Improving Metaheuristic Algorithms With Information Feedback Models

Gai-Ge Wang, *Member, IEEE*, and Ying Tan, *Senior Member, IEEE*

***Abstract*—In most metaheuristic algorithms, the updating process fails to make use of information available from individuals in previous iterations. If this useful information could be exploited fully and used in the later optimization process, the quality of the succeeding solutions would be improved significantly. This paper presents our method for reusing the valuable information available from previous individuals to guide later search. In our approach, previous useful information was fed back to the updating process. We proposed six information feedback models. In these models, individuals from previous iterations were selected in either a fixed or random manner. Their useful information was incorporated into the updating process. Accordingly, an individual at the current iteration was updated based on the basic algorithm plus some selected previous individuals by using a simple fitness weighting method. By incorporating six different information feedback models into ten metaheuristic algorithms, this approach provided a number of variants of the basic algorithms. We demonstrated experimentally that the variants outperformed the basic algorithms significantly on 14 standard test functions and 10 CEC 2011 real world problems, thereby, establishing the value of the information feedback models.**

***Index Terms*—Benchmark, evolutionary algorithms (EAs), evolutionary computation, information feedback, metaheuristic algorithms, optimization algorithms, swarm intelligence.**

I. Introduction

IN VARIOUS aspects of daily life, people try their best to maximize their benefits and minimize their costs. This type of reasoning is modeled mathematically by optimization problems. In mathematics, computer science, decision-making, and other fields, optimization problems seek the maximum or minimum value of a given objective function. These problems are often approached using optimization algorithms. Optimization algorithms can be divided loosely into two categories: 1) the traditional deterministic methods and 2) modern metaheuristic algorithms. The former will generate the same results for different runs under the same conditions. For the latter, different runs will generate different solutions in most cases, even under the same conditions. Because metaheuristic algorithms can solve many complicated problems successfully, they have received increased attention in many fields, ranging from academic research to engineering practice.

Inspired by nature, a variety of metaheuristic algorithms have been proposed recently to deal with complicated optimization problems [1]–[5]. Many of them have solved complex, challenging problems that are difficult to approach using traditional mathematical optimization techniques. These nature-inspired algorithms include ant colony optimization (ACO) [6], [7], artificial bee colony [8], [9], differential evolution (DE) [10]–[12], evolutionary strategy (ES) [13], cuckoo search (CS) [14], [15], fireworks algorithm (FWA) [16], brain storm optimization [17], [18], earthworm optimization algorithm [19], elephant herding optimization [20], krill herd (KH) [21]–[28], biogeography-based optimization (BBO) [29], genetic algorithm (GA) [30]–[32], harmony search (HS) [33]–[35], monarch butterfly optimization (MBO) [36], probability-based incremental learning (PBIL) [37], moth search algorithm [38], particle swarm optimization (PSO) [39]–[46], and bat algorithm (BA) [47], [48].

However, these basic metaheuristic algorithms have failed to make full use of valuable information available from the individuals in previous iterations to guide their current and later search. Some of them, such as ABC [8], ACO [6], [49], BA [47], and BBO [29], [50], abandon previous instances directly. Others, such as CS [14], FWA [16], [51], PSO [39]–[42], KH [21], [22], and MBO [36], use only the best previous individuals. In practice, any of the previous individuals could contain a variety of useful information. If such information could be fully exploited and utilized in the later optimization process, the performance of these metaheuristic algorithms surely would be significantly improved.

Accordingly, many researchers enhanced these metaheuristic algorithms, and some useful information obtained from the surrogate, an individual, the whole population/swarm, dynamical environments, and/or neighbors has been extracted

Manuscript received September 20, 2017; revised November 30, 2017; accepted December 3, 2017. Date of publication December 22, 2017; date of current version January 15, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61503165, Grant 61673025, Grant 61375119, and Grant 61673196, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20150239, in part by the Beijing Natural Science Foundation under Grant 4162029, and in part by the National Key Basic Research Development Plan (973 Plan) Project of China under Grant 2015CB352302. This paper was recommended by Y. S. Ong. *(Corresponding author: Ying Tan.)*

G.-G. Wang is with the Department of Computer Science and Technology, Ocean University of China, Qingdao 266100, China, also with the School of Computer, Jiangsu Normal University, Xuzhou 221116, China, and also with the Institute of Algorithm and Big Data Analysis and School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China (e-mail: gaigewang@gmail.com).

Y. Tan is with the Key Laboratory of Machine Perception and the Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: ytan@pku.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2017.2780274

and reused to a certain degree. Few of these improvements were based on a fitness function, with the exception of Bingul [52]. Bingul transformed the multiobjective problem (MOP) into single-objective problems by using a fitness function. In addition, the square-based fitness function was used in Bingul [52]. In contrast, most of the previous studies aimed to improve the performance of a particular metaheuristic algorithm by reusing the exploited information. However, they failed to form a general framework for reusing the obtained information.

In this paper, we present our research, based on a fitness function, in which we constructed a systematic information feedback model that reused the information from individuals in previous iterations. This proposed information feedback model was demonstrated to provide a general framework that could be used to improve the performance of most metaheuristic algorithms.

In this paper, we studied how to make the best use of the information available from previous individuals by using the following techniques. First, a certain number of individuals in previous iterations were selected in either a fixed or random manner. For this paper, we selected one, two, or three individuals from previous iterations. Second, the previous individuals selected as feedback information were given to the updating process. In this way, the information from previous individuals could be reused fully. Last, each individual of the current iteration was updated according to the individual generated by the basic algorithm and some selected previous individuals through a weighted sum method. It should be noted that there were many different ways to determine their weights. This paper used their fitness to do so. An individual with better fitness had a greater weight.

Combining information feedback models with metaheuristic algorithms led to improved methods. They were then benchmarked through 14 test cases and ten CEC 2011 real world problems. The experimental results demonstrated that the information feedback from previous individuals significantly outperformed all the basic algorithms.

The organization of this paper is as follows. Section II provides a review of the related literature regarding reusing information in metaheuristic algorithms. In Section III, we introduce the optimization process for metaheuristic algorithms. This section then explains how we incorporated the useful information in previous individuals into the basic methods, and demonstrates how to improve PSO with information feedback models. Section IV provides the mathematical analyses. In Section V, we explore various experimental models and provide the simulation results. Further discussion is given in Section VI. Section VII concludes this paper.

II. Related Work

Recently, in order to improve the performance of the metaheuristic algorithms, many scholars have extracted and reused the information from various sources, such as the surrogate, an individual, the whole population/swarm, and/or a neighbor. They have also used information from dynamical environments, directional information, mutual information (MI), and other forms of information. Their work regarding various types of information reuse is reviewed briefly below.

A. Surrogate Information

Surrogate information is found to be very effective in reducing user effort. Therefore, many researchers have improved various metaheuristic algorithms through the use of surrogate information, as in GA and PSO.

Sun *et al.* [31] proposed a new surrogate-assisted interactive genetic algorithm (IGA), where the uncertainty in subjective fitness evaluations was exploited both in training the surrogates and in managing surrogates. Moreover, uncertainty in the interval-based fitness values was also considered in model management, so that not only the best individuals but also the most uncertain individuals would be chosen to be re-evaluated by the human user. The experimental results indicated that the new surrogate-assisted IGA could alleviate user fatigue effectively and was more likely to find acceptable solutions in solving complex design problems.

Gong *et al.* [53] proposed a computationally cheap surrogate model-based multioperator search strategy for evolutionary optimization. In this strategy, a set of candidate offspring solutions were generated by using the multiple offspring reproduction operators. The best one according to the surrogate model was chosen as the offspring solution. The proposed strategy was used to implement a multioperator ensemble in two popular evolutionary algorithms (EAs), DE, and PSO.

Aiming to solve medium-scale problems (i.e., 20–50 decision variables), Liu *et al.* [54] proposed a Gaussian process surrogate model-assisted EA for medium-scale computationally expensive optimization problems (GPEME). A new framework was developed and used in GPEME that carefully coordinated the surrogate modeling and the evolutionary search. In this way, the search could focus on a small promising area and was supported by the constructed surrogate model. Sammon mapping was also introduced to transform the decision variables from tens of dimensions to a few dimensions, in order to take advantage of Gaussian process surrogate modeling in a low-dimensional space.

Wang *et al.* [55] divided data-driven optimization problems into two categories: 1) offline and 2) online data-driven optimization. An EA was then presented to optimize the design of a trauma system, which is a typical offline data-driven multiobjective optimization problem. As each single function evaluation involved a large amount of patient data, Wang *et al.* [55] developed a multifidelity surrogate management strategy to reduce the computation time of the evolutionary optimization.

Mendes *et al.* [56] proposed the use of genetic programming to obtain high-quality surrogate functions that were evaluated quickly. Such functions could be used to compute the values of the optimization functions in place of the burdensome methods. The proposal was tested successfully on a version of the TEAM 22 benchmark problem with uncertainties in decision parameters.

Kattan and Ong [57] proposed a surrogate genetic programming (or sGP for short) to retain the appeal of the semantic-based evolutionary search for handling challenging problems with enhanced efficiency. The proposed sGP divided the population into two parts, then it evolved the population using standard GP search operators and meta-models that served as a surrogate to the original objective function evaluation. In contrast to previous works, two forms of meta-models were introduced in this paper to make the idea of using a surrogate in GP search feasible and successful.

Rosales-Pérez *et al.* [58] introduced an approach for addressing model selection for support vector machines used in classification tasks. The model selection problem was transferred mathematically as a multiobjective one, aiming to minimize simultaneously two components closely related to the error of a model. A surrogate-assisted evolutionary multiobjective optimization approach was adopted to explore the hyper-parameters space. The surrogate-assisted optimization was used to reduce the number of solutions evaluated by the fitness functions so that the computational cost would be reduced as well.

Hildebrandt and Branke [59] presented a new way to use surrogate models with GP. Rather than using the genotype directly as input to the surrogate model, they used a phenotypic characterization in their method. This phenotypic characterization could be computed efficiently, which allowed them to define approximate measures of equivalence and similarity. Using a stochastic, dynamic job shop scenario as an example of simulation-based GP with an expensive fitness evaluation, they demonstrated that these ideas can be used to construct surrogate models and improve the convergence speed and solution quality of GP.

