Adaptive Operator Selection With Bandits for a Multiobjective Evolutionary Algorithm Based on Decomposition

Ke Li, *Student Member, IEEE,* Álvaro Fialho, *Member, IEEE,* Sam Kwong, *Senior Member, IEEE,* and Qingfu Zhang, *Senior Member, IEEE*

Abstract—Adaptive operator selection (AOS) is used to determine the application rates of different operators in an online manner based on their recent performances within an optimization process. This paper proposes a bandit-based AOS method, fitness-rate-rank-based multiarmed bandit (FRRMAB). In order to track the dynamics of the search process, it uses a sliding window to record the recent fitness improvement rates achieved by the operators, while employing a decaying mechanism to increase the selection probability of the best operator. Not much work has been done on AOS in multiobjective evolutionary computation since it is very difficult to measure the fitness improvements quantitatively in most Pareto-dominance-based multiobjective evolutionary algorithms. Multiobjective evolutionary algorithm based on decomposition (MOEA/D) decomposes a multiobjective optimization problem into a number of scalar optimization subproblems and optimizes them simultaneously. Thus, it is natural and feasible to use AOS in MOEA/D. We investigate several important issues in using FRRMAB in MOEA/D. Our experimental results demonstrate that FRRMAB is robust and its operator selection is reasonable. Comparison experiments also indicate that FRRMAB can significantly improve the performance of MOEA/D.

Index Terms—Adaptive operator selection (AOS), decomposition, multiarmed bandit, multiobjective evolutionary algorithm based on decomposition (MOEA/D), multiobjective optimization.

I. INTRODUCTION

A MULTIOBJECTIVE optimization problem (MOP) consists of several conflicting objectives to optimize. An

MOP can be stated as

$$
\begin{aligned}
\text{minimize} \quad & F(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T \\
\text{subject to} \quad & x \in \Omega
\end{aligned}
\tag{1}
$$

where $\Omega \subset R^n$ is the decision (variable) space and $x = (x_1, \ldots, x_n)^T \in \Omega$ is a candidate solution. $F : \Omega \to R^m$ constitutes m objective functions.

Let $u, v \in R^m$, and u is said to dominate v if and only if $u_i \leq v_i$ for every $i \in \{1, \ldots, m\}$ and $u_j < v_j$ for at least one index $j \in \{1, \ldots, m\}$. A solution x^* is Pareto optimal to (1) if there is no other solution $x \in \Omega$ such that $F(x)$ dominates $F(x^*)$. $F(x^*)$ is then called a Pareto optimal (objective) vector. In other words, any improvement in a Pareto optimal solution in one objective must lead to deterioration in at least one other objective. The set of all the Pareto optimal solutions is called the Pareto set (PS). Accordingly, the set of all the Pareto optimal vectors, $PF = \{F(x) \in R^m | x \in PS\}$, is called the Pareto front (PF) [1].

Evolutionary algorithms (EAs) are stochastic optimization algorithms inspired by the Darwinian evolution theory. Over the last two decades, much effort has been devoted to developing multiobjective evolutionary algorithms (MOEAs). These methods can generate a set of representative Pareto optimal solutions in a single run. Moreover, they can be used for solving problems without good mathematical properties. Since the pioneering work of Schaffer [2], a number of MOEAs has been proposed (e.g., [3]–[8]) and applied to a wide range of problem domains [9], [10].

A major issue in using EAs, including MOEAs, is that one has to set a number of control parameters and select genetic operators to which the algorithm performance is often very sensitive. Different problems require different settings; people from other domains often find it very hard to make a proper setting for their problems without help from EA experts. For this reason, automatic parameter and operator configuration has been a very important and active research topic in the EA community [11].

In this paper, we propose and study an adaptive operator selection (AOS) method for deciding which operator should be employed at a time point in an MOEA. There are two main tasks that need to be performed in an AOS method. One is to decide how much reward (i.e., credit) should be

assigned to an operator based on its recent performance in the search process. The other is to select an operator to use at the next time point based on these current reward values. A fundamental issue behind these two highly related tasks is the so-called exploration versus exploitation (EvE) dilemma: one hopes to give more chances to operators with better track records (exploitation), but also wants to explore poor operators in the future search (exploration), since an operator might perform significantly differently at different search stages.

The EvE dilemma has intensively been studied in the game theory community for dealing with the multiarmed bandit (MAB) problem. Many efficient methods for the MAB problem can be found in the literature. A prominent method is the upper confidence bound (UCB) algorithm [12], which guarantees asymptotic optimality in terms of the total cumulative reward. However, it is not straightforward to apply these methods to the AOS problem in EAs. The MAB problem in the game theory considers stationary reward distributions, while the performance of an operator in the AOS problem might significantly vary during the search process. Moreover, the range of reward values is problem dependent in the AOS scenario, while, in the original MAB case, the rewards are always Boolean.

Some effort has been dedicated to the modification of MAB algorithms, in particular, the UCB algorithm to address the dynamic and unpredictable aspects of the AOS problem. The dynamic MAB algorithm [13] recomputes the bandit statistics from scratch whenever a change in the reward distribution is detected. The sliding MAB method [14] uses a sliding window to achieve a faster update of the quality measure of each operator to better fit the current stage of the search process. An extreme value-based credit assignment was proposed in [15], which assumes that rare but high fitness improvements are more important than frequent and moderate ones. To improve algorithm robustness for different fitness landscapes, two rank-based credit assignment schemes, the area-under-curve (AUC) and the sum-of-ranks (SR), were proposed in [16]. These bandit-based operator selection methods and credit assignment schemes have been used in several different EAs, including genetic algorithms and differential evolution (DE). Other AOS methods, based on probabilities, can be found in the literature, such as the ones used in SaDE [17], JADE [18], and *j*DE [19] algorithms. However, most of the existing works concern the single-objective optimization problems, in which the fitness improvement of an offspring solution against its parent can easily be measured. In MOPs, however, some solutions are incomparable. Thus, it is not straightforward to define and compute the quality difference between two solutions. Probably, due to this reason, the use of AOS in MOEAs is still relatively scarce. Some pioneering works along this direction include JADE2 [20], MOSaDE [21], OW-MOSaDE [22], Adap-MODE [23], and MODE/SN [24]. It is also worth noting that some other methods consider the so-called multioperators, such as the ensemble-based method [25] and the composite operators [26]. However, a bandit-based AOS method is different from these. It selects an operator according to some deterministic rules based on MAB algorithms, while the others usually select an operator based on some stochastic rules.

Our proposed bandit-based AOS method (FRRMAB), in this paper, employs a sliding window to follow the dynamics of the search process. Instead of using the raw fitness value, the fitness improvement rate is used to evaluate the quality difference between the parent solution and its offspring in a normalized manner. In addition, in order to increase the influence of the best operator at the current time point, a decaying mechanism is introduced to calculate the final credit value of each operator. The FRRMAB AOS method is used within an multiobjective evolutionary algorithm based on a decomposition (MOEA/D) variant, MOEA/D-DRA [27], which was the winning algorithm in the CEC 2009 MOEA contest. The underlying mechanism of MOEA/D [6] is to decompose an MOP into a number of single-objective optimization problems. Thus, strategies for single-objective optimization can easily be used in MOEA/D. The resulting algorithmic combination, referred to as MOEA/D-FRRMAB, is experimentally studied on a set of multiobjective optimization benchmark problems. This outperforms some baseline algorithms, including some state-of-the-art variants of MOEA/D, and MOEA/D using other AOS methods.

The remainder of this paper is organized as follows. The background and some related works are reviewed in Section II. Our proposed AOS method is described in detail in Section III, while its combination with the MOEA/D algorithm is presented in Section IV. Experimental results are given and analyzed in Section V. Section VI concludes this paper and discusses some possible research issues for further work.

II. BACKGROUND AND RELATED WORK

The performance of an EA depends on its parameter settings. Operators can also be regarded as parameters in an EA. A brief overview of different manners of doing a parameter setting in EAs is given in Section II-A. Then, Section II-B presents a more detailed description and review on the AOS methods.

A. Parameter Setting in Evolutionary Algorithms

Different kinds of approaches for parameter setting in EAs have been proposed in the literature. A taxonomy proposed in [28] classifies them into two categories. One is for parameter tuning and the other is for parameter control. The former sets the parameters in an offline manner based on statistics extracted from several runs, and provides a fixed parameter setting that can be used for solving a new instance. Parameter control methods, which dynamically adjust the parameters during the run, can be further divided into three classes. The first is the deterministic method that adjusts the parameters according to some predefined problem-specific rules. The second is the self-adaptive method that lets the parameter values evolve by themselves by encoding them within the genotype of candidate solutions. The last one is the adaptive or feedback-based method, which adjusts the parameter values based on the feedbacks received from the previous search steps.

B. Adaptive Operator Selection

AOS is a recent paradigm to adaptively select operators for generating a new solution during the search. The selection

is conducted based on the recent performances of operators. AOS involves two major tasks: the credit assignment and the operator selection. The former defines how to reward an operator based on its recent performance in the search process, while the latter uses these rewards to decide which operator should be applied next.

