, 2 and 2 functions respectively. Concerning the results of 100D functions as shown in S. Table 5, DFnABC is superior to qABC, BABC, GABC, CABC and MABC on 16, 9, 14, 9 and 10 cases out of 22 functions, respectively, and DFnABC is inferior to qABC, BABC, GABC, CABC and MABC on 2, 3, 1, 2 and 2 functions, respectively. Obviously, the superiority of DFnABC is not affected by the growth of the

and MEABC on 15, 14, 14, 13 and 16 cases out of 28 test functions. However, DFnABC is significantly inferior to GABC, qABC and dABC on 4, 7, 6, 6 and 4 functions, respectively. Evidently, DFnABC is better than the other ABC methods. Overall, these experimental results clearly show that DFnABC performs better than or at least competitive to these state-of-the-art ABC methods on CEC2013 test problems.

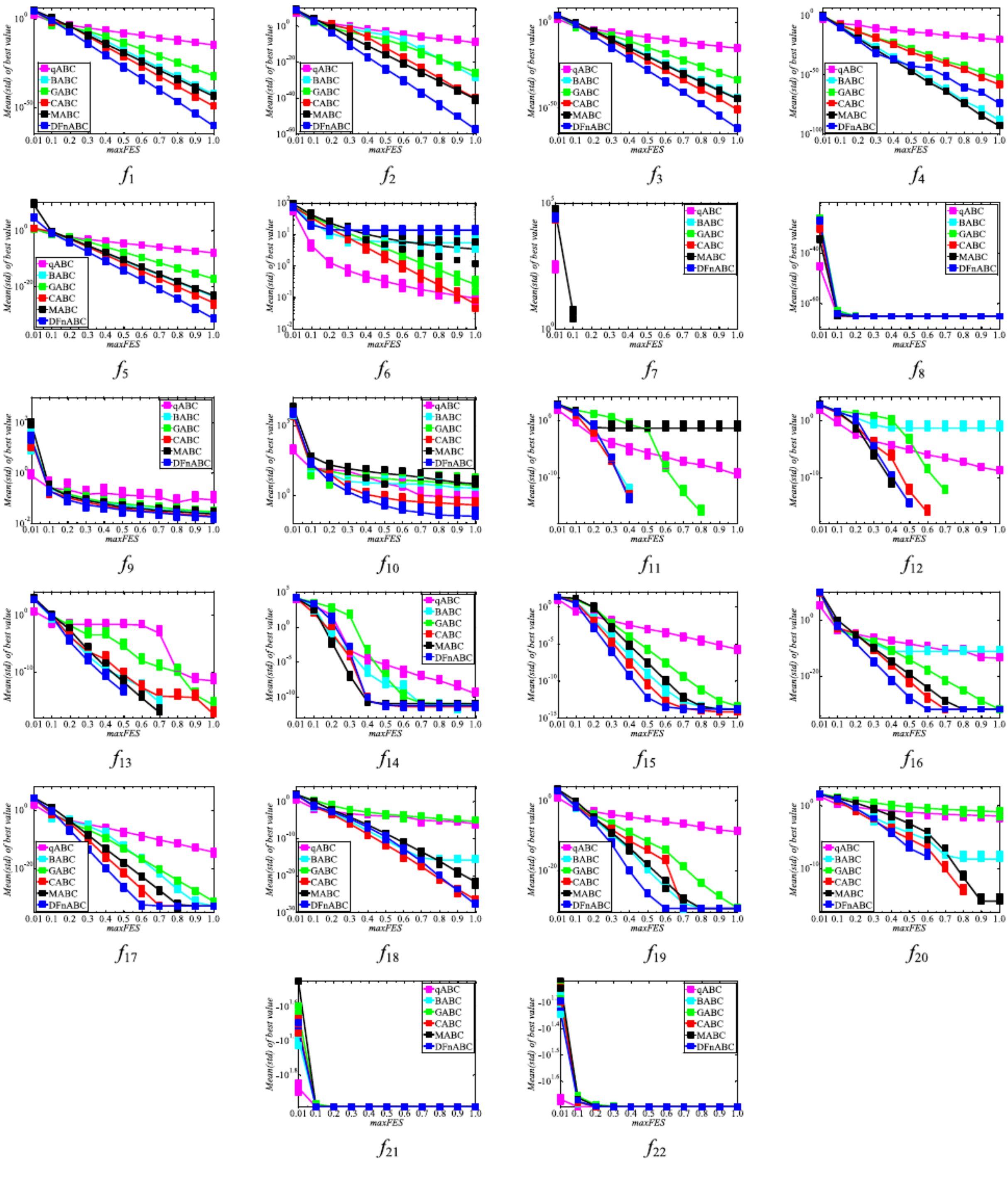


Fig. 6. Convergence curve of different ABCs on 22 test functions with $D = 50$.