PSO is one of the most excellent swarm intelligence-based metaheuristic algorithms [39], in which particles are updated according to the best individuals in the population and the best position for each particle so far. Lin *et al.* [60] proposed a binary PSO based on surrogate information with proportional acceleration coefficients (BPSOSIPAC) for the 0-1 multidimensional knapsack problem (MKP). The BPSOSIPAC was based on the surrogate information concept to repair an infeasible particle and make the infeasible solution become a feasible one.

B. Individual Information

ABC is a relatively new swarm intelligence-based metaheuristic algorithm [8]. In the basic ABC, previous individuals were not reused at all. In addition, Gao *et al.* [61] proposed a bare bones ABC called BABC that used parameter adaptation and fitness-based neighborhood. In BABC, the useful information in the best individual and a Gaussian search equation were used to generate a new candidate individual at the onlooker phase [61]. On other hand, at the employed bee phase, the information from the previous search and from the better individuals was incorporated into the parameter adaptation strategy and a fitness-based neighborhood mechanism in order to improve the search ability [61].

GA has been applied successfully to address all kinds of engineering problems, especially in discrete optimization [30], [31]. Bingul [52] first used information feedback in adaptive GAs for dynamic MOPs. Bingul transformed the multiobjective optimization problem into a single-objective problem by using a static fitness function and rule-based weight fitness function. Bingul [52] also used a square-based fitness function because it generated the best solutions among various types of fitness functions.

Gong *et al.* [62] combined the advantages of the GA and PSO, and proposed a generalized "learning PSO" paradigm, the *L-PSO. In *L-PSO, genetic operators were used to generate exemplars according to the historical search information of particles. By performing crossover, mutation, and selection on the historical information of particles, the constructed exemplars were not only well diversified but also highly qualified.

Ly and Lipson [63] proposed a strategy to select the most informative individuals in a teacher-learner type coevolution by using the surprisal of the mean, based on Shannon information theory. This selection strategy was verified by an iterative coevolutionary framework, which consisted of symbolic regression for model inference, and a GA for optimal experiment design.

In order to exploit fully both global statistical information and individual location information, Zhou *et al.* [64] combined an estimation of distribution algorithm with computationally cheap and expensive local search (LS) methods.

Xiong *et al.* [65] combined stochastic elements into a resource investment project scheduling problem (RIPSP), and proposed a stochastic extended RIPSPs. A knowledge-based multiobjective EA (K-MOEA) was proposed to solve the problem. In K-MOEA, the useful information in the obtained nondominated solutions (individuals) was extracted and then used to update the population periodically to guide subsequent search.

C. Population/Swarm Information

Gao *et al.* [66] proposed a novel ABC algorithm based on information learning, called ILABC. In ILABC, at each generation, the whole population was divided dynamically into several subpopulations by the clustering partition based on the previous search experience. Furthermore, the different individuals in one subpopulation and in different subpopulations exchanged information after all the individuals were updated. In this way, all the individuals would find the best solution cooperatively. In addition to ILABC, Gao *et al.* [67] proposed another improved ABC algorithm using more information-based search equations.

Inspired by the echo location behavior of bats in nature, BA was proposed for global optimization problems [47]. The position of the bats was updated by the bats' frequency, velocity, and distance to food. Therefore, their position had no relationship with any kind of information reuse. Wang *et al.* [68] proposed a multiswarm BA (MBA) for global optimization problems. In MBA, the information between different swarms was exchanged by an immigration operator with different parameter settings. Thus, this configuration was able to make a good tradeoff between global and LS.

With regard to DE, it is well accepted that two control parameters: 1) scale factor (F) and 2) crossover rate (Cr), have great influence on the performance of DE. Based on information from the population, Ghosh *et al.* [69] proposed a simple yet useful adaptation technique for tuning F and Cr.

In order to boost the population diversity when addressing large-scale global problems, Ali *et al.* [70] proposed a new, improved DE called *m*DE-*b*ES. This version was a multipopulation algorithm, and the population was divided into independent subgroups, each with different mutation and update strategies. The information of the best individual was used to generate a novel mutation strategy that produced quality solutions with a balance between exploration and exploitation. At each generation, the individuals exchanged their information between the subgroups.

Cui *et al.* [71] designed a novel adaptive multiple subpopulations-based DE named MPDE, in which the parent population was split into three subpopulations based on their fitness values. In MPDE, the useful information from the trial vectors and target vectors was exploited fully to form a replacement strategy that aimed to improve the search ability.

Inspired by team cooperation in the real world, Gao *et al.* [72] proposed a dual-population DE (DPDE) with coevolution for constrained optimization problems (COPs). The COP was divided into two objectives that were solved by two subpopulations at each generation, respectively. In DPDE, an information-sharing strategy was used to exchange search information between the different subpopulations.

Wang *et al.* [73] proposed a cooperative multiobjective DE (CMODE) with multiple populations for multiobjective optimization problems (MOPs), which included M single-objective optimization subpopulations and an archive population for an M-objective optimization problem. These $(M + 1)$ populations cooperated to optimize all objectives of MOPs by using adaptive DEs. The additional difference term was added to the proposed method with the aim of sharing information from the archive. In this way, an individual could use the search information not only from its own subpopulation but also from other populations. The individual was expected to search along the whole Pareto front (PF) by using the information of all the populations instead of being attracted to the margin or extreme point only by the search information of its own subpopulation. Hence, CMODE could approximate the whole PF quickly with the help of the archived information.

Dhal *et al.* [74] proposed two variants of FA: 1) FA via Lévy flights and 2) FA via chaotic sequence. In these two algorithms, the information of population diversity was fully extracted to generate the individuals at each generation.

Pan *et al.* [75] proposed a local-best HS algorithm with dynamic subpopulations (DLHS) for global optimization problems. In DLHS, the whole harmony memory (HM) was divided into a certain number of small-sized sub-HMs that exchanged information with each other by using a periodic regrouping schedule. Furthermore, the useful information in the local best harmony vector was used to generate a novel harmony improvisation scheme [75].

D. Information From Dynamical Environments

Though many versions of multiobjective PSO (MOPSO) have been designed, few MOPSOs have been designed to adjust the balance between exploration and exploitation dynamically according to the feedback information detected from the evolutionary environment. Hu and Yen [76] proposed a new algorithm, the parallel cell coordinate system (PCCS), according to the information about the evolutionary environment, including density, rank, and diversity indicators. PCCS was then incorporated into a self-adaptive MOPSO, and a new MOPSO was proposed: the pccsAMOPSO.

Foss investigated how a viable system, the honey bee swarm, gathered meaningful information about potential new nest sites in its problematic environment [77]. This investigation used a cybernetic model of a self-organizing information network to analyze the findings from the last 60 years of published research about swarm behavior. Information gathering by a honey bee swarm was first modeled as a self-organizing information network.

E. Neighborhood and Direction Information

In the basic DE, the base and difference vectors are always selected randomly from the whole population for the mutation operators, but the neighborhood and direction information fails to be used effectively [10], [11], [78], [79]. In order to address this problem, several scholars have put forward improved strategies.

Peng *et al.* [80] proposed a novel DE framework with distributed direction information-based mutation operators (DE-DDI) for dealing with complex problems in big data. In DE-DDI, the distributed topology was used to generate a neighborhood for each individual first. Then the direction information derived from the neighbors was introduced into the mutation operator of DE. Consequently, the neighborhood and direction information fully exploited the regions of better individuals, and guided the search to the promising area.

Liao *et al.* [81] proposed another DE framework with a directional mutation based on cellular topology, called cellular direction information-based DE (DE-CDI). For each individual in DE-CDI, the cellular topology was formed to define a neighborhood. Next, the direction information based on the neighborhood was incorporated into the mutation operator. In this way, DE-CDI not only extracted the neighborhood information to exploit the regions of better individuals and accelerate convergence but also introduced the direction information to guide the search to the promising area.

In order to use the neighborhood and direction information fully, Cai *et al.* [82] proposed a new DE framework with neighborhood and direction information (NDi-DE). Though NDi-DE had better performance than most of the DEs, its performance relied mainly on the selection of direction information. To overcome this disadvantage, the adaptive operator selection mechanism was incorporated into the NDi-DE, which was able to select adaptively the direction information for the specific DE mutation strategy. Accordingly, an improved NDi-DE called adaptive direction information-based NDi-DE (aNDi-DE) was proposed by Cai *et al.* [82], which performed much better than NDi-DE.

Fang *et al.* [83] proposed a decentralized quantum-inspired PSO (QPSO) with cellular structured population called cQPSO. In cQPSO, the particles were located in a 2-D grid and allowed to get information only from their neighbors. The overlapping particles exchange the information among the nearest neighborhoods.

Wang *et al.* [84] proposed an improved version of BA namely variable neighborhood bat algorithm (VNBA), is thus proposed. In VNBA, the bat individual can get useful information from their neighbors.

F. Mutual Information

He *et al.* [85] introduced the multiresolution analysis, MI, and PSO into artificial neural network models. They proposed a hybrid wavelet neural network model for forecasting monthly rainfall from antecedent monthly rainfall and climate indices.

G. Other Information

ACO is one of the most representative metaheuristic algorithms for global optimization problems, especially, for discrete optimization [6], [49]. Because the ants are updated according to the pheromone, the previous information fails to be used in ACO.

Shang *et al.* [86] introduced heuristic information into ant-decision rules, and then proposed a new version of ACO named AntMiner for epistasis detection. In AntMiner, the heuristic information was used to guide ants during the search process with the aim of enhancing the computational efficiency and solution accuracy.

Wang and Tang [87] proposed an adaptive DE based on analysis of search data for the MOPs. In this algorithm, first the useful information was derived from the search data during the evolution process by using clustering and statistical methods. Then the derived information was used to guide the generation of new population and the LS.

Park and Lee [88] proposed a novel opposition-based learning method by using a beta distribution with partial dimensional change and selection switching. They combined this approach with DE to enhance the convergence speed and search ability. In the proposed method, the partial dimensional changing scheme was used to preserve useful information.