1) *Credit Assignment:* The most commonly used metric in credit assignment for measuring the quality of an operator is based on the fitness improvement obtained by a new solution, compared with a given baseline solution. In [29], for instance, the best solution of the current population is considered baseline, while in [30] the offspring's parent is used for comparison. Once the quality is measured, there are different ways to use it for rewarding the operator. The most popular approach is to set the reward amount as the average of the recent fitness improvements [31]. Assuming that rare, but large, improvements might be more important than small ones, [32] suggests that each operator should be rewarded based on the maximal fitness improvement that it recently achieved. To avoid getting trapped at locally optimal solutions, [33] takes account of both diversity and fitness improvement for rewarding operators. Aiming at robustness, [16] has proposed two rank-based credit assignment schemes: the AUC and the SR methods.

2) *Operator Selection:* Guided by the reward values, operator selection methods select operators for generating new solutions. The two most promising selection mechanisms are: 1) the probability-based methods, such as the probability matching (PM) [34] and the adaptive pursuit (AP) [35]; and 2) the bandit-based methods. As these methods will be used as baselines for comparison in Section V, we will give a detailed review in the following.

   a) *Probability matching and adaptive pursuit:* These methods often use a roulette wheel-like process for selecting an operator. Let the set (or pool) of operators be $S = \{s_1, \ldots, s_K\}$, the selection probability of each operator at time point t be $P_t = (p_{1,t}, \ldots, p_{K,t})(\forall t : p_{\min} \leq p_{i,t} \leq 1; \sum_{i=1}^{K} p_{i,t} = 1)$, and the estimated quality of operator i be $\hat{q}_{i,t}$. At each time point t, these methods:

1) randomly select an operator, say, operator i, based on the probability vector P_t, and use a credit assignment scheme to give it a reward r;
2) update the quality of operator i

$$\hat{q}_{i,t+1} = (1 - \alpha) \times \hat{q}_{i,t} + \alpha \times r \qquad (2)$$

where $\alpha \in [0, 1]$ is a user-defined parameter, the adaptation rate.

There are different approaches for updating the selection probability. The PM approach does it as

$$p_{i,t+1} = p_{\min} + (1 - K \times p_{\min}) \times \frac{\hat{q}_{i,t+1}}{\sum_{j=1}^{K} \hat{q}_{j,t+1}} \qquad (3)$$

where $p_{\min} \in [0, 1]$ is the minimal selection probability value for each operator. This ensures that all the operators have a nonzero selection probability, which can avoid losing a currently bad operator that might become useful in the future. An inefficient operator will have its selection probability converging toward $p_{\min}$, while an operator that works very well

will be selected with probability $p_{\max} = (1 - (K - 1) \times p_{\min})$ after getting very good rewards for a long time period. A criticism of the PM approach is that it gives high probabilities to operators with average performances, which may slow down the underlying optimization algorithms. The AP method, originally proposed for learning automata, adopts a winner-take-all strategy to increase the chance of selecting the best operator i_t^*. This works as

$$\begin{cases} i_t^* = \underset{i \in \{1 \cdots K\}}{\operatorname{argmax}}\{\hat{q}_{i,t}\} \\ p_{i,t+1} = p_{i,t} + \beta \times (p_{\max} - p_{i,t}), & \text{if } i = i_t^* \\ p_{i,t+1} = p_{i,t} + \beta \times (p_{\min} - p_{i,t}), & \text{otherwise} \end{cases} \qquad (4)$$

where the learning rate $\beta \in [0, 1]$ controls the greediness of the winner-take-all strategy.

Some other AOS methods based on probabilities can also be found in the literature, such as the ones used in the SaDE [17], JADE [18], and jDE [19] algorithms.

   b) *Bandit-based operator selection:* The operator selection problem can be regarded as an instance of the EvE dilemma: one should exploit the operators set by selecting good operators as often as possible, while also doing some exploration to give chances to poor operators since they may become better in the future search. The EvE dilemma has intensively been studied in the context of the MAB problem, which considers a set of K independent arms (equivalent to variation operators or strategies in the AOS literature), and each arm has an unknown probability of being rewarded. An optimal arm selection strategy is the one that maximizes the cumulative reward along time. Many MAB algorithms have been proposed to tackle this problem. Most of the recent ones are based on the UCB algorithm [12], which provides asymptotic optimality guarantees. In an UCB-based MAB algorithm, the ith arm has an empirical quality estimate $\hat{q}_i$ and a confidence interval that depends on the number of times n_i it has been tried before. At each time point t, the arm maximizing the following function is selected:

$$\hat{q}_{i,t} + C \times \sqrt{\frac{2 \times \ln \sum_{j=1}^{K} n_{j,t}}{n_{i,t}}} \qquad (5)$$

where C is a scaling factor to control the tradeoff between exploitation (the first term that favors the arms with best empirical rewards) and exploration (the square root term that favors the infrequently tried arms).

Some variants of the UCB algorithm have recently been proposed to address the dynamic aspect of the AOS problem in a more efficient way. A very recent one, the sum-of-ranks MAB (SRMAB) algorithm [16], will be used as a baseline for comparison with the new bandit-based AOS method we are proposing in this paper. More specifically, at a given time point t, SRMAB selects an operator to be applied next as follows. First, the latest W fitness improvements achieved by the operators are ranked. Then, a decaying mechanism is applied over these rank values so that the top-ranked rewards, i.e., the highest fitness improvements exert a higher influence on the calculation of the credits assigned to the operators. Finally, it selects the operator that maximizes (5), in which the empirical quality estimate $\hat{q}_{i,t}$ of each operator is the sum

of their corresponding decayed rank values, and $n_{i,t}$ is the number of times operator i has been applied in the recent W applications. The SRMAB algorithm is well known as being both very efficient and robust with respect to different fitness landscapes. The robustness comes mainly from the use of ranks on its credit assignment module. A more detailed survey of SRMAB and other bandit-based algorithms for the AOS problem can be found in [36].

III. FITNESS-RATE-RANK-BASED MULTIARMED BANDIT ADAPTIVE OPERATOR SELECTION

In this section, we propose a new bandit-based method for the AOS problem. The proposed method pays particular attention to the AOS dynamic nature. This consists of two modules. One is for credit assignment and the other is for operator selection.

A. Credit Assignment

In credit assignment, one needs, as discussed in Section II-B1, to address the following two issues:

1) how to measure the impact in the search process caused by the application of an operator;
2) how to assign an appropriate credit value to an operator based on this measured impact.

As for the first issue, the most commonly used approach is to directly use the raw values of the fitness improvements caused by the recent uses of the operator under assessment. However, the range of raw fitness improvements varies from problem to problem and even at the different stages of an optimization process. It is common that the raw fitness improvement value is much larger at early stages than at later ones. Therefore, as discussed in [36], the direct use of raw fitness improvement could deteriorate the algorithm's robustness. To alleviate this problem, our proposed method uses the fitness improvement rates (FIR). More specifically, the FIR achieved by an operator i at time point t is defined as

$$FIR_{i,t} = \frac{pf_{i,t} - cf_{i,t}}{pf_{i,t}} \qquad (6)$$

where $pf_{i,t}$ is the fitness value of the parent, and $cf_{i,t}$ is the fitness value of the offspring.

A sliding window with fixed size W is used to store the FIR values of the recently used operators. It is organized as a first-in, first-out (FIFO) queue, i.e., the FIR value of the most recently used operator is added at the tail of the sliding window, while the oldest record (the item at the head of the queue) is removed to keep the window size constant. Fig. 1 illustrates the structure of a sliding window. Each slot in the sliding window stores two components:

1) the index of the operator op used;
2) its FIR value.

The major reason for using the sliding window is that, in dynamic AOS environments, the performance of an operator in a very early stage may be irrelevant to its current performance. The sliding window ensures that the stored FIR information is for the current situation of the search.

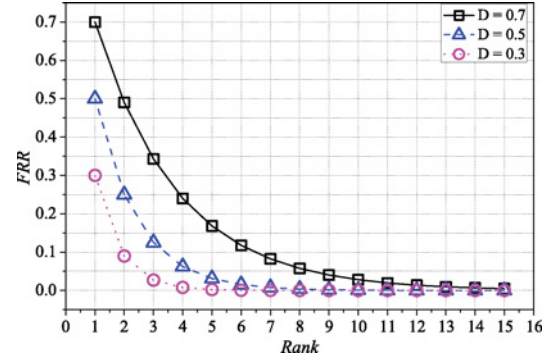Fig. 1. Illustration of the FIFO sliding window structure.

Fig. 2. Comparison between different decaying mechanisms.