5.5. Experiment 4: Comparison on 5 real life optimization problems

(1) RP-1: Gas Transmission compressor design [2]

The purpose of the gas transmission compressor design

In this subsection, we use DFnABC to solve 5 real world optimization problems (short for RP-1 to RP-5) that are well known in various fields of engineering designs. All the problems considered in this study are unconstrained and highly nonlinear. The dimensions of these problems vary from 2 to 20. For the fifth problem (RP-5), two dimension sizes (*i.e.*, $D = 10$ and $D = 20$) are tested and named as RP-5a and RP-5b, respectively. The detailed introduction of the 5 ROPs is given as follows.

problem is to find the optimal values of design parameters x_1, x_2 and x_3 when 100 million cu. Ft. of gas per day are delivered with minimum cost for a gas pipe line transmission system. x_1 is the length between compressor stations (in miles), x_2 is the compressor ratio and x_3 is the pipe inside diameter (in inches). The mathematical model of RP-1 is

defined as follow.

$$\begin{aligned}\text{Min } f(x) = {} & 8.61 \times 10^5 x_1^{1/2} x_2 x_3^{-2/3} \left(x_2^2 - 1\right)^{-1/2} \\ & + 3.69 \times 10^4 x_3 \\ & + 7.72 \times 10^8 x_1^{-1} x_2^{0.219} - 765.43 \times 10^6 x_1^{-1}\end{aligned}$$

Bounds: $10 \le x_1 \le 55, \quad 1.1 \le x_2 \le 2, \quad 10 \le x_3 \le 40$

(2) RP-2: Optimal capacity of gas production facility [2]
This practical problem aims at finding the optimum combination of capacities of the production facilities of an oxygen production and inventory system. The values of design parameters, *i.e.*, the oxygen production rate (x_1) and the oxygen storage pressure (x_2), need to be optimized for saving costs. The mathematical model of RP-2 is given as follow.

$$\begin{aligned}\text{Min } f(x) = {} & 61.8 + 5.72 x_1 \\ & + 0.2623[(40 - x_1)\ln(x_2/200)]^{-0.85} \\ & + 0.087(40 - x_1)\ln(x_2/200) + 700.23 x_2^{-0.75}\end{aligned}$$

Bounds: $17.5 \le x_1 \le 40, \quad 300 \le x_2 \le 600$

(3) RP-3: Gear Train Design [3]
The design requirement of compound gear train is that the gear ratio is as close as possible to 1/6.931. For each gear, the number of teeth must be in the range of [12,58]. Since the number of teeth must be an integer, the variables must be integers. The mathematical model of RP-3 is given as follow.

$$\text{Min } f(x) = \left\{\frac{1}{6{,}931} - \frac{T_d T_b}{T_a T_f}\right\}^2 = \left\{\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right\}^2$$

Bounds: $12 \le x_i \le 60, \quad i = 1, 2, 3, 4$

$[x_1, x_2, x_3, x_4] = [T_d, T_b, T_a, T_f]$, x_i should be integers.

T_a, T_b, T_d and T_f are the number of teeth on gears *a*, *b*, *d* and *f*, respectively.

(4) RP-4: Frequency modulation sounds parameter identification [4]
The frequency modulation sound model is a highly complex multimodal problem with minimum value zero. It is mathematically represented as follows:

$$\begin{aligned}y(t) = {} & a_1 \times \sin(w_1 \times t \times \theta + a_2 \\ & \times \sin(w_2 \times t \times \theta + a_3 \times \sin(w_3 \times t \times \theta)))\end{aligned}$$

where $\theta = 2\pi/100$ and a_1, w_1, a_2, w_2, a_3, w_3 are six parameters in the bounds $[-6.4, 6.35]$. The fitness function is defined as the sum of square error between the evolved data and the model data as follows:

$$f(a_1, w_1, a_2, w_2, a_3, w_3) = \sum_{t=0}^{100} (y(t) - y_0(y))^2$$

$$\begin{aligned}y_0(t) = {} & 1.0 \times \sin(5.0 \times t \times \theta \\ & + 1.5 \times \sin(4.8 \times t \times \theta + 2.0\end{aligned}$$

where $X = \left\{(x_1, \ldots, x_n) \in R^n \,\middle|\, 0 \le x_j \le 2\pi, j = 1, 2, \ldots, n\right\}$ and $m = 2n - 1$, with:

$$f_{2i-1}(x) = \sum_{j=i}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right), \quad i = 1, 2, \ldots, n;$$

$$f_{2i-1}(x) = 0.5 + \sum_{j=i+1}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right),$$

$i = 1, 2, \ldots, n - 1;$

$$f_{m+i}(X) = -f_i(X), \quad i = 1, 2, \ldots, m.$$

Moreover, to clearly show the image of this problem, its images for $n = 2$ are shown in Fig. 7.