Simulated annealing (SA) is one of the oldest classical metaheuristic algorithms [89] that is a trajectory-based optimization algorithm. Yang and Kumar [90] proposed an information guided framework for SA. Information gathered from the exploration stage was used as feedback to drive the optimization procedure, leading to the rise of the annealing temperature during the optimization process. The resulting algorithm had two phases: phase I performed nearly unrestricted exploration, and phase II "re-heated" the annealing procedure and exploited information gathered during phase I.

Muñoz *et al.* [91] proposed a robust information content-based method for continuous fitness landscapes that generated four measures related to the landscape features. In addition, it could overcome the disadvantage of sampling the fitness landscape using random walks with variable step size.

From the descriptions above, we can see that for most metaheuristic algorithms, some useful information obtained from a surrogate, an individual, the whole population/swarm, dynamical environments, neighbor and direction, and/or mutual relationship is extracted and reused to a certain degree. However, few of them are based on a fitness function (except [52]). Bingul [52] transferred the MOP into some single-objective problems by using a fitness function as explained previously. Furthermore, while most of the studies above aimed to improve the performance of a certain metaheuristic algorithm by reusing the exploited information, they failed to form a general framework for reusing the obtained information.

In this paper, we present our research, based on a fitness function, in which we constructed a systematic information feedback model that reused the information from individuals in previous iterations. This proposed information feedback model was demonstrated to provide a general framework that could be used to improve the performance of most metaheuristic algorithms.

III. Improving Metaheuristic Algorithms With Information Feedback Models

In this section, we explain how metaheuristic algorithms have been improved based on information feedback models. First, we provide a brief outline of the basic optimization process, and then we give a description of the information feedback models. Finally, using PSO as an example, we demonstrate how to improve the algorithm using information feedback models.

A. Optimization Process

Despite the fact that different metaheuristic algorithms have different updating strategies, their optimization processes can be summarized briefly by the following general steps.

1) Initialization: Initialization can be divided into population initialization and parameter initialization. The running environments for later search are set during this process.

2) Search: In general, metaheuristic algorithms first implement global search and then LS, i.e., exploration and then exploitation. These two searches perform in parallel, being adjusted by certain parameters. The search process is repeated until some termination condition is satisfied.

3) Output: Output the final best solutions.

B. Information Feedback Models

In theory, for our model $k (k \geq 1)$ previous individuals can be selected, but using a substantial number of individuals might complicate the method. Therefore, in this paper, $k \in \{1, 2, 3\}$. As mentioned above, we will take PSO as an example to illustrate the framework of our proposed method. Some symbols are given before the information feedback models are described.

Suppose that x_i^t is the ith individual at iteration t, and x_i and f_i^t are its position and fitness value, respectively. Here,

t is the current iteration, $1 \leq i \leq N_P$ is an integer number, and N_P is the population size. y_i^{t+1} is the individual generated by the basic PSO, and f_i^{t+1} is its fitness. The framework of the proposed method is given through the individuals at the $(t-2)$th, $(t-1)$th, tth, and $(t+1)$th iterations.

1) Model F1 and Model R1: This is the simplest case. The ith individual x_i^{t+1} can be generated as follows:

$$x_i^{t+1} = \alpha y_i^{t+1} + \beta x_j^t \tag{1}$$

where x_j^t is the position for individual $j (j \in \{1, 2, \ldots, N_P\})$ at iteration t, and f_j^t is its fitness.

α and β are weighting factors satisfying $\alpha + \beta = 1$. They can be given as

$$\alpha = \frac{f_j^t}{f_i^{t+1} + f_j^t}, \beta = \frac{f_i^{t+1}}{f_i^{t+1} + f_j^t}. \tag{2}$$

Here, individual j can be determined in the following ways.

Definition 1: The model in (1) is called model $F1$ when $j = i$.

The individuals in previous and current generations are used to generate the individual for the next generation.

Definition 2: The model in (1) is called model $R1$ when $j = r_1$, where r_1 is an integer randomly selected between 1 and N_P.

The individual generated by Definition 2 has a higher population diversity than the one generated by Definition 1. We can see that if $r_1 = i$, the model R1 will be F1 with the probability of $1/N_P$. Their incorporation into the basic PSO results in PSOF1 and PSOR1, respectively.

2) Model F2 and Model R2: Two individuals at two previous iterations are collected and used to generate individual i. For this case, the ith individual x_i^{t+1} can be generated as follows:

$$x_i^{t+1} = \alpha y_i^{t+1} + \beta_1 x_{j_1}^t + \beta_2 x_{j_2}^{t-1} \tag{3}$$

where $x_{j_1}^t$ and $x_{j_2}^{t-1}$ are the position for individuals j_1 and $j_2 (j_1, j_2 \in \{1, 2, \ldots, N_P\})$ at iteration t and $t-1$, and $f_{j_1}^t$ and $f_{j_2}^{t-1}$ are their fitness values, respectively.

α, β_1, and β_2 are weighting factors satisfying $\alpha + \beta_1 + \beta_2 = 1$. They can be provided as follows:

$$\begin{aligned}
\alpha &= \frac{1}{2} \bullet \frac{f_{j_2}^{t-1} + f_{j_1}^t}{f_i^{t+1} + f_{t-1} + f_{j_1}^t} \\
\beta_1 &= \frac{1}{2} \bullet \frac{f_{j_2}^{t-1} + f_i^{t+1}}{f_i^{t+1} + f_{j_2}^{t-1} + f_{j_1}^t} \\
\beta_2 &= \frac{1}{2} \bullet \frac{f_i^{t+1} + f_{j_1}^t}{f_i^{t+1} + f_{j_2}^{t-1} + f_{j_1}^t}.
\end{aligned} \tag{4}$$

Individuals j_1 and j_2 in (3) can be determined in several different ways. For this model, this paper focused on Definitions 3 and 4.

Definition 3: The model in (3) is called model $F2$ when $j_1 = j_2 = i$.

The individuals at two previous and current generations are used to generate the individual for the next generation.

Definition 4: The model in (3) is called model $R2$ when $j_1 = r_1$, and $j_2 = r_2$, where r_1 and r_2 are integers that are randomly selected between 1 and N_P.

Similarly, the individual generated by Definition 4 has more diversity of population than the individual generated by Definition 3. Here, we can see, if $r_1 = r_2 = i$, the model R2 will be F2 with the probability of $1/N_P$. Their incorporation into the basic PSO results in PSOF2 and PSOR2, respectively.

3) Model F3 and Model R3: Three individuals at three previous iterations are collected and used to generate individual i. For this case, the ith individual x_i^{t+1} can be generated as follows:

$$x_i^{t+1} = \alpha y_i^{t+1} + \beta_1 x_{j_1}^t + \beta_2 x_{j_2}^{t-1} + \beta_3 x_{j_3}^{t-2} \tag{5}$$

where $x_{j_1}^t$, $x_{j_2}^{t-1}$, and $x_{j_3}^{t-2}$ are the position of individuals j_1, j_2, and $j_3 (j_1, j_2, j_3 \in \{1, 2, \ldots, N_P\})$ at iteration t, $t-1$, and $t-2$, and $f_{j_1}^t, f_{j_2}^{t-1}$, and $f_{j_3}^{t-2}$ are their fitness values, respectively.

Their weighting factors are α, β_1, β_2, and β_3 with $\alpha + \beta_1 + \beta_2 + \beta_3 = 1$, which can be given as

$$\begin{aligned}
\alpha &= \frac{1}{3} \bullet \frac{f_{j_1}^t + f_{j_2}^{t-1} + f_{j_3}^{t-2}}{f_i^{t+1} + f_{j_1}^t + f_{j_2}^{t-1} + f_{j_3}^{t-2}} \\
\beta_1 &= \frac{1}{3} \bullet \frac{f_i^{t+1} + f_{j_2}^{t-1} + f_{j_3}^{t-2}}{f_i^{t+1} + f_{j_1}^t + f_{j_2}^{t-1} + f_{j_3}^{t-2}} \\
\beta_2 &= \frac{1}{3} \bullet \frac{f_i^{t+1} + f_{j_1}^t + f_{j_3}^{t-2}}{f_i^{t+1} + f_{j_1}^t + f_{j_2}^{t-1} + f_{j_3}^{t-2}} \\
\beta_3 &= \frac{1}{3} \bullet \frac{f_i^{t+1} + f_{j_1}^t + f_{j_2}^{t-1}}{f_i^{t+1} + f_{j_1}^t + f_{j_2}^{t-1} + f_{j_3}^{t-2}}.
\end{aligned} \tag{6}$$

Though $j_1 - j_3$ can be determined in many different ways, we adopted Definitions 5 and 6 for this model.

Definition 5: The model in (5) is called model $F3$ when $j_1 = j_2 = j_3 = i$.

The individuals at two previous and current generations are used to generate the individual for the next generation.

Definition 6: The model in (5) is called model $R3$ when $j_1 = r_1, j_2 = r_2$, and $j_3 = r_3$, where $r_1 - r_3$ are integer numbers that are selected randomly between 1 and N_P.

Similarly, the individual generated by Definition 6 has more population diversity. Here, we can see that if $r_1 = r_2 = r_3 = i$, model R3 will be F3 with the probability of $1/N_P$. Their incorporation into the basic PSO results in PSOF3and PSOR3, respectively.

By incorporating the information feedback model into the basic optimization process, we have a new updating optimization process as shown in Fig. 1.

C. PSO Using Model F1

We now take PSO and model F1 as an example to explain how to introduce information feedback into a metaheuristic algorithm.

PSO [39] is one of the most representative swarm intelligence paradigms. The solutions (called particles) are located initially in the whole search region at random. Subsequently,

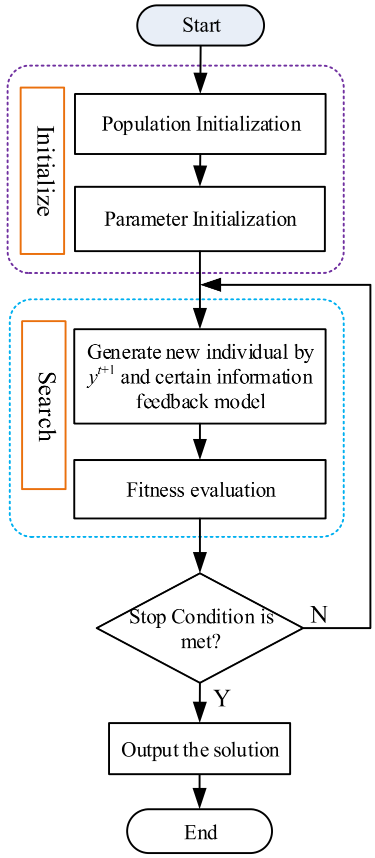

Fig. 1. Schematic flowchart of updating optimization process.