To address the second issue set at the outset of this subsection, we first compute $Reward_i$, the sum of all FIR values for each operator i in the current sliding window. Then, we rank all these $Reward_i$ values in a descending order. Let $Rank_i$ be the rank value of operator i, inspired by other recently proposed rank-based credit assignment schemes [16], [32], and to give more chances to the best operators, we introduce a decaying factor $D \in [0, 1]$ to transform $Reward_i$ to

$$Decay_i = D^{Rank_i} \times Reward_i. \qquad (7)$$

Then, we assign the following credit value to operator i:

$$FRR_{i,t} = \frac{Decay_i}{\sum_{j=1}^{K} Decay_j}. \qquad (8)$$

Clearly, the smaller the value of D, the larger the influence for the best operator. Fig. 2 illustrates FRR versus $Rank$ with three different values of D in a case of 15 distinct rank values.

The pseudocode of the proposed credit assignment scheme is given in Algorithm 1.

B. Operator Selection

Based on the received credit values, the operator selection scheme selects operators for generating new solutions. This paper uses a bandit-based operator selection scheme. Our scheme is similar to the original UCB algorithm [12]. The major difference is that we use FRR values as the quality index instead of the average of all the rewards received so far for an operator. In addition, n_i indicates the number of times operator i has been selected in the recent W applications.

The pseudocode of our proposed bandit-based operator selection scheme is given in Algorithm 2. The combination of this operator selection and the credit assignment schemes constitutes our proposed AOS method, FRRMAB. It is worth noting that no operator has yet been applied at the beginning of the search; thus, we give each operator an equal chance to be selected in this case. FRRMAB is not employed until each operator has been applied at least once.

Algorithm 1: Procedure for the Credit Assignment

---

1 Initialize each reward $Reward_i = 0$;
2 Initialize $n_i = 0$;
3 **for** $i \leftarrow 1$ **to** SlidingWindow.*length* **do**
4 op = SlidingWindow.*GetIndexOp*(i);
5 FIR = SlidingWindow.*GetFIR*(i);
6 $Reward_{op} = Reward_{op} + FIR$;
7 n_{op}++;
8 **end**
9 Rank $Reward_i$ in descending order and set $Rank_i$ to be the rank value of operator i;
10 **for** $op \leftarrow 1$ **to** K **do**
11 $Decay_{op} = D^{Rank_{op}} \times Reward_{op}$;
12 **end**
13 $DecaySum = \sum_{op=1}^{K} Decay_{op}$;
14 **for** $op \leftarrow 1$ **to** K **do**
15 $FRR_{op} = Decay_{op}/DecaySum$;
16 **end**

Algorithm 2: The procedure for the bandit-based operator selection

---

1 **if** *There are operators that have not been selected* **then**
2 op_t = one that uniformly randomly selected from the operators pool;
3 **else**
4 $op_t = \underset{i=\{1...K\}}{\arg\max} \left(FRR_{i,t} + C \times \sqrt{\frac{2 \times \ln \sum_{j=1}^{K} n_{j,t}}{n_{i,t}}} \right)$;
5 **end**

IV. INTEGRATION OF FRRMAB WITH MOEA/D

As mentioned in Section I, most existing AOS methods were designed and applied on single-objective optimization. One of the major reasons could be that most popular MOEAs, such as NSGA-II [3], SPEA2 [4], and PAES [37], compare solutions' qualities mainly based on their dominance relations. In these algorithmic frameworks, it is very difficult to quantify the quality difference between two solutions and thus the improvement caused by the application of an operator. Therefore, it is not trivial to use AOS methods, which require a metric for measuring operator performances to enhance these dominance-based MOEAs.

MOEA/D [6] decomposes an MOP into a number of single-objective subproblems. Then, a population-based method is used to solve these subproblems simultaneously. Techniques for single-objective optimization can be naturally, at least in principle, applied to the MOEA/D framework. Several MOEA/D variants have been proposed for dealing with various MOPs, and MOEA/D has also been used as a basic element in some hybrid algorithms (e.g. [38]–[42]). Any improvement on MOEA/D could be of practical interest. For these reasons, this paper investigates how to use FRRMAB to improve MOEA/D.

A. MOEA/D

It is well known that a Pareto optimal solution to an MOP, under mild conditions, is an optimal solution of a single-objective optimization problem whose objective is a weighted (linear or nonlinear) aggregation of all the individual objectives. To obtain a set of different Pareto optimal solutions to approximate the PF, MOEA/D solves a set of such single-objective optimization subproblems with different weight vectors simultaneously. MOEA/D defines neighborhood relations among these subproblems based on the distances between their weight vectors. Each subproblem is optimized in MOEA/D by using information mainly from its neighboring subproblems.

There are several different variants of MOEA/D. This paper uses MOEA/D with dynamical resource allocation (MOEA/D-DRA [27]), which won the CEC 2009 MOEA contest. In principle, MOEA/D can use any decomposition approach for defining their subproblems; we employ the Tchebycheff approach in this paper. Let $\Lambda = \{\lambda^1, \ldots, \lambda^N\}$ be a set of N evenly spread weight vectors. Each $\lambda^j = (\lambda_1^j, \ldots, \lambda_m^j)^T$ satisfies $\sum_{i=1}^{m} \lambda_i^j = 1$ and $\lambda_i^j \geq 0$ for all $i \in \{1, \cdots, m\}$. Let $z^* = (z_1^*, \ldots, z_m^*)^T$ be a utopian ideal point. Then, the problem of approximating the PF of (1) can be decomposed into N scalar optimization subproblems, and the objective function of the jth subproblem is to minimize the following function:

$$g^{te}(x|\lambda^j, z^*) = \max_{1 \leq i \leq m} \{\lambda_i^j |f_i(x) - z_i^*|\}. \quad (9)$$

It is worth noting that z^* is usually unknown before the search; the algorithm uses the lowest f_i-value found during the search to substitute z_i^*. During the search, MOEA/D maintains:

1) a population of N solutions $x^1, \ldots, x^N \in \Omega$, where x^i is the current solution to the ith subproblem;
2) $FV^1, \ldots, FV^N$, where FV^i is the F-value of x^i, i.e., $FV^i = F(x^i)$ for each $i \in \{1, \ldots, N\}$;
3) $z^* = (z_1^*, \ldots, z_m^*)^T$, where z_i^* is the best (i.e., smallest) value found so far for objective f_i;
4) π^i: utility of subproblem i, which measures how much improvement has been caused by x^i in reducing the objective of this subproblem; this is defined as

$$\pi^i = \begin{cases} 1, & \text{if } \Delta^i > 0.001 \\ \left(0.95 + 0.05 \times \frac{\Delta^i}{0.001}\right) \times \pi^i, & \text{otherwise} \end{cases} \quad (10)$$

where Δ^i is the relative decrease of the objective function value of subproblem i.

For each weight vector, its T-neighborhoods are the set of T closest weight vectors to it. Correspondingly, each solution and each subproblem have their own T-neighborhoods.

At each generation, a set of solutions are selected from the current population based on their utilities. For each selected solution x^i, MOEA/D does the following procedures.

1) Set the mating and update range P to be the T-neighborhoods of x_i with a large probability δ, and the whole population otherwise.
2) Randomly select several current solutions from P.
3) Apply genetic operators on the above-selected solutions to generate a new solution y, and evaluate $F(y)$.

4) Replace at most n_r solutions in P by y if it is better than them in terms of their respective subproblems' objective functions.

Further details of MOEA/D-DRA can be found in [27].

B. Using FRRMAB to Enhance MOEA/D

To use FRRMAB to enhance MOEA/D, the operators pool and reward calculation are set as follows.

1) Operators Pool: Many different DE mutation operators have been proposed. Four of them, which present distinct search characteristics, are chosen for the AOS in our experiments:

1) DE/rand/1
   $$v^i = x^i + F \times (x^{r_1} - x^{r_2});$$
2) DE/rand/2
   $$v^i = x^i + F \times (x^{r_1} - x^{r_2}) + F \times (x^{r_3} - x^{r_4});$$
3) DE/current-to-rand/2
   $$v^i = x^i + K \times (x^i - x^{r_1}) + F \times (x^{r_2} - x^{r_3}) + F \times (x^{r_4} - x^{r_5});$$
4) DE/current-to-rand/1
   $$v^i = x^i + K \times (x^i - x^{r_1}) + F \times (x^{r_2} - x^{r_3})$$

where x^i is called the target vector and v^i is the mutant vector. The variables x^{r_1}, x^{r_2}, x^{r_3}, x^{r_4}, and x^{r_5} are different solutions randomly selected from P, which are also different from x^i. The scaling factor $F > 0$ controls the impact of the vector differences on the mutant vector. $K \in [0, 1]$ plays a similar role to F. For the last two mutation operators, the offspring is the direct output of mutation, where the crossover operator will not be used. For the first two mutation operators, a crossover operator is applied upon x^i and v^i for generating the offspring u^i. The binomial crossover [43] is used in our experiments. It works as

$$u^i_j = \begin{cases} v^i_j, & \text{if } rand \leq CR \text{ or } j = j_{\text{rand}} \\ x^i_j, & \text{otherwise} \end{cases} \quad (11)$$

where $j \in \{1, \ldots, n\}$ and *rand* is a uniformly random number from [0, 1]. The crossover rate $CR \in [0, 1]$ is a user-defined control parameter. j_{rand} is an integer randomly chosen from the set $S = \{1, \ldots, n\}$. After the application of these DE operators, a generated offspring might undergo, with a small probability, the polynomial mutation operator [44].