DFnABC is compared with six ABC variants, *i.e.* ABC [11], GABC [20], BsfABC [59], CABC [18], ABCVSS [28] and EABC [22] on these 5 real world problems. To make a fair comparison, *SN* and *limit* of all compared algorithms are respectively set to 50 and $SN \cdot D$, and the maximum number of function evaluations (*maxFES*) is used as the termination condition, which is set depend on each problem. Each algorithm conducts 30 independent runs on each problem. The parameters of DFnABC are set the same as above experiment and the parameters of the competitors are kept the same as the original papers. The mean (Mean) and standard deviation (std.dev.) of the best objective function value over 30 independent runs are used to evaluate the performance. The results are given in S. Table 10 and the best results are marked in boldface.

As shown in S. Table 10, all algorithms show the similar performance on RP-1 and RP-2 in terms of the mean best objective function value. Moreover, DFnABC can get the best and second best standard deviation value on RP-1 and RP-2 respectively. For RP-3, DFnABC is better than ABC, GABC and EABC, but it is worse than BsfABC, CABC and ABCVSS. Considering the RP-4, DFnABC performs significantly better than all competitors. With regard to RP-5, DFnABC obtains the best and second best result on 10*D* and 20*D*, respectively. Moreover, based on the Friedman test [56], the final rankings of all ABC variants on each design problem are shown in S. Table 11 and the best results are marked in boldface. Obviously, DFnABC ranks first when considering all problems.

Overall, all experimental results demonstrate that our proposed algorithm DFnABC is able to effectively deal with the benchmark test problems and the real world engineering optimization problems.

5.6. Experiment 5: Comparisons of time complexity

In original ABC, the time complexity in each generation is $o(SN \cdot D)$ [13]. For DFnABC, in employed bee phase, the maximum time complexity of selecting the near-good-neighbor x_k, selecting the explore scope δ and choosing the leader x_{pbest} are $o(SN \cdot D)$, $o(SN \cdot \log SN \cdot D)$ and $o(SN \cdot \log SN \cdot D)$, respectively. Therefore, the times complexity of employed bee phase in each generation is $o(SN \cdot \log SN \cdot D)$. In onlooker bee phase, the time complexity of calculating the selection probability and selecting the best position of the neighbors are $o(SN \cdot D)$ and $o(SN \cdot D)$ respectively. Thus, the times complexity of onlooker bee phase in each generation is $o(SN \cdot D)$. Totally, the computational complexity of DFnABC is

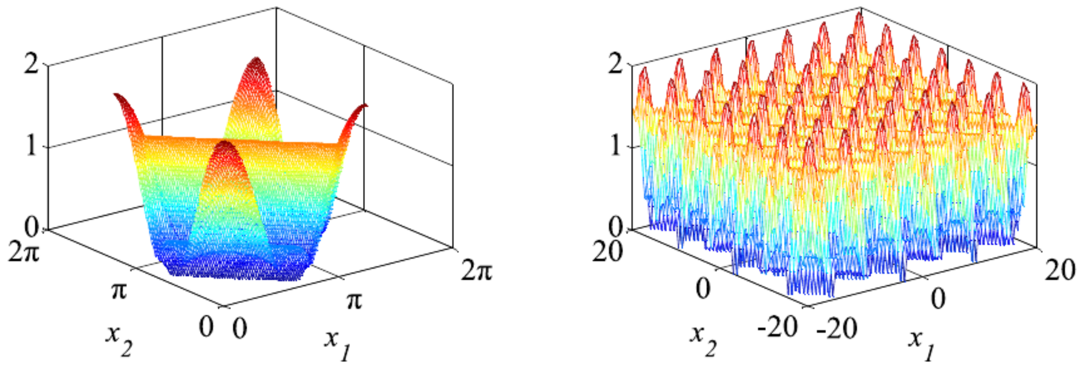
$\times \sin(4.9 \times t \times \theta)))$

(5) RP-5: The spread spectrum radar polyphase code design [5] This problem is a continuous min–max global optimization problem. Its objective function is piece wise smooth and it has numerous local optimal solutions. The mathematical model of RP-5 is represented as follows:

$$\text{Min } f(x) = \max\{f_1(X), \ldots, f_{2m}(X)\}$$

$o(SN \cdot \log SN \cdot D)$ in each generation.

In addition, we compare the AET (*i.e.*, the average execution time when function evaluation times (*FES*) reaches the *maxFES*) and ABT (*i.e.*, average runtimes when the function error reaches the acceptable value) between DFnABC and five ABC variants (*i.e.*, qABC, BABC, GABC, CABC and MABC) on 22 commonly benchmark functions ($D = 30, 50, 100$), all of which are conducted 25 independent runs. Moreover, we also record the AET of all competitors on CEC2013 ($D = 30, 50, 100$) and 5 real life problems. Note that since



(a): Search range is $[0, 2\pi]$ for each variable (b): Search range is $[-20, 20]$ for each variable

Fig. 7. The image of RP-5 in 2-*D* space.