Improving PSO with information feedback models (k = 1)

Begin

Step 1: Initialization. The velocity and position of all particles are set randomly to within pre-defined ranges.

Step 2: Evaluate the fitness for each particle determined by objective function $f(x)$.

Step 3: ***while*** $t<G_{max}$ ***do***

for $i = 1{:}N_P$ (all N_P particles) ***do***

Update the velocity according to equation (7);

Update the position according to equation (8);

The newly-generated particle is called y_i;

Check y_i and limit it to the allowed range;

if $t > 1$ ***then***

Update the particle according to equation (1).

end if

Update $p_{i,best}$ and $g_{i,best}$.

end for i

end while

Step 4: Return the values *of* g_{best} *and* $f(g_{best})$ as its solution.

End.

Fig. 2. Improving PSO with information feedback models ($k = 1$).

the velocity and position of the particles are updated as (7) and (8), respectively.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 \left(p_{i,\text{best}} - x_i\right) + c_2 r_2 \left(g_{i,\text{best}} - x_i\right) \quad (7)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (8)$$

where x_i and v_i are the position and velocity of particle i, respectively; $p_{i,\text{best}}$ and $g_{i,\text{best}}$ are the position with the optimal objective value searched until now by particle i and the whole population, respectively; w is an inertia parameter controlling the dynamics of flying; r_1 and r_2 are random real numbers in [0, 1]; and c_1 and c_2 are factors controlling the related weighting of corresponding terms. After updating velocity and position for particle i, $p_{i,\text{best}}$ and $g_{i,\text{best}}$ will be updated. This process will be repeated until a certain stop condition is met.

Next, looking at the general outline of the optimization process, we can see the main steps for improving PSO by using the information feedback model ($k = 1$).

1) Initialization: The parameters used in PSO are set, and the particle population is initialized randomly with the predefined regions. This process is the same as performed in the basic PSO.

2) Search: This is the critical part for improving PSO. First, the velocity and position of particle i are updated according to (7) and (8). The updated particle can be called y_i. If the generation count t is bigger than 1, particle i will be further updated by (1), and the newly generated particle will be considered as the final particle for the next generation. The search process is repeated until some termination condition is satisfied.

3) Output: PSO returns the values of g_{best} and $f(g_{\text{best}})$ as its final solution.

The detailed steps of the combination of PSO and the information feedback model ($k = 1$) can be seen in Fig. 2. In Fig. 2, $G_{\max}$ is the maximum of the generation.

Similarly, the other five models (R1–3, F2–3) can be incorporated into the basic PSO. Given the limits on the length of this paper, we will not describe them in detail.

IV. Mathematical Analyses

In this section, we provide a mathematical analysis to prove the convergence of the proposed method. We first prove the algorithm under model F3 and R3. Here, the following lemmas are provided, and they are true for any algorithm discussed in this paper.

Lemma 1: An algorithm A can reach its final solution x_{best} all of the time.

Here, algorithm A can be any of the algorithms discussed in this paper, such as ACO [6], BA [47], BBO [29], CS [14], DE [10], ES [13], KH [21], MBO [36], PBIL [37], and PSO [39].

x_{best} is the best solution for algorithm A, and its lower bound and upper bound are $x_{\min}$ and $x_{\max}$, respectively. Lemma 1 indicates that algorithm A is able to find the final solution all of the time, if it can search for the given domain with enough time.

TABLE I
BENCHMARK FUNCTIONS

No.	*Function*	*lb*[a]	*ub*[a]	*opt*	*Sep*[b]	*M*[c]
F01	Ackley	−32.768	32.768	0	S	MM
F02	Fletcher−Powell	−π	π	0	NS	MM
F03	Griewank	−600	600	0	NS	MM
F04	Penalty #1	−50	50	0	NS	MM
F05	Penalty #2	−50	50	0	NS	MM
F06	Quartic	−1.28	1.28	1	NS	MM
F07	Rastrigin	−5.12	5.12	0	NS	MM
F08	Rosenbrock	−2.048	2.048	0	NS	UM
F09	Schwefel 2.26	−512	512	0	NS	MM
F10	Schwefel 1.2	−100	100	0	NS	UM
F11	Schwefel 2.22	−10	10	0	NS	UM
F12	Schwefel 2.21	−100	100	0	NS	UM
F13	Sphere	−5.12	5.12	0	NS	UM
F14	Step	−5.12	5.12	0	NS	UM

[a]*lb* denotes lower bound, *ub* denotes upper bound, *opt* denotes optimum point.
[b]*Sep*, S, and NS represent Separability, Separable, and Nonseparable, respectively.
[c]*M*, UM, and MM represent Modality, Unimodal, and Multimodal, respectively.

Lemma 2: The solution x_i^{t+1} is one of the feasible solutions for algorithm A.

Proof: Here, we only prove that the lower bound and upper bound of x_i^{t+1} for algorithm A are $x_{\min}$ and $x_{\max}$, respectively. For ease of description, (5) can be described in the following form:

$$x_i^{t+1} = \alpha y_i + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3. \quad (9)$$

It is clear that for algorithm A, the lower bound and upper bound of the solutions x_1, x_2, and x_3 are $x_{\min}$ and $x_{\max}$, respectively. In other words, $x_{\min} \leq y_i \leq x_{\max}, x_{\min} \leq x_1 \leq x_{\max}, x_{\min} \leq x_2 \leq x_{\max}$, and $x_{\min} \leq x_3 \leq x_{\max}$.

Next, we can get $\alpha \times x_{\min} \leq \alpha \times y_i \leq \alpha \times x_{\max}$, $\beta_1 \times x_{\min} \leq \beta_1 \times x_1 \leq \beta_1 \times x_{\max}$, $\beta_2 \times x_{\min} \leq \beta_2 \times x_2 \leq \beta_2 \times x_{\max}$, and $\beta_3 \times x_{\min} \leq \beta_3 \times x_3 \leq \beta_3 \times x_{\max}$. Therefore, we get

$$\begin{aligned}&(\alpha + \beta_1 + \beta_2 + \beta_3) \times x_{\min}\\ &\quad \leq x_i^{t+1} = \alpha y_i + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3\\ &\quad \leq (\alpha + \beta_1 + \beta_2 + \beta_3) \times x_{\max}.\end{aligned} \quad (10)$$

According to the definition of α, β_1, β_2, and β_3 in (5), we know $\alpha + \beta_1 + \beta_2 + \beta_3 = 1$. Therefore, (10) can be updated as

$$x_{\min} \leq x_i^{t+1} = \alpha y_i + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \leq x_{\max}. \quad (11)$$

We observe clearly that $x_{\min} \leq x_i^{t+1} \leq x_{\max}$. In other words, the newly generated solution x_i^{t+1} via our proposed method is a feasible solution for algorithm A. ■

Theorem 3: A proposed algorithm A′ can reach its final solution x'_{best} all the time.

Proof: Here, A′ represents the proposed algorithm discussed in the previous section. Therefore, according to Lemmas 1 and 2, the proposed algorithm A′ is able to find the final solution x'_{best} if it can search for the given domain with enough time.

For Models F1–2 and R1–2, it is obvious that these models are special cases of Models F3 and R3. Therefore, any proposed algorithm A′ is similarly proven. We do not give them in detail in this paper.

In sum, for each information feedback model, where the model is Model F1–F3 or R1–R3, an algorithm A under these models can reach its final solution x'_{best} every time. ■

V. SIMULATION RESULTS

Section III gives six information feedback models, i.e., F1–F3 and R1–R3, each of which can be incorporated into a basic metaheuristic algorithm, thereby, yielding six variants of each basic method. For example, given PSO, we have PSOF1-3 and PSOR1-3. The basic PSO can be named as PSOF0. Simply, we can call them F0–F3, and R1–R3 for short.

We must point out that in order to investigate fully the superiority of different information feedback models, six variants were compared with each other only and with the corresponding basic algorithm. Through this comparison, we were able to look at the performance of six information feedback models and determine whether these models could improve the performance of the basic algorithm.

Six information feedback models were combined with the basic metaheuristic algorithms, and these newly combined methods were further benchmarked by 14 standard test functions as shown in Table I [29]. Each function had 20 independent variables, that is, the dimension of each problem was 20. Some of functions were multimodal, which means that they had multiple local minima. Some were nonseparable, which means that they could not be written as a sum of functions of individual variables.

The benchmarks were compared by implementing the integer versions of all the metaheuristic algorithms in MATLAB [29]. The granularity or precision of each benchmark function was 0.1, except for the Quartic function. Since the domain of each dimension of the Quartic function was only ±1.28, it was implemented with a granularity of 0.01 [29]. More information about these functions can be seen by referring to [29].

First, we investigated the performance of PSO under Models F1–F3 and R1–R3, and then these six models were extended to be incorporated into more metaheuristic algorithms.

A. Performance of PSO With Models F1–F3 and R1–R3

In this section, we will look at the performance of PSO under Models F1–F3 and R1–R3 on 14 benchmarks in Table I.

In order to get their representative statistical results, 50 independent runs were done for PSO. In addition, PSO had a population size of 50, an elitism parameter of 2, and was run for 50 generations. The results were recorded in Table II.

In more detail, the *best*, *average*, and *worst* performances of each method were collected, as shown in Table II. The results were highlighted in bold if PSO performed the best on a benchmark. The total numbers of the bold results were collected, as shown in the last row in Table II. In order to investigate the influence of F1–3 and R1–3, the number of functions on which PSO performed the best was calculated, as shown in the last two columns of Table II.