2) Reward Calculation: In a recent work on AOS for selecting DE mutation operators [32], a DE mutation operator is rewarded based on the fitness improvement of the offspring compared with its corresponding target vector. In MOEA/D, however, the generated offspring is not directly compared with its target vector. In fact, an offspring is compared with several solutions randomly selected from the mating pool P, and one offspring can replace up to n_r solutions in P. Different solutions correspond to different subproblems, and thus, different comparisons are based on different aggregations of objective functions. In this paper, if an offspring successfully replaces x^i, the DE mutation operator that generates it will receive the following reward:

$$\eta = \frac{g(x^i|\lambda^i, z^*) - g(y|\lambda^i, z^*)}{g(x^i|\lambda^i, z^*)} \quad (12)$$

where λ^i is the weight vector for subproblem i.

An operator *op* may receive several rewards due to its generated offspring y. We sum up all these reward values as its final reward FIR_{op}.

The pseudocode of the MOEA/D with FRRMAB, denoted as MOEA/D-FRRMAB, is given in Algorithm 3. It is worth pointing out that FRRMAB can be used within other MOEA/D variants in the same way.

V. EXPERIMENTAL STUDIES

Our experimental studies can be divided into six parts.

1) Section V-B investigates how good the performance of MOEA/D-FRRMAB is compared with three other recent MOEA/D variants, namely, MOEA/D-DE [45], MOEA/D-DRA [27], and ENS-MOEA/D [46]. The latter ensembles different neighborhood sizes by using a probability-based method.
2) Section V-C investigates what benefits can be obtained by the efficient use of a pool of operators in MOEA/D-FRRMAB. To this end, MOEA/D-FRRMAB is compared with four MOEA/D variants, each of which uses only one DE mutation operator from the operators pool.
3) Section V-D analyzes what advantages FRRMAB has over the other operator selection mechanisms. To this end, it compares MOEA/D-FRRMAB with MOEA/D variants using other AOS methods.
4) To gain a deeper understanding about the proposed FRRMAB, the dynamics of its operator selection process are experimentally studied in Section V-E.
5) The proposed FRRMAB has several control parameters. Section V-F conducts a sensitivity analysis on some important ones.
6) To study the performance of our proposed algorithm on many-objective optimization problems, Section V-G tests MOEA/D-FRRMAB on three five-objective test instances.

A. Experimental Settings

1) *Test instances*: Ten unconstrained MOP test instances (UF 1 to UF 10) with various characteristics proposed in [47] and three five-objective test instances (WFG 1, WFG 8, and WFG 9 from [48]) are used in our experimental studies. The UF instances have nonlinear PSs in the decision space, and their corresponding PFs are given in Fig. 3. The number of decision variables of the UF instances is set to 30; for the WFG instances it is set to 28, eight of which are position related, while the other 20 are distance related.

2) *Performance metrics*: As discussed in [49] and [50], no unary performance metric is able to give a comprehensive measure on the performance of an MOEA. In our experimental studies, we consider the following two widely used performance metrics.

   a) *Inverted generational distance (IGD) metric [51]*: Let P^* be a set of uniformly distributed Pareto optimal solutions in the objective space. Let S be the obtained approximation to the PF in the
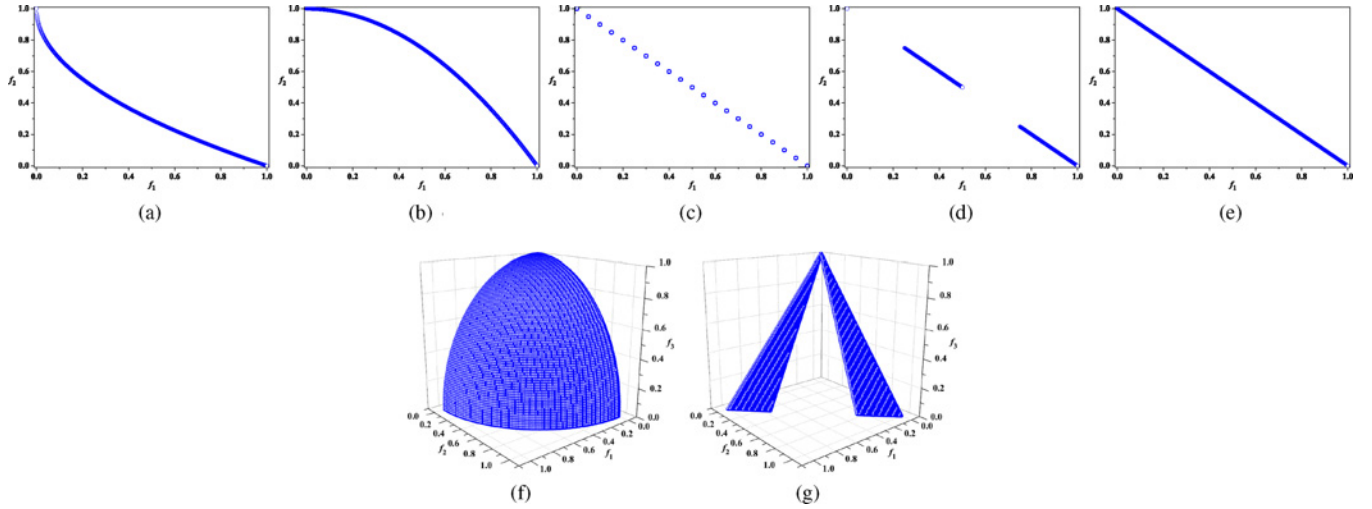
Fig. 3.   PF of the benchmark problems. (a) UF 1, UF 2, UF 3. (b) UF 4. (c) UF 5. (d) UF 6. (e) UF 7. (f) UF 8, UF 10. (g) UF 9.
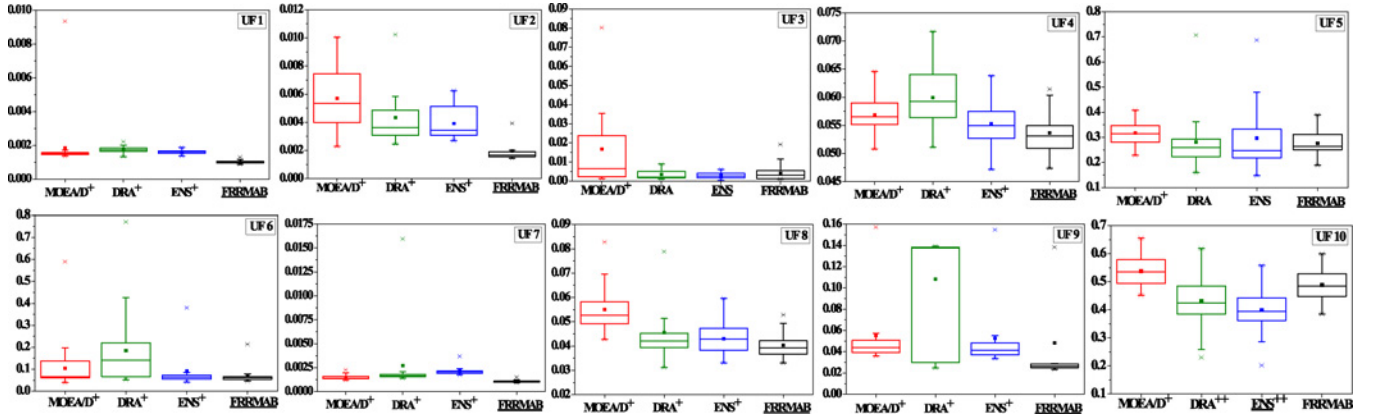

Fig. 4.   Box plots for the comparison of MOEA/D-DE, MOEA/D-DRA, ENS-MOEA/D, and MOEA/D-FRRMAB on *IGD*. Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean *IGD* value is underlined.