Table 3
Average runtime (in seconds) used by different contenders.

D	*mean*											
	22 benchmark functions											
	AET						ABT					
	qABC	BABC	GABC	CABC	MABC	DFnABC	qABC	BABC	GABC	CABC	MABC	DFnABC
30	31.073	30.778	30.002	31.093	**5.986**	13.625	13.931	8.087	11.169	9.799	**1.350**	3.846
50	56.690	58.095	55.562	57.276	**12.901**	26.423	18.984	20.089	18.239	19.167	**6.530**	8.089
100	127.52	123.93	122.14	120.29	**37.305**	67.359	46.232	49.725	33.768	42.396	22.318	**21.516**

D	*mean*					
	28 CEC2018 functions (AET)					
	GABC	qABC	dABC	EABC	MEABC	DFnABC
30	58.422	59.911	60.443	76.684	**15.142**	30.026
50	107.82	111.43	112.77	124.89	**32.531**	53.519
100	328.43	338.31	327.39	294.39	**156.61**	196.49

RP	*mean*						
	5 real life problems (AET)						
	ABC	GABC	BsfABC	CABC	ABCVSS	EABC	DFnABC
RP-1	5.152	5.710	5.018	5.564	6.233	3.349	**2.145**
RP-2	3.376	3.769	3.366	3.857	4.274	2.235	**1.454**
RP-3	6.837	7.489	7.031	7.727	8.834	4.495	**2.854**
RP-4	5.703	5.282	5.176	5.441	6.085	10.268	**3.362**
RP-5a	7.895	6.211	6.109	6.687	7.700	12.767	**3.270**
RP-5b	19.831	15.676	15.968	16.544	18.531	32.710	**10.439**
Avg	8.132	7.356	7.111	7.637	8.610	9.881	**3.921**

the acceptable values of CEC2013 and 5 real life problems are not provided, we only give the AET value. The final summary results are shown in Table 3 and the detailed data are provided in S. Tables 12–15 of the supplementary file.

For 22 benchmark functions with $D = 30, 50, 100$, it can be seen from Table 3 that AET and ABT of DFnABC are less than those of qABC, BABC, GABC and CABC in all cases. However, DFnABC costs more time than MABC on 30D and 50D functions. But, with regards to 100D functions, MABC needs more time. The reason for these $D = 30, 50, 100$, the AET of DFnABC is superior to GABC, qABC, dABC and EABC, while it is worse than MEABC. This is also because that MEABC does not include the onlooker bee phase. With regard to the 5 real life problems, DFnABC can get the smallest AET in all cases.

5.7. Experiment 6: Comparisons with non-ABC methods and state-of-the-art ABC variants

phenomena can be explained that for qABC, BABC, GABC and CABC, each onlooker bee selects the food source by the roulette wheel method, which is a very time-consuming process since the selected probabilities of all food sources are very small. But, MABC does not utilize the roulette wheel method so that its AET and ABT is the lowest. For DFnABC, the selected probabilities of food sources are significantly augmented by Eq. (18) compared with Eq. (4), and thus the utilization frequency of roulette wheel method is apparently reduced. So, AET and ABT of DFnABC are simultaneously better than all contenders except for MABC. For CEC2013 with

In this subsection, DFnABC is compared with three non-ABC variants (*i.e.*, SMO [60], CLPSO [61], jDEscop [58]) and two state-of-the-art ABC variants (*i.e.*, AABCLS [62], LFABC [63]). Each algorithm will be run 30 times for 22 test functions with 30*D*. In this experiment, the parameter settings of DFnABC are the same as that of ***Experiment* 2**. For a fair comparison, *SN* and *limit* are 50 and $SN \cdot D$, and the maximum number of function evaluations (*maxFES*) is utilized as the termination condition, which is set to $5000 \cdot D$. The parameters of other comparison algorithms are set as the suggestion in their original papers. Furthermore, in order

to show the significant differences between DFnABC and other algorithms, Wilcoxon's rank sum test at 5% significance level is conducted. For clarity, we mark the results of the best algorithms in boldface.