From Table II, we see that R1 was the best information feedback model, having the greatest impact on PSO. F3 was inferior only to R1. In addition, for six information feedback

TABLE II
FUNCTION FITNESS OBTAINED BY PSO WITH SIX MODELS

		F0	F1	R1	F2	R2	F3	R3	F1~3	R1~3
F01	B[a]	14.24	9.02	8.92	6.47	7.08	**5.52**	5.80	1	0
	M[b]	15.89	10.35	10.87	7.66	7.97	**6.55**	6.88	1	0
	W[c]	16.99	11.51	11.91	8.63	9.21	**7.78**	8.09	1	0
F02	B[a]	3.63E5	**2.03E5**	2.11E5	2.98E5	2.96E5	2.12E5	2.49E5	1	0
	M[b]	5.75E5	4.67E5	**4.62E5**	6.20E5	6.42E5	6.45E5	6.34E5	0	1
	W[c]	1.06E6	1.00E6	**1.03E6**	1.13E6	1.10E6	1.10E6	1.01E6	0	1
F03	B[a]	45.26	3.88	1.72	1.97	1.74	2.05	**1.52**	0	1
	M[b]	82.71	6.39	**2.55**	3.47	3.14	3.62	3.59	0	1
	W[c]	111.90	11.53	**3.95**	7.25	6.00	7.33	7.77	0	1
F04	B[a]	2.08E5	3.91E4	18.42	**6.07**	8.54	6.43	7.89	1	0
	M[b]	7.94E6	3.54E5	1.94E5	2.20E3	7.51E3	**524.90**	698.30	1	0
	W[c]	3.52E7	6.00E5	4.80E5	4.08E4	1.36E5	2.33E4	**9.61E3**	0	1
F05	B[a]	4.13E6	6.70E5	2.88E5	419.70	2.43E4	131.20	**16.60**	0	1
	M[b]	2.88E7	1.45E6	1.07E6	2.08E5	3.94E5	**6.28E4**	1.62E5	1	0
	W[c]	8.05E7	2.22E6	1.74E6	1.04E6	1.64E6	**5.24E5**	1.03E6	1	0
F06	B[a]	0.76	0.15	0.11	0.02	0.05	**0.01**	0.03	1	0
	M[b]	3.16	0.22	0.16	0.13	0.17	**0.09**	0.13	1	0
	W[c]	5.70	0.32	**0.24**	0.32	0.34	0.26	0.29	0	1
F07	B[a]	144.50	87.32	94.42	**66.22**	69.62	70.02	83.06	1	0
	M[b]	164.70	116.90	119.20	106.70	106.30	**101.10**	101.30	1	0
	W[c]	191.20	132.10	133.90	122.80	127.50	118.80	**117.20**	0	1
F08	B[a]	331.70	52.57	**39.08**	51.58	47.80	44.81	49.17	0	1
	M[b]	593.60	75.78	**53.36**	73.13	67.76	69.09	73.03	0	1
	W[c]	977.80	105.50	**87.96**	107.20	95.88	102.20	94.55	0	1
F09	B[a]	**3.75E3**	5.26E3	5.18E3	5.16E3	5.15E3	4.57E3	5.33E3	0	0
	M[b]	**5.17E3**	6.05E3	5.95E3	6.14E3	6.01E3	6.01E3	5.98E3	0	0
	W[c]	**6.04E3**	6.71E3	6.38E3	6.72E3	6.60E3	6.70E3	6.64E3	0	0
F10	B[a]	3.96E3	508.10	355.00	358.60	406.20	**317.60**	429.70	1	0
	M[b]	8.30E3	727.80	**551.10**	736.00	771.50	747.10	856.40	0	1
	W[c]	1.26E4	1.14E3	**798.90**	1.27E3	1.32E3	1.17E3	1.49E3	0	1
F11	B[a]	27.67	9.70	6.60	4.58	4.79	**3.61**	5.07	1	0
	M[b]	47.66	15.26	14.66	**7.95**	8.09	8.43	9.28	1	0
	W[c]	106.50	19.59	26.85	**12.33**	13.84	14.90	17.14	0	1
F12	B[a]	37.45	10.60	6.97	4.92	4.71	**4.47**	4.58	1	0
	M[b]	51.54	13.66	10.77	7.41	7.19	7.19	**6.54**	0	1
	W[c]	71.18	17.80	14.59	12.14	12.21	11.83	**9.98**	0	1
F13	B[a]	15.14	0.53	0.36	0.32	0.34	**0.27**	0.37	1	0
	M[b]	23.00	1.53	**0.56**	0.76	0.70	0.90	0.87	0	1
	W[c]	30.06	2.42	**0.80**	1.57	1.20	1.82	1.86	0	1
F14	B[a]	5.95E3	241.00	138.00	104.00	**74.00**	133.00	97.00	0	1
	M[b]	9.17E3	600.70	**227.10**	288.40	255.90	299.90	279.30	0	1
	W[c]	1.12E4	914.00	**436.00**	638.00	587.00	549.00	555.00	1	0
T[d]	B[a]	1	1	1	2	1	6	2	9	4
	M[b]	1	0	6	1	0	5	1	6	7
	W[c]	1	0	7	1	0	2	3	3	10

[a]B is an abbreviation for Best; [b]M is an abbreviation for Mean; [c]W is an abbreviation for Worst; and [d]T is an abbreviation for Total.

models and F0, their average ranking from good to bad was as follows: R1 > F3 > R3 > F2 > F0 > R1 > F1 = R2. Models R1–3 have slightly greater impact than F1–3 for the PSO algorithm on 14 benchmarks (21 versus 18).

From Table II, we can see that our six proposed models, especially R1 and F3, were able to improve significantly the performance of PSO by balancing the exploration and exploitation. Let us give the detailed analyses as follows.

In PSO, particle i learned mainly from the information of the global search and its own best position. On one hand, this situation meant that most particles would fly toward the promising area, and the PSO would have a fast convergence. That is to say, PSO would have a good exploration ability. On other hand,

TABLE III
PARAMETER SETTINGS

Alg	*Parameter settings*
ACO	Initial pheromone value τ_0 = 1E-6, pheromone update constant Q = 20, exploration constant q_0 = 1, global pheromone decay rate ρ_g = 0.9, local pheromone decay rate ρ_l = 0.5, pheromone sensitivity α = 1, and visibility sensitivity β = 5.
BA	Loudness A = 0.5, pulse rate r = 0.5, and scaling factor ε = 0.001.
BBO	Habitat modification probability = 1, immigration probability bounds per gene = [0, 1], step size for numerical integration of probabilities = 1, maximum immigration and migration rates for each island = 1 and mutation probability = 0.005.
CS	A discovery rate p_a = 0.25.
DE	A weighting factor F = 0.5 and a crossover constant CR = 0.5.
ES	The number of offspring λ = 10 produced in each generation, and standard deviation σ = 1 for changing solutions.
KH	The foraging speed V_f = 0.02, the maximum diffusion speed $D^{\max}$ = 0.014, the maximum induced speed $N^{\max}$ = 0.01.
MBO	Max step $S_{\max}$ = 1.0, butterfly adjusting rate BAR = 5/12, migration period $peri$ = 1.2, and the migration ratio p = 5/12.
PBIL	A learning rate of 0.05, 1 good population member and 0 bad population members to use to update the probability vector each generation, an elitism parameter of 1, and a 0 probability vector mutation rate.

[a]*alg* is an abbreviation for algorithm.

TABLE IV
FUNCTION FITNESS OBTAINED BY TEN METAHEURISTIC ALGORITHMS WITH SIX INFORMATION FEEDBACK MODELS

	F0	*F1*	*R1*	*F2*	*R2*	*F3*	*R3*	*F1~3*	*R1~3*
ACO	5	5	8	8	**10**	9	**10**	22	**28**
BA	6	3	2	**23**	1	7	0	**33**	3
BBO	8	**28**	0	5	0	1	0	**34**	0
CS	1	5	0	**27**	0	9	0	**41**	0
DE	2	**21**	0	15	0	3	1	**39**	1
ES	3	0	**17**	14	3	4	1	18	**21**
KH	4	0	**27**	3	5	3	1	6	**33**
MBO	10	4	9	**11**	**11**	1	0	16	**20**
PBIL	3	2	4	1	1	0	**31**	3	**36**
PSO	3	1	**14**	4	1	13	6	18	**21**
Total	45	69	81	**111**	34	50	50	**230**	163

TABLE V
CPU TIME USED BY TEN METAHEURISTIC ALGORITHMS WITH SIX INFORMATION FEEDBACK MODELS

	F0	*F1*	*R1*	*F2*	*R2*	*F3*	*R3*
ACO	**1.00**	1.27	1.28	1.02	1.03	1.01	1.02
BA	1.31	1.02	1.58	**1.00**	1.09	1.03	1.07
BBO	**1.00**	1.14	1.15	1.15	1.17	1.21	1.18
CS	**1.00**	1.19	1.19	1.37	1.41	1.43	1.43
DE	**1.00**	1.24	1.22	1.20	1.21	1.28	1.26
ES	**1.00**	1.37	1.43	1.43	1.42	1.44	1.46
KH	**1.00**	1.18	1.19	1.19	1.14	1.14	1.14
MBO	**1.00**	1.27	1.27	1.26	1.27	1.27	1.31
PBIL	**1.00**	1.46	1.49	1.48	1.52	1.53	1.55
PSO	**1.00**	1.22	1.21	1.25	1.27	1.23	1.23
Total	**1.00**	1.27	1.28	1.02	1.03	1.01	1.02

if the optimal were local, it would be hard to escape from it. R1 introduced diversity into the optimization process of PSO, which would enable the trapped particles to escape from the local positions. If the particles were not trapped into local positions, the addition of population diversity did no harm to PSO, because the global best particle was always memorized during the whole optimization process. This is why F3 performed better than other models except R1. In sum, the PSO combined with six proposed models (especially Models R1 and F3) performed better than or equally to the basic PSO.

B. Performance of Six Information Feedback Models

In this section, we explain how six information feedback models were combined with other nine metaheuristic algorithms, i.e., ACO [6], BA [47], BBO [29], CS [14], DE [10], ES [13], KH [21], MBO [36], and PBIL [37]. These newly combined methods were further benchmarked by 14 standard test functions, as shown in Table I [29].

For an algorithm, different parameter settings have a great impact on its performance. In order to compare fairly, their parameters were set as shown in Table III. For ACO, BBO, DE, ES, PBIL, and PSO, their parameters were the same as in [29].