Fig. 5.   Box plots for the comparison of MOEA/D-DE, MOEA/D-DRA, ENS-MOEA/D, and MOEA/D-FRRMAB on I_H. Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean I_H value is underlined.
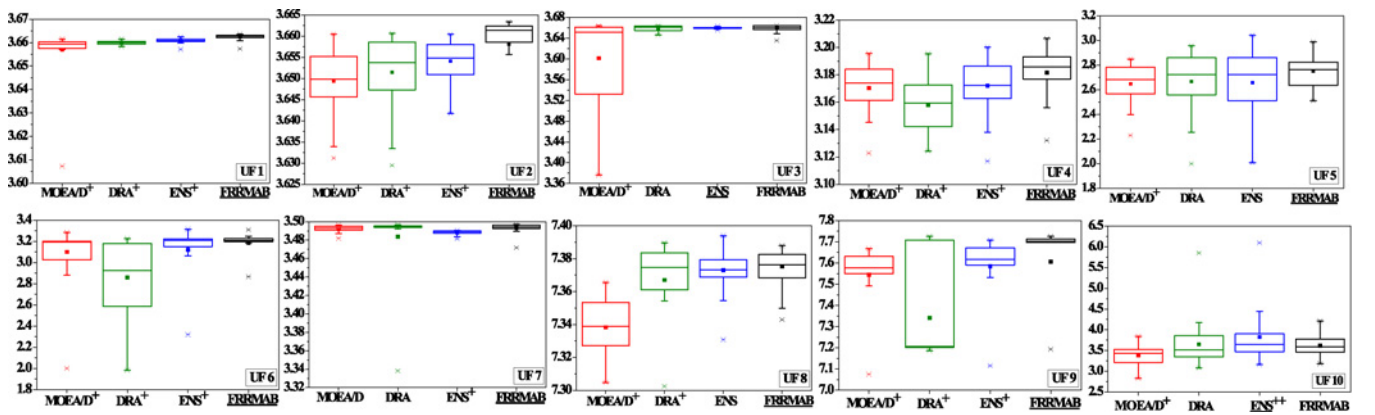

Algorithm 3: MOEA/D-FRRMAB

---

1 Initialize the population $\{x^1, \cdots, x^N\}$ and z^*;
2 set *gen* = 0;
3 **for** $i \leftarrow 1$ **to** N **do**
4 $\quad B(i) = \{i_1, \cdots, i_T\}$ where $\lambda^{i_1}, \cdots, \lambda^{i_T}$ are the T closest weight vectors to λ^i and set $\pi^i = 1$;
5 **end**
6 **while** *Stopping criterion is not satisfied* **do**
7 $\quad$ Let all the indices of the subproblems whose objectives are MOP individual objectives f_i form the initial I. By using 10-tournament selection based on π^i, select other $\lfloor N/5 \rfloor - m$ indices and add them to I.
8 $\quad$ **for** *each* $i \in I$ **do**
9 $\quad\quad$ $op = FRRMAB(FRR)$;
10 $\quad\quad$ **if** $uniform(0, 1) < \delta$ **then**
11 $\quad\quad\quad$ $P := B(i)$;
12 $\quad\quad$ **else**
13 $\quad\quad\quad$ $P :=$ the entire population;
14 $\quad\quad$ **end**
15 $\quad\quad$ Randomly select some solutions from P;
16 $\quad\quad$ Generate a candidate $\bar{y}$ by the application of the chosen DE mutation operator op over the selected solutions;
17 $\quad\quad$ Apply polynomial mutation operator on $\bar{y}$ with probability p_m, to produce the offspring y;
18 $\quad\quad$ Update the current reference point z^*;
19 $\quad\quad$ $c = 0$;
20 $\quad\quad$ **while** $c < n_r \| P! = \emptyset$ **do**
21 $\quad\quad\quad$ Randomly pick a solution x^j from P;
22 $\quad\quad\quad$ $\eta = \frac{g(x^j|\lambda^j,z^*) - g(y|\lambda^j,z^*)}{g(x^j|\lambda^j,z^*)}$;
23 $\quad\quad\quad$ **if** $\eta > 0$ **then**
24 $\quad\quad\quad\quad$ Replace x^j with y;
25 $\quad\quad\quad\quad$ Remove x^j from P;
26 $\quad\quad\quad\quad$ $FIR_{op} = FIR_{op} + \eta$;
27 $\quad\quad\quad$ **end**
28 $\quad\quad\quad$ c++;
29 $\quad\quad$ **end**
30 $\quad\quad$ SlidingWindow.*SetIndex*(*op*);
31 $\quad\quad$ SlidingWindow.*SetFIR*(FIR_{op});
32 $\quad\quad$ *CreditAssignment*(*Reward, SlidingWindow*);
33 $\quad\quad$ *Decay*(*Reward, FRR*);
34 $\quad\quad$ *gen*++;
35 $\quad\quad$ **if** $mod(gen, 50) == 0$ **then**
36 $\quad\quad\quad$ `// mod is the modulo operator`
37 $\quad\quad\quad$ update the utility π^i of each subproblem;
38 $\quad\quad$ **end**
39 $\quad$ **end**
40 **end**

---

objective space. The IGD value of S is defined as

$$IGD(S, P^*) = \frac{\sum_{x \in P^*} dist(x, S)}{|P^*|} \quad (13)$$

where $dist(x, S)$ is the minimal Euclidean distance between x and the points in S, and $|P^*|$ is the cardinality of P^*. In general, the IGD metric is able to give a reliable measure to the quality of S if $|P^*|$ is large enough. In our experiments, the number of points in P^* is 1000 for the biobjective test instances, 10 000 for the three-objective ones, and 5000 for the five-objective ones.

b) *Hypervolume indicator (I_H) [52]*: Let $y^* = (y_1^*, \ldots, y_m^*)$ be a antioptimal reference point in the objective space that is dominated by all Pareto optimal objective vectors. Let S be the obtained approximation to the PF in the objective space. Then, the I_H value of S (with regard to y^*) is the volume of the region dominated by S and bounded by y^*

$$I_H(P) = volume(\bigcup_{y \in S}[y_1(x), y_1^*] \times \ldots [y_m, y_m^*]). \quad (14)$$

In our experiments, $y^* = (2.0, 2.0)$ for biobjective test instances, $y^* = (2.0, 2.0, 2.0)$ for three-objective ones, and $y^* = (3.0, 5.0, 7.0, 9.0, 11.0)$ for five-objective cases.

Generally speaking, the lower the IGD value, the better S approximates the true PF. For I_H, the higher the better. Moreover, in order to have statistically sound conclusions, Wilcoxon's rank sum test at a 0.05 significance level is adopted to compare the significance of the differences between the solution sets obtained by two competing algorithms.

3) *Parameter settings*: The parameters of MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D are set according to [27], [45], and [46], respectively. All algorithms are implemented in Java,[1] except for ENS-MOEA/D, which is written in MATLAB.[2] The parameter settings of MOEA/D-FRRMAB are as follows.

a) *Control parameters in DE and polynomial mutation*: Following [27] and [45], we set $CR = 1.0$ and $F = 0.5$ for the DE operators, and $\eta = 20$, $p_m = 1/n$ for the polynomial mutation operator. $K = 0.5$, as in [17].

b) *Population size*: $N = 600$ for biobjective test instances, 1000 for the three-objective ones, and 1200 for the five-objective cases.

c) *Number of independent runs and maximum number of function evaluations*: 30 independent runs are conducted for each algorithm on each test instance. The maximum number of function evaluations is fixed to be 300 000.

d) *The neighborhood size*: $T = 20$.

e) *Maximum number of solutions replaced by each new solution*: $n_r = 2$.

f) *Probability with regard to selecting P*: $\delta = 0.9$ as in [27].

g) *Control parameters in FRRMAB*: except for Section V-F, in which the robustness of these control parameters is analyzed, they are constantly set as follows.

  i) Scaling factor: $C = 5.0$.

TABLE I

STATISTICS OF PERFORMANCE COMPARISONS
WITH THE OTHER MOEA/D VARIANTS

		MOEA/D	DRA	ENS
IGD	−	10	7	7
	+	0	1	1
	≈	0	2	2
I_H	−	9	6	6
	+	0	0	1
	≈	1	4	3

"−," "+," and "≈" denote the number of times the performance of the corresponding algorithm is worse than, better than, and similar to that of the proposed MOEA/D-FRRMAB, respectively, according to the Wilcoxon rank sum test at a 0.05 significance level.

ii) Size of sliding window: $W = 0.5 \times N$.
iii) Decaying factor: $D = 1.0$.

B. MOEA/D-FRRMAB Versus State-of-the-Art MOEA/D Variants

We have first compared MOEA/D-FRRMAB with three MOEA/D variants, namely, MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D on UF 1 to UF 10. Figs. 4 and 5 present the box plots of the *IGD* and I_H metric values obtained from 30 independent runs. Table I provides the statistics summarizing these performance comparisons. In these figures and tables, MOEA/D-DE, MOEA/D-DRA, ENS-MOEA/D, and MOEA/D-FRRMAB are denoted as MOEA/D, DRA, ENS, and FRRMAB, respectively. These results clearly show that MOEA/D-FRRMAB is the best. Actually, it has obtained better results in 53 out of its 60 performance comparisons with other algorithms (one such comparison compares MOEA/D-FRRMAB with another algorithm on one instance based on *IGD* or I_H), being significantly better in 45 out of all these comparisons. It is worth noting that our proposed algorithm significantly outperforms the recently proposed MOEA/D variant, ENS-MOEA/D, in 13 out of 20 comparisons. Thus, we can conclude that the use of FRRMAB has successfully improved the performance of the baseline MOEA/D.

On UF 3 and UF 10, MOEA/D-FRRMAB has been beaten by ENS-MOEA/D and MOEA/D-DRA. Both UF 3 and UF 10 have many local PFs. This may imply that FRRMAB could not efficiently improve the global search ability. One may consider the search diversity in credit assignment as in [33], which will be a research issue for our further work.