The experimental results are shown in S. Table 16. It can be seen from S. Table 16 that DFnABC is better than or at least comparable to other contenders on almost all the cases. DFnABC obtains the second best results on f_4 and gets the worst performance on f_6. With regard to the noisy quartic function f_9, DFnABC only outperforms LFABC. Overall, DFnABC is better than SMO, CLPSO, jDEscop, AABCLS and LFABC on 16, 18, 13, 10 and 15 out of 22 functions, respectively, and DFnABC is only beaten by SMO, CLPSO, jDEscop, AABCLS and LFABC on 2, 2, 3, 2 and 1 function, respectively. Moreover, according to the Friedman test [54], the average ranking of DFnABC for all functions is better than that of other contenders, which is shown in S. Table 17. Based on the above analysis, it can be concluded that DFnABC is better than or at least comparable to these non-ABC variants and state-of-the-art ABC variants on 22 test functions with $D = 30$.

6. Conclusion and future work

In this paper, two novel search strategies based on the distance-fitness-based neighbor information are introduced to generate the candidate solution for employed bee and onlooker bee, respectively. Moreover, a new selection mechanism is designed for onlooker bee, which makes the onlooker bee preferentially choose the food source positions that not only have high quality but also are far away from the current best position to search. By synergizing these three operations, we develop a novel ABC variant, named after DFnABC. The performance of our proposed algorithm is validated by the comparisons with some state-of-the-art ABC variants on 22 benchmark functions, 28 CEC2013 test functions and 5 real world engineering optimization problems. The experimental results prove that the performance of DFnABC is significantly better than that of some state-of-the-art ABC variants.

In the future, we will extend our work for the study on how to integrate the ABC algorithm with upcoming optimizers, like Group Counseling Optimizer [64], and how to improve the optimization performance for solving some real-life applications, such as energy optimization in cloud computing [65], air-based information network [66], convex optimization problems [67–69] and deblurring images corrupted [70].

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants 61772345, 61402294, 61732011 and 61672358, Major Fundamental Research Project in the Science and Technology Plan of Shenzhen, China under Grants JCYJ20160310095523765 and JCYJ20160307111232895.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.future.2018.06.054.

[5] N. Mladenovic, J. Petrovic, V.K. Vujcic, M. Cangalovic, Solving spread spectrum radar poly phase code design problem by tabu search and variable neighborhood search, European J. Oper. Res. 151 (2) (2003) 389–399.
[6] J. Wang, Y.Q. Gao, B. Li, H.J. Kim, S.Y. Lee, Particle swarm optimization based clustering algorithm with mobile sink for WSNs, Futere Gener. Comp. Syst. 67 (2017) 452–457.
[7] G.H. Li, L.Z. Cui, X.H. Fu, Z.K. Wen, N. Lu, J. Lu, Artificial bee colony algorithm with gene recombination for numerical function optimization, Appl. Soft Comput. 52 (2017) 146–159.
[8] M.K. Qiu, M. Zhong, J. Li, K. Gai, Z. Zong, Phase-change memory optimization for green cloud with genetic algorithm, IEEE Trans. Comput. 64 (12) (2015) 3528–3540.
[9] M.K. Qiu, Z. Chen, J.W. Niu, Z.L. Zong, G. Quan, X. Qin, L.T. Yang, Data allocation for hybrid memory with genetic algorithm, IEEE Trans. Emerg. Top. Commun. 3 (4) (2015) 544–555.
[10] L.Z. Cui, G.H. Li, Q.Z. Lin, J.Y. Chen, N. Lu, Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations, Comput. Oper. Res. 67 (2015) 155–173.
[11] L.B. Ma, K.Y. Hu, Y.L. Zhu, B. Niu, H.N. Chen, M.W. He, Discrete and continuous optimization based on hierarchical artificial bee colon optimizer, J. Appl. Math. 2014 (2014) 1–20.
[12] L.Z. Cui, K. Zhang, G.H. Li, X.H. Fu, Z.K. Wen, Modified Gbest-guided artificial bee colony algorithm with new probability model, Soft Comput. (2017). http://dx.doi.org/10.1007/s00500-017-2485-y.
[13] W. Gao, F.T.S. Chan, L. Huang, S. Liu, Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood, Inform. Sci. 316 (2015) 180–200.
[14] L.Z. Cui, G.H. Li, Q.Z. Lin, Z.H. Du, W.F. Gao, J.Y. Chen, N. Lu, A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation, Inform. Sci. 367–368 (2016) 1012–1044.
[15] M.S. Kiran, A. Babalik, Improved artificial bee colony algorithm for continuous optimization problems, J. Comput. Commun. 2 (4) (2014) 108–116.
[16] D. Karaboga, B. Gorkemli, A quick artificial bee colony (qABC) algorithm and its performance on optimization problems, Appl. Soft Comput. 23 (5) (2014) 227–238.
[17] L.Z. Cui, G.H. Li, Q.Z. Lin, J.Y. Chen, N. Lu, G.J. Zhang, Artificial bee colony algorithm based on neighboring information learning, in: ICONIP 2016: Neural Information Processing, 2016, pp. 279–289.
[18] W.F. Gao, S.Y. Liu, L.L. Huang, A novel artificial bee colony algorithm based on modified search equation and orthogonal learning, IEEE Trans. Cybern. 43 (3) (2013) 1011–1024.
[19] J.J. Liang, B.Y. Qu, P.N. Suganthan, G.H. Alfredo, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, in: Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, January 2013.
[20] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, Appl. Math. Comput. 217 (7) (2010) 3166–3173.
[21] W.F. Gao, S.Y. Liu, A modified artificial bee colony algorithm, Comput. Oper. Res. 39 (3) (2012) 687–697.
[22] W.F. Gao, S.Y. Liu, L.L. Huang, Enhancing artificial bee colony algorithm using more information-based search equations, Inform. Sci. 270 (1) (2014) 112–133.
[23] J. Luo, Q. Wang, X.H. Xiao, A modified artificial bee colony algorithm based on converge-onlookers approach for global optimization, Appl. Math. Comput. 219 (20) (2013) 10253–10262.
[24] Y.J. Shi, C.M. Pun, H.D. Hu, H. Gao, An improved artificial bee colony and its application, Knowl. Based Syst. 107 (2016) 14–31.
[25] M.S. Kiran, O. Findik, A directed artificial bee colony algorithm, Appl. Soft Comput. 26 (2015) 454–462.
[26] I. Babaoglu, Artificial bee colony algorithm with distribution-based update rule, Appl. Soft Comput. 34 (2015) 851–861.
[27] J.J. Liu, H.Q. Zhu, Q. Ma, L.L. Zhang, H.L. Xu, An artificial bee colony algorithm with guide of global & local optima and asynchronous scaling factors for numerical optimization, Appl. Soft Comput. 37 (2015) 608–618.
[28] M.S. Kiran, H. Hakli, M. Gunduz, H. Uguz, Artificial bee colony algorithm with