For most algorithms, different runs may generate different results. In order to get their representative statistical results, 50 independent runs were done for each method. In addition, each method had a population size of 50, an elitism parameter of 2, and were run for 50 generations. The *best*, *average*, and *worst* performances of each method were collected and summarized in Table IV. The results were highlighted in bold if the algorithms performed the best for a benchmark. In order to investigate the influence of F1–3 and R1–3, the number of functions on which the metaheuristic algorithms performed the best was calculated, as shown in the last two columns of Table IV. Table V shows the average CPU time for each method on each benchmark. We must point out that PSO was also included in Tables IV and V in order to get more accurate statistical results.

From Table IV, we see that F2 was the best information feedback model, and had the greatest impact on the three algorithms: 1) BA; 2) CS; and 3) MBO. R1 is inferior only to F2 and had the greatest impact on three algorithms: 1) ES; 2) KH; and 3) PSO. F1 ranked third and had the greatest impact on two algorithms: 1) BBO and 2) DE. For R2 and R3, except ACO, they had the best impact on MBO and PBIL, respectively. Looking carefully at Table IV, for ACO, R2, and R3 had the same impact; for MBO, F2, and R2 had the same impact. In addition, for six information feedback models and F0, their average ranking from good to bad was as follows: F2 > R1 > F1 > F3 = R3 > F0 > R2. Models F1–3 had a greater impact than R1–3 for ten metaheuristic algorithms on 14 benchmarks (230 versus 163).

From Table V, we observed that, except BA, all the variants consumed more time than their respective basic algorithms. This result falls fully under the adage, "there is no free lunch" [92]. The additional time was used mainly to evaluate the fitness values, and that action can be time consuming.

C. Comparisons Using t-Test

Based on the final results of 50 independent runs on 14 functions, Table VI presents the t values on every function of the two-tailed test, with the 5% level of significance between the basic method and improved methods with six information

TABLE VI
COMPARISONS BETWEEN THE BASIC METHOD AND SIX IMPROVED METHODS WITH INFORMATION FEEDBACK MODELS AT α=0.05 ON TWO-TAILED t-TESTS

		F1	R1	F2	R2	F3	R3	Total
ACO	Better	**10**	**12**	**11**	**12**	**10**	**11**	**66**
	Equal	2	1	1	0	2	1	7
	Worse	2	1	2	2	2	2	11
BA	Better	**11**	**11**	**12**	**12**	**12**	**12**	**70**
	Equal	0	0	0	0	0	0	0
	Worse	3	3	2	2	2	2	14
BBO	Better	**10**	**5**	**8**	**4**	**5**	2	**34**
	Equal	1	3	2	2	5	3	16
	Worse	3	6	4	8	4	9	34
CS	Better	0	**11**	**12**	**11**	**12**	**11**	**57**
	Equal	**14**	0	0	0	0	0	14
	Worse	0	3	2	3	2	3	13
DE	Better	**12**	**11**	**12**	**12**	1	**12**	**60**
	Equal	1	0	0	0	0	0	1
	Worse	1	3	2	2	**13**	2	23
ES	Better	**13**	**13**	**12**	**12**	**12**	**12**	**74**
	Equal	0	0	1	1	1	1	4
	Worse	1	1	1	1	1	1	6
KH	Better	0	**13**	**5**	5	4	4	31
	Equal	4	1	4	3	3	3	18
	Worse	**10**	0	**5**	**6**	**7**	**7**	**35**
MBO	Better	**12**	**12**	**11**	**11**	**11**	**11**	**68**
	Equal	0	0	1	1	1	1	4
	Worse	2	2	2	2	2	2	12
PBIL	Better	**13**	**12**	**12**	**12**	**12**	**12**	**73**
	Equal	0	1	1	0	0	0	2
	Worse	1	1	1	2	2	2	9
PSO	Better	**13**	**13**	**12**	**12**	**12**	**12**	**74**
	Equal	0	0	1	1	1	1	4
	Worse	1	1	1	1	1	1	6
Total	Better	**94**	**113**	**107**	**103**	**91**	**99**	**607**
	Equal	22	6	11	8	13	10	70
	Worse	24	21	22	29	36	31	163

TABLE VII
TEN REAL WORLD PROBLEMS SELECTED FROM CEC 2011

No.	*Function*
f01	Parameter Estimation for Frequency-Modulated Sound Waves
f02	Lennard-Jones Potential Problem
f03	Optimal Control of a Non-Linear Stirred Tank Reactor
f04	Transmission Network Expansion Planning
f05	DED instance 1
f06[a]	DED instance 2
f07[a]	ELD Instance 1
f08[a]	ELD Instance 2
f09[a]	ELD Instance 4
f10[a]	ELD Instance 5

[a]f06-f10 are five instances of The ELD Problems in CEC 2011. DED and ELD denote Dynamic Economic Dispatch and Economic Load Dispatch, respectively.

TABLE VIII
OPTIMIZATION RESULTS OBTAINED BY TEN METAHEURISTIC ALGORITHMS WITH SIX INFORMATION FEEDBACK MODELS FOR CEC 2011 RWPS

	F0	*F1*	*R1*	*F2*	*R2*	*F3*	*R3*	*F1∼3*	*R1∼3*
ACO	8	3	**9**	5	4	4	3	12	**16**
BA	5	8	**10**	5	1	0	1	**13**	12
BBO	**10**	**10**	3	4	1	1	1	**15**	5
CS	6	9	1	**12**	0	2	0	**23**	1
DE	10	7	0	**12**	0	0	1	**19**	1
ES	1	2	**18**	2	0	6	1	10	**19**
KH	2	5	**8**	**8**	3	3	1	**16**	12
MBO	6	**9**	4	6	1	4	0	**19**	5
PBIL	**7**	6	**7**	2	1	5	2	**13**	10
PSO	7	**8**	4	1	4	6	0	**15**	8
Total	62	**67**	64	57	15	31	10	**155**	89

feedback models. In the table, the value of t with 98 degrees of freedom was significant at $\alpha = 0.05$ by a two-tailed test. Boldface indicates that the corresponding method performed significantly better than the basic method. The *best*, *equal*, and *worst* in Table VI indicate that the corresponding method performed better than, equal to, or worse than its basic one. In more detail, the *best*, *equal*, and *worst* performance of each method was collected and summarized, as shown in Table VI.

For instance, comparing ACO and six variants of ACO, ACOF1–3, and ACOR1–3 outperformed ACO significantly on ten, twelve, eleven, twelve, ten, and eleven functions, respectively, and performed as well as ACO on two, one, one, zero, two, and one functions, respectively. These results indicate that six variants of ACO generally performed better than ACO in terms of the solution accuracy. Though the performance of ACOF1–3 and ACOR1–3 was slightly weaker on some functions, Table VI also reveals that they outperformed ACO on most functions.

Similarly, Table VI shows that most methods (ACO, BA, CS, DE, ES, MBO, PBIL, and PSO) had absolute advantage over their basic algorithms. The performance of BBO and KH was better than or equal to their basic ones on most benchmarks. In addition, as seen from the last three rows of Table VI, R1 was the best information model; F1, R1, and F2 were the three best models among the six different information feedback models. This conclusion coincides with results in Table IV.

D. Real World Problems

In addition to the standard functions discussed in the section above, ten more real world problems (RWPs) (see Table VII) were also used to validate the six information feedback models. More detailed information about ten RWPs can be found in [93].

Here, the parameters used in the ten approaches were the same as the above. The population size and generations were set to 50 and 50, respectively. The results obtained by 50 independent runs on ten RWPs were recorded in Table VIII. The results were highlighted in bold if an algorithm performed the best on a benchmark. For each model, the total numbers of the bold results were collected, as shown in the last row.

From Table VIII, we see that F1 was the best information feedback model, and had the greatest impact on the three metaheuristic algorithms: 1) BBO; 2) MBO; and 3) PSO. R1 was only slightly inferior to F1 and had the greatest impact on five metaheuristic algorithms: 1) ACO; 2) BA; 3) ES; 4) KH; and 5) PBIL. F2 ranked the third and had the greatest impact on three metaheuristic algorithms: 1) BA; 2) CS; and 3) KH. For the other three information feedback models (R2, F3, and R3), F3 had a greater influence on ten metaheuristic algorithms than R2 and R3. For KH, we can see, R1 and F2 had equal influence. Moreover, for BBO, F1 had the same performance as F0 (the basic BBO). For PBIL, R1 had the same performance as F0 (the basic PBIL). In addition, for six information feedback models and F0, their average ranking from good to bad was as follows: F1 > R1 > F0 > F2 > F3 > R2 > R3. Models F1–3 had a greater impact than R1–3 for ten metaheuristic algorithms on ten RWPs (155 versus 89). From the results on 14 standard functions and ten RWPs, F1, R1, and F2 performed the best among six information models.

Except RWPs studied here, there are many difficult issues deserving to be extensively studied, like cloud data [94], encrypted outsourced data [95], [96], and image copy detection [97]. Shen *et al.* [94] designed a new efficient and effective public auditing protocol with novel dynamic structure for cloud data with the aim of decreasing the computational and communication overheads. Devising a searchable and desirable encryption scheme cannot only support personalized search but also improve user search experience. For this purpose, Fu *et al.* [96] handled the issue of personalized multikeyword ranked search over encrypted data while preserving privacy in cloud computing. Fu *et al.* [95] presented a content-aware search scheme, which can make semantic search more smart. In addition, they verified the privacy and efficiency of their schemes in the experiments. Zhou *et al.* [97] designed a global context verification scheme to filter false matches for copy detection. Concretely, the overlapping region-based global context descriptor was designed to verify these matches to filter false matches. Gu and Sheng [98] proposed an equivalent dual formulation for v-SVC and a robust v-SvcPath based on lower upper decomposition with partial pivoting. Also, their robust regularization path algorithm can avoid the exceptions completely, and handling the singularities in the key matrix.

VI. Discussion

From the experiments conducted in the previous section, each of the ten algorithms was improved by a particular information feedback model. Here, KH is taken as an example to analyze why the information feedback model can improve the performance of all of the algorithms on 14 functions.