The good performance presented by MOEA/D-FRRMAB does not come for free. Here, we also intend to compare the CPU time of the proposed algorithm to the baseline MOEA/D-DRA, MOEA/D-DE, and ENS-MOEA/D, respectively. The average CPU time out of 30 independent runs used by MOEA/D-FRRMAB and its competitors, on all the benchmark test problems, are given in Table II.[3] The lowest CPU time is highlighted in bold type with a gray background.

As can be seen, compared to MOEA/D-DRA and MOEA/D-DE, the FRRMAB module needs extra CPU time for updating the credit value of each operator and for making

[3]All experiments were run on a 2.3-GHz Intel Core i7 machine, with 16-GB 1600-MHz DDR3 of RAM memory, running Mac OS X 10.8.

TABLE II

AVERAGE CPU TIME (IN SECONDS) NEEDED BY MOEA/D-FRRMAB,
MOEA/D-DE, MOEA/D-DRA, AND ENS-MOEA/D

	FRRMAB	MOEA/D	DRA	ENS
UF 1	5.13	**3.66**	4.63	1563
UF 2	5.39	**3.81**	4.76	1655
UF 3	6.18	**4.35**	5.36	1666
UF 4	5.55	**3.91**	4.95	1621
UF 5	5.46	**3.82**	4.93	1634
UF 6	5.44	**3.90**	5.01	1613
UF 7	5.14	**3.53**	4.58	1601
UF 8	7.58	**5.09**	7.61	1782
UF 9	7.53	**5.22**	7.59	1756
UF 10	8.05	**5.60**	7.92	1799

TABLE III

STATISTICS OF COMPARISONS WITH MOEA/D VARIANTS
USING A SINGLE DE MUTATION OPERATOR

		Op. 1	Op. 2	Op. 3	Op. 4
IGD	−	7	8	5	9
	+	0	0	0	0
	≈	3	2	5	1
I_H	−	7	7	6	10
	+	0	0	0	0
	≈	3	3	4	0

"−," "+," and "≈" denote the number of times the performance of the corresponding algorithm is worse than, better than, and similar to that of the proposed MOEA/D-FRRMAB, respectively, according to the Wilcoxon rank sum test at a 0.05 significance level.

a decision on which operator to apply at every generation. More specifically, it spends about 2 more seconds on two-objective test instances, namely, UF 1 to UF 7, and an extra 3 s on three-objective test instances. This corresponds to around 40% of extra time, on average, for all the instances. However, it is worth noting that the CPU time of ENS-MOEA/D is much longer than the other MOEA/D variants. This is mainly due to the fact that it was implemented in MATLAB and its code has not been fully vectorized.

C. Operators Pool Versus a Single DE Mutation Operator

Now, we investigate what benefits can be obtained by efficiently using a pool of operators in MOEA/D-FRRMAB. For this purpose, MOEA/D-FRRMAB is compared with four MOEA/D variants, each of them using only one DE mutation operator from the operators pool. A variant using a DE mutation operator *i* is called Op. *i*. To have a fair comparison, all the parameter settings in Op. *i* are kept the same as those in MOEA/D-FRRMAB, except that MOEA/D-FRRMAB uses an operators pool while Op. *i* only uses DE mutation operator *i*.

From the experimental results in Figs. 6, 7, and Table III, it is evident that MOEA/D-FRRMAB is the best among all the five algorithms. Actually, MOEA/D-FRRMAB has produced the best results on almost all the test instances, except for UF 3 and UF 6, in which Op. 2 and Op. 3 perform better, respectively. This indicates that using FRRMAB to manage an operators pool is beneficial, compared with using a single operator. It is noticeable that there is no clear winner among the four MOEA/D variants using a single operator.
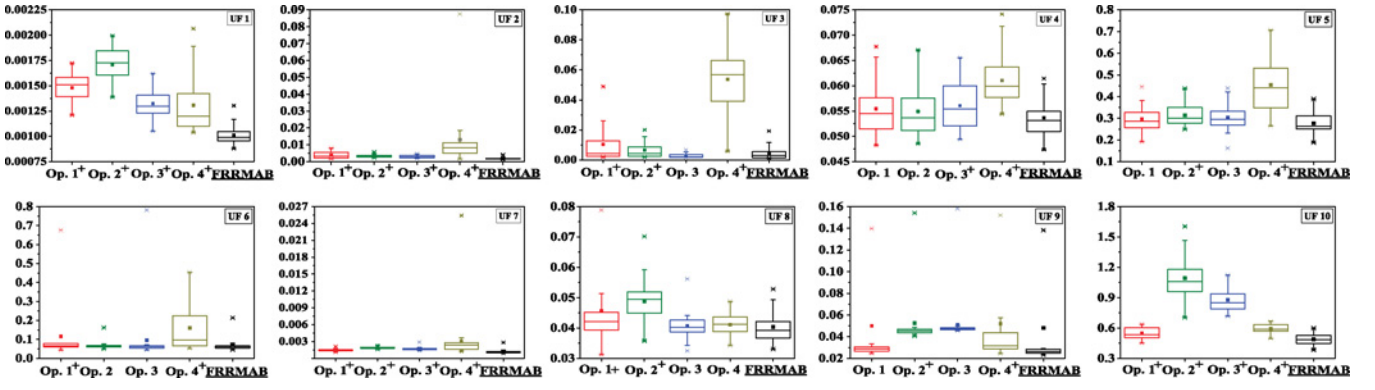
Fig. 6. Box plots for the comparison of MOEA/D-FRRMAB and MOEA/D variants with a single DE mutation operator on *IGD*. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of the static variants. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean *IGD* value is underlined.)
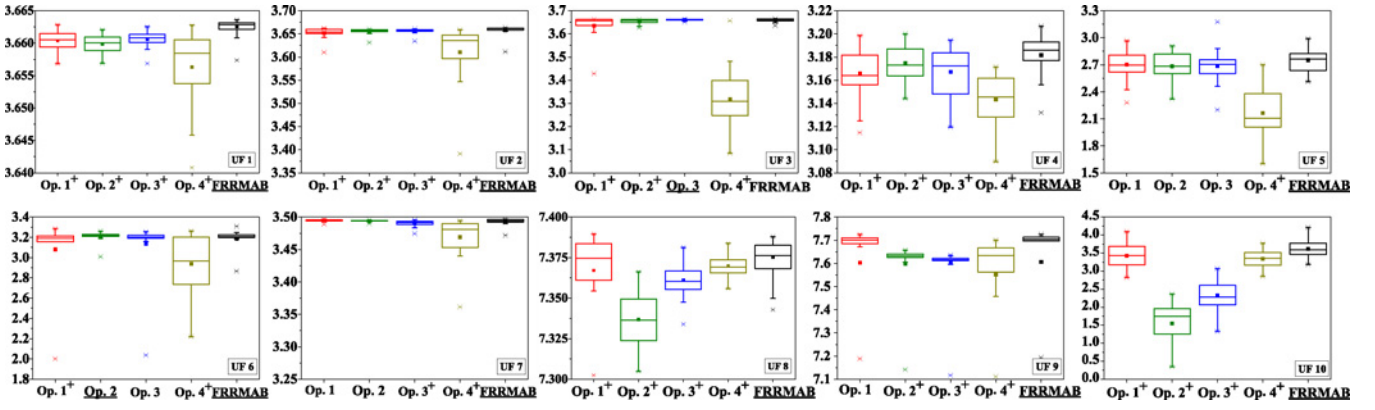

Fig. 7. Box plots for the comparison of MOEA/D-FRRMAB and MOEA/D variants with single DE mutation operator on I_H. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of the static variants. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean I_H value is underlined.)

D. FRRMAB Versus Other AOS Methods

As mentioned in Section II, some other AOS methods have been proposed before. Among them, PM [34], AP [35], and SRMAB [16] are three representative approaches. We have compared FRRMAB with these three AOS methods in the context of MOEA/D. In our implementations, we have replaced FRRMAB in MOEA/D-FRRMAB by each of these three methods, producing three algorithms for comparison. More specifically, PM and AP are coupled with the same credit assignment scheme proposed in Section III, while SRMAB uses its own credit assignment scheme [16]. The control parameters in these AOS methods are set as recommended by their corresponding original papers [35]. In particular, for PM and AP, the minimal operator selection probability $p_{\min} = 0.1$ and adaptation rate $\alpha = 0.8$. For AP, the learning rate β is set to 0.8. The control parameters in SRMAB are set to the same as in our proposed FRRMAB.

From the experimental results in Figs. 8, 9, and Table IV, it is clear that the proposed FRRMAB is the best in this comparison. It won on 73 out of 80 performance comparisons and performed significantly better on 63 comparisons. The second best competitor is the other bandit-based method, SRMAB, which won on UF 4, UF 6, and UF 10. The outperformance

TABLE IV
STATISTICS OF PERFORMANCE COMPARISONS
WITH THE OTHER AOS VARIANTS

		PM	AP	SRMAB	Uniform
	−	9	7	5	10
IGD	+	0	0	1	0
	≈	1	3	4	0
	−	9	9	4	10
I_H	+	0	0	1	0
	≈	1	1	5	0

"−," "+," and "≈" denote the number of times the performance of the corresponding algorithm is worse than, better than, and similar to that of the proposed MOEA/D-FRRMAB, respectively, according to the Wilcoxon rank sum test at a 0.05 significance level.

of SRMAB on these three test instances should be attributed to its comparison-based credit assignment scheme, which might be more suitable for these fitness landscapes. These results confirm that bandit-based methods are able to outperform others in the AOS context. The superiority of bandit should come from its theoretically sound way of balancing exploration and exploitation.