References

[1] L.F. Zhang, Y.H. Hu, C. Li, J.C. Yao, A new linear convergence result for the iterative soft thresholding algorithm, Optimization 66 (7) (2017) 1–13.
[2] C.S. Beightler, D.T. Phillips, Applied Geometric Programming, Vol. 28, Wiley, 1976 (2).
[3] G.C. Onwubolu, B.V. Babu, New Optimization Techniques in Engineering, Vol. 141, Springer Berlin Heidelberg, 2004.
[4] C.G. Martínez, M. Lozano, F. Herrera, D. Molina, A.M. Sánchez, Global and local real coded genetic algorithms based on parent centric crossover operators, European J. Oper. Res. 185 (3) (2008) 1088–1113.

variable search strategy for continuous optimization, Inform. Sci. 300 (2015) 140–157.
[29] H. Wang, Z. Wu, S. Rahnamayan, H. Sun, Y. Liu, J. Pan, Multi-strategy ensemble artificial bee colony algorithm, Inform. Sci. 279 (2014) 587–603.
[30] W.F. Gao, L.L. Huang, S.Y. Liu, F.T.S. Chan, C. Dai, Artificial bee colony algorithm with multiple search strategies, Appl. Math. Comput. 271 (2015) 269–287.
[31] Y. Alkin, E. Erdal, An adaptive artificial bee colony algorithm for global optimization, Appl. Math. Comput. 271 (2015) 1004–1023.
[32] F. Kang, J.J. Li, Z.Y. Ma, Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, Inform. Sci. 181 (16) (2011) 3508–3531.