KH is a relatively new and promising algorithm proposed by Gandomi and Alavi in 2012 [21]. R1 had the greatest impact on KH among six information feedback models, i.e., $k = 1$, and $j = r_1$ in (1). For krill i, the first and second movements are based mainly on the best krill [21]. This will surely make most krill move toward the promising area. However, at the later search stage, the KH algorithm might be trapped into the local optimum. R1 added more diversity to the population for the optimization process at the later search stage. Meanwhile, the generated krill had a smaller possibility of surpassing the given range. So, the performance of KH was improved significantly.

In addition, different models were able to create a good balance between exploration and exploitation. When k was small, few of the previous individuals were used. In this way, the ability of exploration could be improved. Conversely, when k was big, as many of the previous individuals were used as possible. In this way, the ability of exploitation could be greatly improved. On other hand, the algorithms under models R1–3 had more population diversity and explorative ability than models F1–3. This could improve significantly the performance of the metaheuristic algorithms at the late stage.

After fully investigating the performance of the proposed methods, the following points should be highlighted in future.

First, the variants of a basic method (except BA) consume more CPU time than the basic one because of increased fitness evaluation. Methods to reduce CPU time are worthy of further study. Second, KH and PSO were taken as examples to explain the principle of our models. Further analysis using theories should be performed to ascertain the reasons why the models can improve the performance of their basic algorithms.

VII. Conclusion

In the study of optimization, few metaheuristic algorithms reuse the previous information to guide the later updating process. In this paper, we extracted and used the previous information in the population to give feedback to the main optimization process. One, two, and three individuals in previous iterations were selected by either fixed or random method. Accordingly, six information feedback models were proposed, and they were then incorporated into ten algorithms. The final individual at the current iteration was updated based on the basic algorithm plus some selected previous individuals by using a simple fitness weighting method.

By incorporating six information feedback models into ten algorithms, we constructed six variants of each basic method. They were compared with each other as well as with their basic algorithms via 14 functions and ten CEC 2011 RWPs.

References

[1] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, “Adaptive replacement strategies for MOEA/D,” *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 474–486, Feb. 2016.

[2] F. Liu, W. Pedrycz, and W.-G. Zhang, “Limited rationality and its quantification through the interval number judgments with permutations,” *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4025–4037, Dec. 2017.

[3] F. Liu and W.-G. Zhang, “TOPSIS-based consensus model for group decision-making with incomplete interval fuzzy preference relations,” *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1283–1294, Aug. 2014.

[4] F. Liu, W.-G. Zhang, and Z.-X. Wang, “A goal programming model for incomplete interval multiplicative preference relations and its application in group decision-making,” *Eur. J. Oper. Res.*, vol. 218, no. 3, pp. 747–754, 2012.

[5] G.-G. Wang, X. Cai, Z. Cui, G. Min, and J. Chen, “High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm,” *IEEE Trans. Emerg. Topics Comput.*, to be published. [Online]. Available: http://ieeexplore.ieee.org/document/7927724/, doi: 10.1109/TETC.2017.2703784.

[6] M. Dorigo and T. Stutzle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.

[7] M. A. Khan, W. Shahzad, and A. R. Baig, “Protein classification via an ant-inspired association rules-based classifier,” *Int. J. Bio Inspired Comput.*, vol. 8, no. 1, pp. 51–65, 2016.

[8] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm,” *J. Glob. Optim.*, vol. 39, no. 3, pp. 459–471, 2007.

[9] J.-Q. Li, Q.-K. Pan, and P.-Y. Duan, “An improved artificial bee colony algorithm for solving hybrid flexible flowshop with dynamic operation skipping,” *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1311–1324, Jun. 2016.

[10] R. Storn and K. Price, “Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces,” *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[11] Y.-L. Li *et al.*, “Differential evolution with an evolution path: A DEEP evolutionary algorithm,” *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1798–1810, Sep. 2015.

[12] S. Hui and P. N. Suganthan, “Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization,” *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 64–74, Jan. 2016.

[13] H. Beyer and H. Schwefel, *Natural Computing*. Dordrecht, Netherlands: Kluwer Acad., 2002.

[14] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proc. World Congr. Nat. Biol. Inspired Comput. (NaBIC)*, Coimbatore, India, 2009, pp. 210–214.

[15] G.-G. Wang, A. H. Gandomi, X. Zhao, and H. E. Chu, "Hybridizing harmony search algorithm with cuckoo search for global numerical optimization," *Soft Comput.*, vol. 20, no. 1, pp. 273–285, 2016.

[16] J. Li and Y. Tan, "Orienting mutation based fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2015, pp. 1265–1271.

[17] Y. Shi, "An optimization algorithm based on brainstorming process," *Int. J. Swarm Intell. Res.*, vol. 2, no. 4, pp. 35–62, 2011.

[18] Y. Shi, J. Xue, and Y. Wu, "Multi-objective optimization based on brain storm optimization algorithm," *Int. J. Swarm Intell. Res.*, vol. 4, no. 3, pp. 1–21, 2013.

[19] G.-G. Wang, S. Deb, and L. D. S. Coelho, "Earthworm optimization algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Int. J. Bio Inspired Comput.*, 2015. [Online]. Available: http://www.inderscience.com/info/ingeneral/forthcoming.php?jcode=ijbic, doi: 10.1504/IJBIC.2015.10004283.

[20] G.-G. Wang, S. Deb, X.-Z. Gao, and L. D. S. Coelho, "A new metaheuristic optimisation algorithm motivated by elephant herding behaviour," *Int. J. Bio Inspired Comput.*, vol. 8, no. 6, pp. 394–409, 2017.

[21] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Commun. Nonlin. Sci. Numer. Simulat.*, vol. 17, no. 12, pp. 4831–4845, 2012.

[22] G.-G. Wang, L. Guo, A. H. Gandomi, G.-S. Hao, and H. Wang, "Chaotic krill herd algorithm," *Inf. Sci.*, vol. 274, pp. 17–34, Aug. 2014.

[23] G. Wang *et al.*, "Incorporating mutation scheme into krill herd algorithm for global numerical optimization," *Neural Comput. Appl.*, vol. 24, nos. 3–4, pp. 853–871, 2014.

[24] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "Stud krill herd algorithm," *Neurocomputing*, vol. 128, pp. 363–370, Mar. 2014.

[25] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "An effective krill herd algorithm with migration operator in biogeography-based optimization," *Appl. Math. Model.*, vol. 38, nos. 9–10, pp. 2454–2462, 2014.

[26] G.-G. Wang, A. H. Gandomi, A. H. Alavi, and G.-S. Hao, "Hybrid krill herd algorithm with differential evolution for global numerical optimization," *Neural Comput. Appl.*, vol. 25, no. 2, pp. 297–308, 2014.

[27] G.-G. Wang, A. H. Gandomi, A. H. Alavi, and S. Deb, "A hybrid method based on krill herd and quantum-behaved particle swarm optimization," *Neural Comput. Appl.*, vol. 27, no. 4, pp. 989–1006, 2016.

[28] G.-G. Wang, S. Deb, A. H. Gandomi, and A. H. Alavi, "Opposition-based krill herd algorithm with Cauchy mutation and position clamping," *Neurocomputing*, vol. 177, pp. 147–157, Feb. 2016.

[29] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[30] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. New York, NY, USA: Addison-Wesley, 1998.

[31] X. Sun, D. Gong, Y. Jin, and S. Chen, "A new surrogate-assisted interactive genetic algorithm with weighted semisupervised learning," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 685–698, Apr. 2013.

[32] A. J. Umbarkar, M. S. Joshi, and W.-C. Hong, "Comparative study of diversity based parallel dual population genetic algorithm for unconstrained function optimisations," *Int. J. Bio Inspired Comput.*, vol. 8, no. 4, pp. 248–263, 2016.

[33] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[34] A. Rezoug and D. Boughaci, "A self-adaptive harmony search combined with a stochastic local search for the 0-1 multidimensional knapsack problem," *Int. J. Bio Inspired Comput.*, vol. 8, no. 4, pp. 234–239, 2016.

[35] T. Niknam and A. Kavousi-Fard, "Optimal energy management of smart renewable micro-grids in the reconfigurable systems using adaptive harmony search algorithm," *Int. J. Bio Inspired Comput.*, vol. 8, no. 3, pp. 184–194, 2016.

[36] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Comput. Appl.*, pp. 1–20, May 2015. [Online]. Available: https://link.springer.com/article/10.1007/s00521-015-1923-y, doi: 10.1007/s00521-015-1923-y.

[37] B. Shumeet, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," Carnegie Mellon Univ., Pittsburgh, PA, USA, Rep. CMU-CS-94-163, 1994.

[38] G.-G. Wang, "Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Comput.*, pp. 1–4, Sep. 2016. [Online]. Available: https://link.springer.com/article/10.1007/s12293-016-0212-3, doi: 10.1007/s12293-016-0212-3.

[39] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.

[40] S. Helwig, J. Branke, and S. Mostaghim, "Experimental analysis of bound handling techniques in particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 259–271, Apr. 2013.

[41] J. Li, J. Zhang, C. Jiang, and M. Zhou, "Composite particle swarm optimizer with historical memory for function optimization," *IEEE Trans. Cybern.*, vol. 45, no. 10, pp. 2350–2363, Oct. 2015.

[42] M. Gong, Q. Cai, X. Chen, and L. Ma, "Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 18, no. 1, pp. 82–97, Feb. 2014.

[43] W. Hu and Y. Tan, "Prototype generation using multiobjective particle swarm optimization for nearest neighbor classification," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2719–2731, Dec. 2015.

[44] A. O. Adewumi and M. A. Arasomwan, "On the performance of particle swarm optimisation without some control parameters for global optimisation," *Int. J. Bio Inspired Comput.*, vol. 8, no. 1, pp. 14–32, 2016.

[45] Y.-F. Zhang and H.-D. Chiang, "A novel consensus-based particle swarm optimization-assisted trust-tech methodology for large-scale global optimization," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2717–2729, Sep. 2017.

[46] W. Hu, G. G. Yen, and G. Luo, "Many-objective particle swarm optimization using two-stage strategy and parallel cell coordinate system," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1446–1459, Jun. 2017.

[47] X. S. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, 2012.

[48] X. Cai, X.-Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *Int. J. Bio Inspired Comput.*, vol. 8, no. 4, pp. 205–214, 2016.