Another issue is regarding whether an intelligent AOS method is better than a random AOS one. To address this issue,
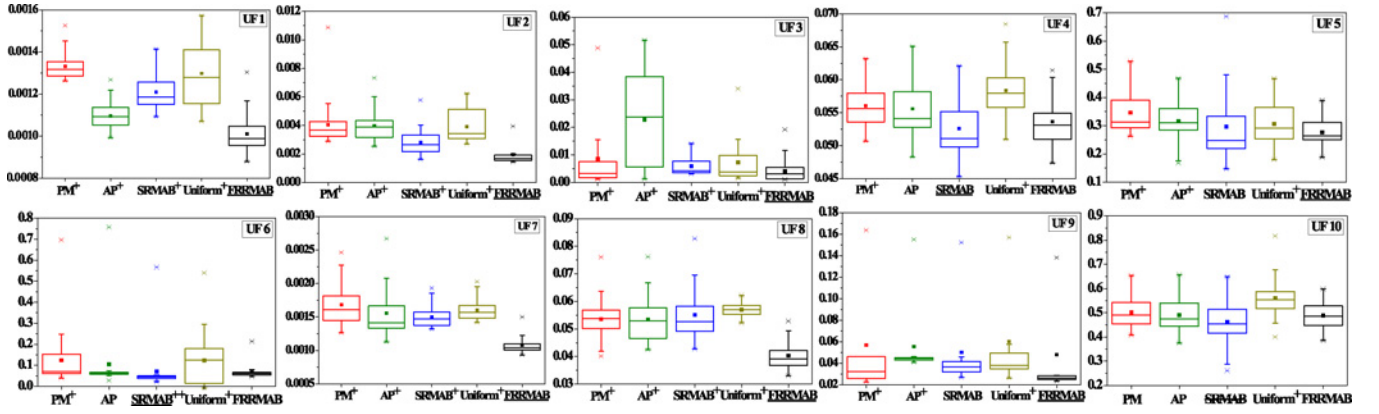
Fig. 8.   Box plots for the comparison of MOEA/D-FRRMAB with other AOS variants on *IGD*. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of the other AOS variants. "+"and "++"denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean *IGD* value is underlined.)
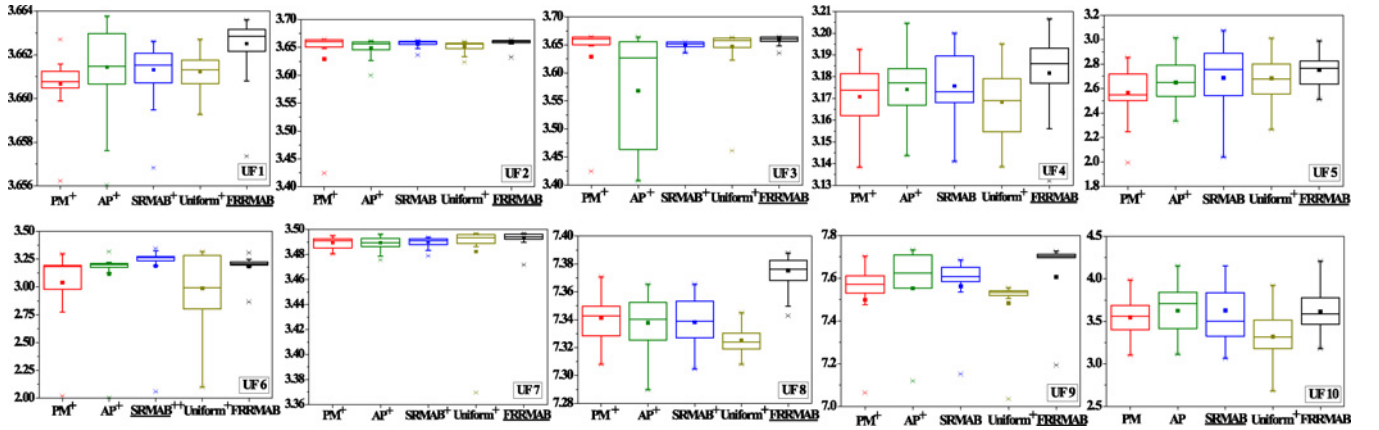

Fig. 9.   Box plots for the comparison of MOEA/D-FRRMAB with other AOS variants on I_H. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of the other AOS variants. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean I_H value is underlined.)

we have compared MOEA/D-FRRMAB with its counter-part using a uniformly random operator selection strategy (denoted as Uniform). From Figs. 8, 9, and Table IV, it is evident that the uniformly random selection strategy is the worst among all the methods. This finding confirms that it is worth exploring information about the recent performance of operators, and intelligent AOS methods can significantly improve the performance of the underlying algorithm.

E. Dynamics of Operator Selection

We have shown that FRRMAB is highly beneficial to MOEA/D performance. To have a deeper understanding of the behavior of FRRMAB, we now investigate how the usage number of each operator changes during the whole search process. For this purpose, we divide the whole search process (i.e., 300 000 function evaluations) into 50 consecutive phases, each of which consists of 6000 function evaluations. We have calculated the usage number of each operator during each phase and plotted the evaluations of these usage numbers in Fig. 10.

From Fig. 10, one can observe that no single operator can dominate over the whole search process on all the test instances. However, the search process for each test instance can be divided into a few stages, each of which is completely

dominated by a single operator. For example, on UF 1, the search process can be divided into three stages. Search phase 1 to phase 19 constitute stage 1, on which operator 2 dominates. Search phase 20 to phase 33 constitute stage 2, on which operator 1 dominates, while search phase 34 to phase 50 constitute stage 3, on which operator 4 dominates. These observations show that FRRMAB can use different operators on different search stages and that it can efficiently switch from one operator to another.