[33] F. Kang, J.J. Li, H.J. Li, Artificial bee colony algorithm and pattern search hybridized for global optimization, Appl. Soft Comput. 13 (4) (2013) 1781–1791.
[34] P. Loubiere, A. Jourdan, P. Siarry, R. Chelouah, A sensitivity analysis method for driving the artificial bee colony algorithm's search process, Appl. Soft Comput. 41 (2016) 515–531.
[35] C.Q. Zhang, J.G. Zheng, Y.Q. Zhou, Two modified artificial bee colony algorithms inspired by Grenade explosion method, Neurocomputing 151 (3)(2015) 1198–1207.
[36] W.F. Gao, S.Y. Liu, L.L. Huang, A novel artificial bee colony algorithm with powell's method, Appl. Soft Comput. 13 (9) (2013) 3763–3775.
[37] A. Banitalebi, M.I.A. Aziz, A. Bahar, Z.A. Aziz, Enhanced compact artificial bee colony, Inform. Sci. 298 (2015) 491–511.
[38] X. Li, G.F. Yang, Artificial bee colony algorithm with memory, Appl. Soft Comput. 41 (2016) 362–372.
[39] T. Bayraktar, A Memory-Integrated Artificial Bee Algorithm for Heuristic Optimization, Bayraktar Tugrul, 2014.
[40] W. Xiang, S. Ma, M. An, hABCDE: A hybrid evolutionary algorithm based on artificial bee colony algorithm and differential evolution, Appl. Math. Comput. 238 (238) (2014) 370–386.
[41] W.F. Gao, L.L. Huang, J. Wang, S.Y. Liu, C.D. Qin, Enhanced artificial bee colony algorithm through differential evolution, Appl. Soft Comput. 48 (2016) 137–150.
[42] A. Abraham, R.K. Jatoth, A. Rajasekhar, Hybrid differential artificial bee colony algorithm, J. Comput. Theor. Nanosci. 9 (2) (2012) 249–257.
[43] T.J. Hsieh, H.F. Hsiao, W.C. Yeh, Mining financial distress trend data using penalty guided support vector machines based on hybrid of particle swarm optimization and artificial bee colony algorithm, Neurocomputing 82 (4) (2012) 196–206.
[44] H.C. Tsai, Integrating the artificial bee colony and bees algorithm to face constrained optimization problems, Inform. Sci. 258 (3) (2014) 80–93.
[45] S.S. Jadon, R. Tiwari, H. Sharma, J.C. Bansal, Hybrid artificial bee colony algorithm with differential evolution, Appl. Soft Comput. (2017). http://dx.doi.org/10.1016/j.asoc.2017.04.018.
[46] C. Ozturk, E. Hancer, D. Karaboga, A novel binary artificial bee colony algorithm based on genetic operators, Inform. Sci. 297 (2015) 154–170.
[47] X.T. Li, M.G. Yin, Parameter estimation for chaotic systems by hybrid differential evolution algorithm and artificial bee colony algorithm, Nonlinear Dyn. 77 (1–2) (2014) 1–11.
[48] Z.H. Ding, Z.R. Lu, J.K. Liu, Parameters identificiation of chaotic systems based on artificial bee colony algorithm combined with cuckoo search strategy, Sci. China Technol. Sci. (2017). http://dx.doi.org/10.1007/s11431-016-9026-4.
[49] S.M. Chen, A. Sarosh, Y.F. Dong, Simulated annealing based artificial bee colony algorithm for global numerical optimization, Appl. Math. Comput. 219 (8) (2012) 3575–3589.
[50] I. Fister, I.J. Fister, J.B.V. Zumer, Memetic artificial bee colony algorithm for large-scale global optimization, in: 2012 IEEE Congress on Evolutionary Computation (2012 CEC) (2012).
[51] M. Tuba, N. Bacanin, Artificial bee colony algorithm hybridized with firefly algorithm for cardinality constrained mean–variance portfolio selection problem, Appl. Math. Inf. Sci. 8 (6) (2014) 2831–2844.
[52] S. Biswas, S. Kundu, D. Bose, S. Das, P.N. Suganthan, Synchronizing differential evolution with a modified affinity-based mutation framework, in: 2013 IEEE Symposium on Differential Evolution, Vol. 19 (18) (2013), pp. 61–68.
[53] S. Biswas, S. Das, S. Kundu, G.R. Patra, Utilizing time-linkage property in DOPs: An information sharing based artificial bee colony algorithm for tracking multiple optima in uncertain environments, Soft Comput. 18 (6) (2014) 1199–1212.
[54] X.Y. Zhou, H. Wang, M.W. Wang, J.Y. Wan, Enhancing the modified artificial bee colony algorithm with neighborhood search, Soft. Comput. 21 (10) (2017) 2733–2743.
[55] H. Shan, T. Yasuda, K. Ohkura, A self adaptive hybrid enhanced artificial bee colony algorithm for continuous optimization problems, Bio. Syst. 132–133 (7) (2015) 43–53.
[62] S.S. Jadon, J.C. Bansal, R. Tiwari, H. Sharma, Accelerating artificial bee colony algorithm with adaptive local search, Memetic. Comp. 7 (2015) 215–230.
[63] H. Sharma, J.C. Bansal, K.V. Arya, X.S. Yang, Lévy flight artificial bee colony algorithm, Int. J. Syst. Sci. 47 (11) (2016) 2652–2670.
[64] S. Biswas, M.A. Eita, S. Das, A.V. Vasilakos, Evaluation the performance of group counseling optimizer on CEC 2014 problems for computational expensive optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC) July 6-11, 2014, Beijing, China.
[65] Y.B. Li, M. Chen, W.Y. Dai, M.K. Qiu, Energy optimization with dynamic task scheduling mobile cloud computing, IEEE Syst. J. 111 (1) (2017) 96–105.
[66] X. Liu, M.K. Qiu, X.D. Wang, W.R. Liu, K.Q. Cai, Energy efficiency optimization for communication of air-based information network with guaranteed timing constraints, Signal Process. Syst. 86 (2–3) (2017) 299–312.
[67] Y.H. Hu, C.K.W. Yu, C. Li, stochastic subgradient method for quasi-convex optimization problems, J. Nonlinear Convex. A 17 (4) (2016) 711–724.
[68] Y.H. Hu, C.K. Sim, X.Q. Yang, A subgradient method based on gradient sampling for solving convex optimization problems, Numer. Func. Anal. Opt. 36 (12) (2015) 1559–1584.
[69] Y.H. Hu, C.K.W. Yu, C. Li, X.Q. Yang, Conditional subgradient methods for constrained quasi-convex optimization problems, J Nonlinear Convex. A 17 (10) (2016) 2143–2158.
[70] J. Lu, Z.P. Yang, L.X. Shen, Z. Lu, H. Yang, C. Xu, A framelet algorithm for Deblurring images corrupted by multiplicative noise, Appl. Math. Model. (2018). http://dx.doi.org/10.1016/j.apm.2018.05.007.