[49] I. Ciornei and E. Kyriakides, "Hybrid ant colony-genetic algorithm (GAAPI) for global continuous optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 234–245, Feb. 2012.

[50] D. Simon, M. Ergezer, D. Du, and R. Rarick, "Markov models for biogeography-based optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 299–306, Feb. 2011.

[51] Y. Tan, *Fireworks Algorithm-A Novel Swarm Intelligence Optimization Method*. Heidelberg, Germany: Springer-Verlag, 2015, p. 323.

[52] Z. Bingul, "Adaptive genetic algorithms applied to dynamic multiobjective problems," *Appl. Soft Comput.*, vol. 7, no. 3, pp. 791–799, 2007.

[53] W. Gong, A. Zhou, and Z. Cai, "A multioperator search strategy based on cheap surrogate models for evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 746–758, Oct. 2015.

[54] B. Liu, Q. Zhang, and G. G. E. Gielen, "A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 180–192, Apr. 2014.

[55] H. Wang, Y. Jin, and J. O. Janson, "Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 939–952, Dec. 2016.

[56] M. H. S. Mendes, G. L. Soares, J. L. Coulomb, and J. A. Vasconcelos, "A surrogate genetic programming based model to facilitate robust multi-objective optimization: A case study in magnetostatics," *IEEE Trans. Magn.*, vol. 49, no. 5, pp. 2065–2068, May 2013.

[57] A. Kattan and Y.-S. Ong, "Surrogate genetic programming: A semantic aware evolutionary search," *Inf. Sci.*, vol. 296, pp. 345–359, Mar. 2015.

[58] A. Rosales-Pérez, J. A. Gonzalez, C. A. C. Coello, H. J. Escalante, and C. A. Reyes-Garcia, "Surrogate-assisted multi-objective model selection for support vector machines," *Neurocomputing*, vol. 150, pp. 163–172, Feb. 2015.

[59] T. Hildebrandt and J. Branke, "On using surrogates with genetic programming," *Evol. Comput.*, vol. 23, no. 3, pp. 343–367, 2015.

[60] C.-J. Lin, M.-S. Chern, and M. Chih, "A binary particle swarm optimization based on the surrogate information with proportional acceleration coefficients for the 0-1 multidimensional knapsack problem," *J. Ind. Prod. Eng.*, vol. 33, no. 2, pp. 77–102, 2016.

[61] W. Gao, F. T. S. Chan, L. Huang, and S. Liu, "Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood," *Inf. Sci.*, vol. 316, pp. 180–200, Sep. 2015.

[62] Y.-J. Gong *et al.*, "Genetic learning particle swarm optimization," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.

[63] D. L. Ly and H. Lipson, "Optimal experiment design for coevolutionary active learning," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 394–404, Jun. 2014.

[64] A. Zhou, J. Sun, and Q. Zhang, "An estimation of distribution algorithm with cheap and expensive local search methods," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 807–822, Dec. 2015.

[65] J. Xiong, J. Liu, Y. Chen, and H. A. Abbass, "A knowledge-based evolutionary multiobjective approach for stochastic extended resource investment project scheduling problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 742–763, Oct. 2014.

[66] W.-F. Gao, L.-L. Huang, S.-Y. Liu, and C. Dai, "Artificial bee colony algorithm based on information learning," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2827–2839, Dec. 2015.

[67] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "Enhancing artificial bee colony algorithm using more information-based search equations," *Inf. Sci.*, vol. 270, pp. 112–133, Jun. 2014.

[68] G.-G. Wang, B. Chang, and Z. Zhang, "A multi-swarm bat algorithm for global optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Sendai, Japan, 2015, pp. 480–485.

[69] A. Ghosh, S. Das, A. Chowdhury, and R. Giri, "An improved differential evolution algorithm with fitness-based adaptation of the control parameters," *Inf. Sci.*, vol. 181, no. 18, pp. 3749–3765, 2011.

[70] M. Z. Ali, N. H. Awad, and P. N. Suganthan, "Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization," *Appl. Soft Comput.*, vol. 33, pp. 304–327, Aug. 2015.

[71] L. Cui, G. Li, Q. Lin, J. Chen, and N. Lu, "Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations," *Comput. Oper. Res.*, vol. 67, pp. 155–173, Mar. 2016.

[72] W.-F. Gao, G. G. Yen, and S.-Y. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 1108–1121, May 2015.

[73] J. Wang, W. Zhang, and J. Zhang, "Cooperative differential evolution with multiple populations for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2848–2861, Dec. 2016.

[74] K. G. Dhal, M. I. Quraishi, and S. Das, "Development of firefly algorithm via chaotic sequence and population diversity to enhance the image contrast," *Natural Comput.*, vol. 15, no. 2, pp. 307–318, 2015.

[75] Q.-K. Pan, P. N. Suganthan, J. J. Liang, and M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations," *Eng. Optim.*, vol. 42, no. 2, pp. 101–117, 2010.

[76] W. Hu and G. G. Yen, "Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 1–18, Feb. 2015.

[77] R. Foss, "A self organising network model of information gathering by the honey bee swarm," *Kybernetes*, vol. 44, no. 3, pp. 353–367, 2015.

[78] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[79] W. Gong, Z. Cai, C. X. Ling, and L. Hui, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 2, pp. 397–413, Apr. 2011.

[80] Z. Peng, J. Liao, and Y. Cai, "Differential evolution with distributed direction information based mutation operators: An optimization technique for big data," *J. Ambient Intell. Humanized Comput.*, vol. 6, no. 4, pp. 481–494, 2015.

[81] J. Liao, Y. Cai, T. Wang, H. Tian, and Y. Chen, "Cellular direction information based differential evolution for numerical optimization: An empirical study," *Soft Comput.*, vol. 20, no. 7, pp. 2801–2827, 2015.

[82] Y. Cai *et al.*, "Adaptive direction information in differential evolution for numerical optimization," *Soft Comput.*, vol. 20, no. 2, pp. 465–494, 2014.

[83] W. Fang, J. Sun, H. Chen, and X. Wu, "A decentralized quantum-inspired particle swarm optimization algorithm with cellular structured population," *Inf. Sci.*, vol. 330, pp. 19–48, Feb. 2016.

[84] G.-G. Wang, M. Lu, and X.-J. Zhao, "An improved bat algorithm with variable neighborhood search for global optimization," in *Proc. IEEE Congr. Evol. Comput. (IEEE CEC)*, Vancouver, BC, Canada, 2016, pp. 1773–1778.

[85] X. He, H. Guan, and J. Qin, "A hybrid wavelet neural network model with mutual information and particle swarm optimization for forecasting monthly rainfall," *J. Hydrol.*, vol. 527, pp. 88–100, Aug. 2015.

[86] J. Shang, J. Zhang, X. Lei, Y. Zhang, and B. Chen, "Incorporating heuristic information into ant colony optimization for epistasis detection," *Genes Genomics*, vol. 34, no. 3, pp. 321–327, 2012.

[87] X. Wang and L. Tang, "An adaptive multi-population differential evolution algorithm for continuous multi-objective optimization," *Inf. Sci.*, vol. 348, pp. 124–141, Jun. 2016.

[88] S.-Y. Park and J.-J. Lee, "Stochastic opposition-based learning using a beta distribution in differential evolution," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2184–2194, Oct. 2016.

[89] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[90] C. Yang and M. Kumar, "An information guided framework for simulated annealing," *J. Glob. Optim.*, vol. 62, no. 1, pp. 131–154, 2014.

[91] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "Exploratory landscape analysis of continuous space optimization problems using information content," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 74–87, Feb. 2015.

[92] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[93] S. Das and P. Suganthan, *Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems*, 2010.

[94] J. Shen, J. Shen, X. Chen, X. Huang, and W. Susilo, "An efficient public auditing protocol with novel dynamic structure for cloud data," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 10, pp. 2402–2415, Oct. 2017.

[95] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang, "Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 8, pp. 1874–1884, Aug. 2017.

[96] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 9, pp. 2546–2559, Sep. 2016.

[97] Z. Zhou, Y. Wang, Q. M. J. Wu, C.-N. Yang, and X. Sun, "Effective and efficient global context verification for image copy detection," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1, pp. 48–63, Jan. 2017.

[98] B. Gu and V. S. Sheng, "A robust regularization path Algorithm for ν-support vector classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1241–1248, May 2017.

Gai-Ge Wang (M'15) received the Ph.D. degree in computational intelligence and its applications with the Chinese Academy of Sciences, Beijing, China.

He proposed five bio-inspired algorithms: monarch butterfly optimization, earthworm optimization algorithm, elephant herding optimization, moth search algorithm, and Rhino Herd. He has published 88 papers. He has over 2300 Google Scholar citations. His current research interests include swarm intelligence, evolutionary computation, and big data optimization.

Dr. Wang served as an Associate Editor of the *International Journal of Computer Information Systems and Industrial Management Applications* and an Editorial Board Member of International Journal of Bio-Inspired Computation from 2016. He served as a Guest-Editor for many journals including the *International Journal of Bio-Inspired Computation*, *Operational Research*, *Memetic Computing*, and *Future Generation Computer Systems*.

Ying Tan (SM'02) received the Ph.D. degree from Southeast University, Nanjing, China, in 1997.

He is a Full Professor and the Ph.D. Advisor with the School of Electronics Engineering and Computer Science, Peking University, Beijing, China. He is the Inventor of fireworks algorithm. His current research interests include swarm intelligence, machine learning, and data mining and their applications.

Dr. Tan was a recipient of the 2nd-Class Natural Science Award of China in 2009. He serves as the Editor-in-Chief of the *International Journal of Computational Intelligence* and *Pattern Recognition* and the Associate Editor of the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and the *International Journal of Swarm Intelligence Research*. He served as an Editor of Springer's LNCS for over 20 volumes and a Guest Editor of several referred journals, including *Information Sciences*, *Soft Computing*, *Neurocomputing*, *Natural Computing*, and the IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS. He has been the Founding General Chair of the series International Conference on Swarm Intelligence since 2010, and the Joint General Chair of the first and second BRICS CCI, and the 2014 IEEE WCCI Program Committee Co-Chair.