Laizhong Cui received the B.S. degree from Jilin University, Changchun, China, in 2007 and Ph.D. degree in computer science and technology from Tsinghua University, Beijing, China, in 2012. He is currently an associate professor in the College of Computer Science and Software Engineering at Shenzhen University, China. He led a project of the National Natural Science Foundation, and several projects of Guangdong Province and Shenzhen City. His research interests include future Internet architecture, edge Computing, IoT, computational intelligence, software-defined network and machine learning.

Kai Zhang was born in Xiangtan, Hunan, in 1992. He received the M.Sc. degrees in computer science from Shenzhen University, Shenzhen, China, in 2018. His current research interests include evolutionary computation and machine learning.

Genghui Li received the M.Sc degree from Shenzhen University, Shenzhen, China, in 2016. He is pursuing his Ph.D degree in computer science from City University of Hong Kong, Hong Kong. His research include evolutionary computation, computational intelligence, and machine learning.

[56] Y. Xiang, Y. Zhou, A dynamic multi-colony artificial bee colony algorithm for multi-objective optimization, Appl. Soft Comput. 35 (2015) 766–785.
[57] H. Wang, Z.J. Wu, S. Rahnamayan, H. Sun, Y. Liu, J.S. Pan, Multi-strategy ensemble artificial bee colony algorithm, Inform. Sci. 279 (2014) 587–603.
[58] J. Brest, M.S. Maucec, Self-adptive differential evolution algorithm using population size reduction and three strategies, Soft. Comput. 15 (2011) 2157–2174.
[59] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, The best-so-far selection in Artificial Bee Colony algorithm, Appl. Soft Comput. 11 (2) (2011) 2888–2901.
[60] J.C. Bansal, H. Sharma, S.S. Jadon, M. Clerc, Spider monkey optimization algorithm for numerical optimization, Memet. Comput. 6 (1) (2014) 31–47.
[61] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.

Xizhao Wang received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1998. He is currently a professor of the College of Computer Science and Software Engineering, Shenzhen University, Guangdong, China. He has more than 180 publications, including four books, seven book chapters, and more than 100 journal papers in the IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Systems, Man, and Cybernetics, IEEE Transactions on Fuzzy Systems, etc. His current research interests include fuzzy measures and integrals, feature extraction, and applications of machine learning.

Shu Yang received the B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2009 and Ph.D. degree in computer science and technology from Tsinghua University, Beijing, China, in 2014. He is currently an associate research fellow in the College of Computer Science and Software Engineering at Shenzhen University, China. His research interests include future Internet architecture, edge Computing, IoT, and blockchain.

Joshua Zhexue Huang is a distinguished professor in College of Computer Science and Software Engineering, Shenzhen University, China and he is director of Big Data Institute, Shenzhen University, China. He received his PhD degree from the Royal Institute of Technology in Sweden in 1993. Since 2009, he served as Chief scientist of High performance computing center, Shenzhen Institute of Advanced Technology, CAS. His interest is in Data mining, machine learning and cloud computing.

Zhong Ming received the Ph.D. degree in Computer Science from Sun Yat-sen University, Guangzhou, China. He is currently a professor of the College of Computer Science and Software Engineering, Shenzhen University, Guangdong, China. He has published more than 100 high-quality papers in top conferences and journals such as IEEE Transactions on Computers, ACM Transactions on Embedded Computing Systems, ACM Transactions on Autonomous and Adaptive Systems, IEEE International Conference on Data Mining (ICDM), etc. His current research interests include cloud computing, internet of things, software engineering, and artificial intelligence.

Nan Lu is a professor in the College of Computer Science and Software Engineering at Shenzhen University, China. He received the Ph.D. degree from Jilin University, Changchun, China. He led several projects of Guangdong Province and Shenzhen City. His research interests include commerce intelligence, machine learning, complex network community structure, and trust mining of social network.