The four DE mutation operators in the operators pool have different characteristics. Operators 2 and 3 have two random-to-random terms, while the other two have only one; therefore, the former two are able to do more exploration than the latter ones. As previously discussed, intuitively, it is desirable to explore the search space in the initial stages of the optimization process, and perform more exploitation at later stages. This is empirically confirmed in Fig. 10: for most of the instances, operators 2 and/or 3 are preferred at early stages, while operators 1 and/or 4 are more frequently used at later ones. Figs. 6, 7, and Table III suggest that, overall, operator 4 performs worst among the four operators in UF test instances. One can also notice that this operator is rarely selected in FRRMAB. These observations imply that FRRMAB is able to select operators in a reasonable manner.
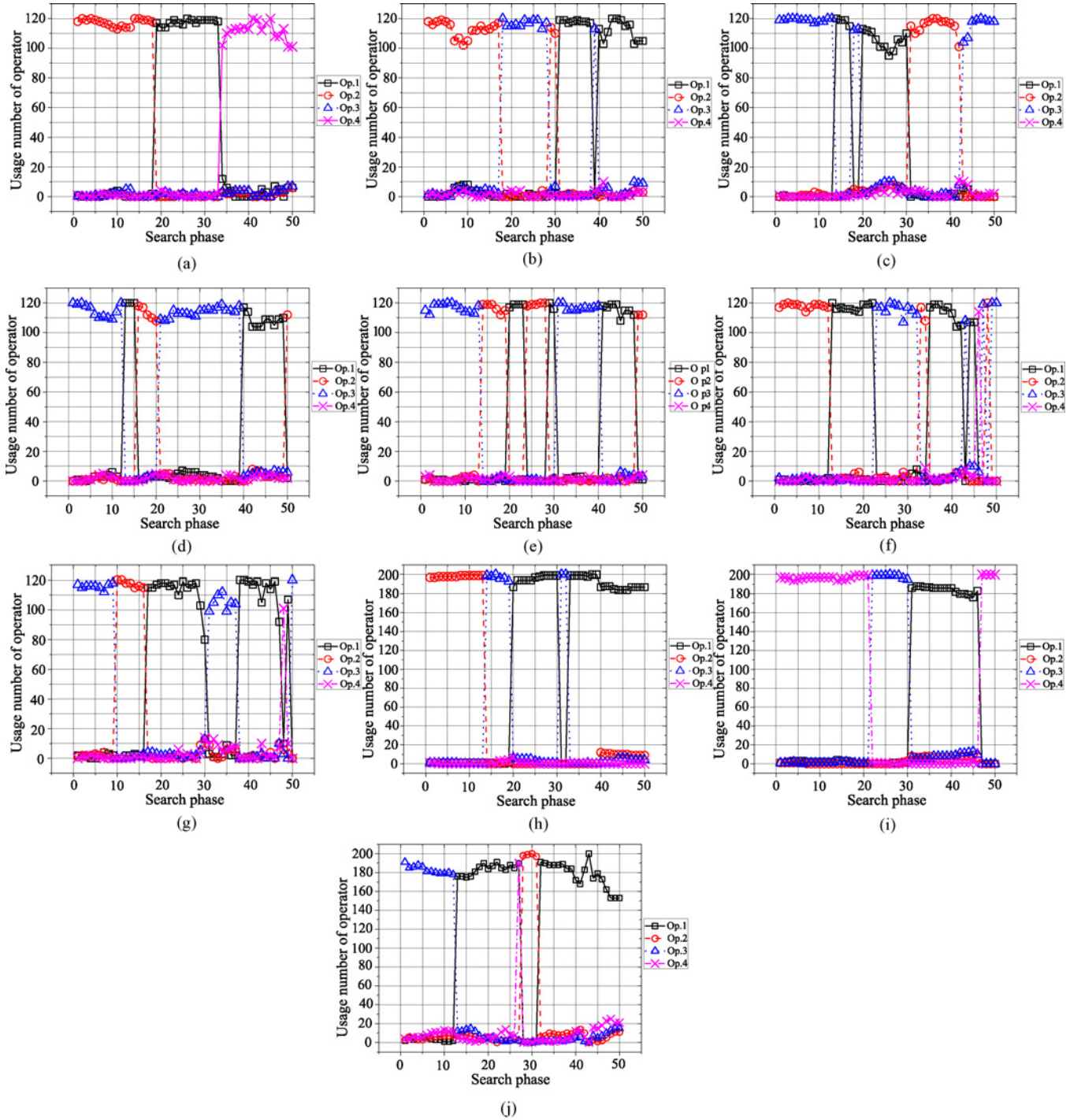
Fig. 10. Operator adaptation trajectories of MOEA/D-FRRMAB on different benchmark problems. (a) Operator adaptation trajectory on UF 1. (b) Operator adaptation trajectory on UF 2. (c) Operator adaptation trajectory on UF 3. (d) Operator adaptation trajectory on UF 4. (e) Operator adaptation trajectory on UF 5. (f) Operator adaptation trajectory on UF 6. (g) Operator adaptation trajectory on UF 7. (h) Operator adaptation trajectory on UF 8. (i) Operator adaptation trajectory on UF 9. (j) Operator adaptation trajectory on UF 10.
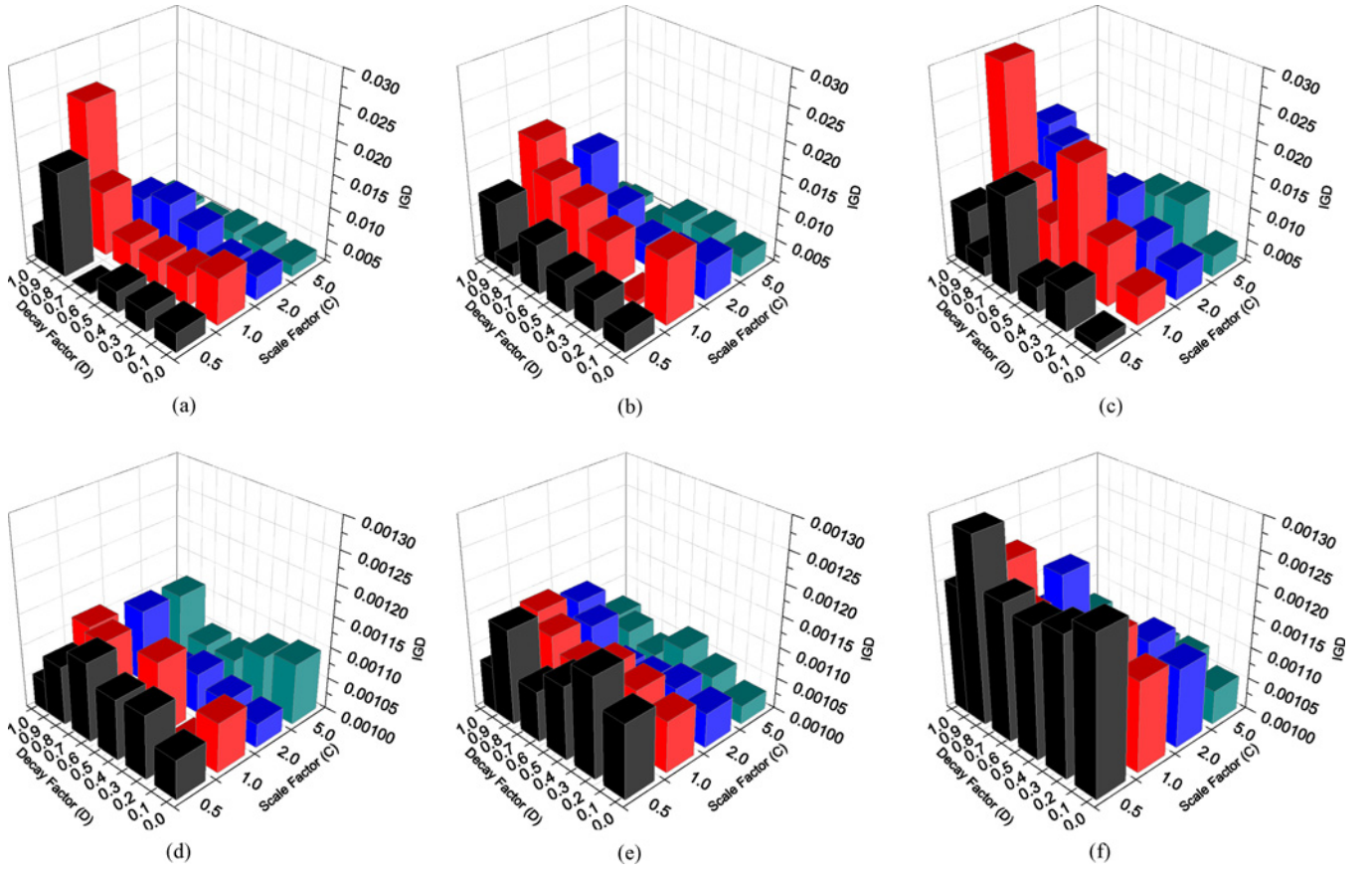
Fig. 11.   Median *IGD* metric values found by MOEA/D-FRRMAB with 72 different combinations of *C*, *D*, and *W* on UF3 (*a* to *c*) and UF7 (*d* to *f*). (a) $W = 0.5 \times NP$. (b) $W = NP$. (c) $W = 2 \times NP$. (d) $W = 0.5 \times NP$. (e) $W = NP$. (f) $W = 2 \times NP$.

F. Parameter Sensitivity Analysis in FRRMAB

There are three control parameters in FRRMAB.

1) The scaling factor *C*. As discussed in Section II-B2b, *C* balances exploration and exploitation in the operator selection process. A low value of *C* will favor the operators with the best previous performance, while a high value of *C* will prefer operators that were infrequently used before.
2) The size of the sliding window *W*. This parameter defines how many operator applications are used to calculate the credit values awarded to the operators.
3) The decaying factor *D*. *D* determines the priority level given to the best rewarded operator; a lower value of *D* will lead to a higher credit value for the top-ranked operators.

To study how MOEA/D-FRRMAB is sensitive to these three parameters, we have tried to cover a large range of values for each parameter. Four values were considered for scaling factor *C*: 0.5, 1.0, 2.0, and 5.0, three values for sliding window size *W*: $0.5 \times N$, *N*, and $2 \times N$, and six values for the decaying factor *D*: 0.1, 0.3, 0.5, 0.7, 0.9, and 1.0. We have taken UF 3 and UF 7 as test instances and tested all the 72 different parameter configurations. Thirty independent runs have been conducted for each configuration on each test instance. Figs. 11 and 12 show the median *IGD* and I_H metric values found with these 72 different configurations,

respectively. From these two figures, one can observe that different configurations can lead to different performances on both test instances. In order to further analyze this issue, for each performance metric and each test instance, we take the best parameter configuration in terms of the median metric value as a baseline, and compare it with each of the other 71 configurations using the Wilcoxon rank sum test at the 0.05 significance level. On UF 3, in terms of the *IGD* metric, the best configuration ($C = 0.5$, $D = 0.7$, $W = 0.5 \times N$) is not significantly different from 30 out of 71 configurations. In terms of I_H, the best configuration is not significantly different from 26 out of 71 configurations. On UF 7, in terms of the *IGD* metric, the best configuration ($C = 1.0$, $D = 0.3$, $W = 0.5 \times N$) is not significantly different from 28 out of 71 configurations. In terms of the I_H metric, the best configuration ($C = 0.5$, $D = 0.5$, $W = 0.5 \times N$) significantly outperforms only ten configurations. Since more than 50% configurations are worse than the best configuration in these cases, one should be careful when setting these parameters.

G. Performance on Many-Objective Problems

Most existing MOEAs are designed for and tested on problems with two or three objectives [53]. Their performances often deteriorate as the number of objectives increases. However, many real-world applications can involve more than three objectives. Many-objective optimization has
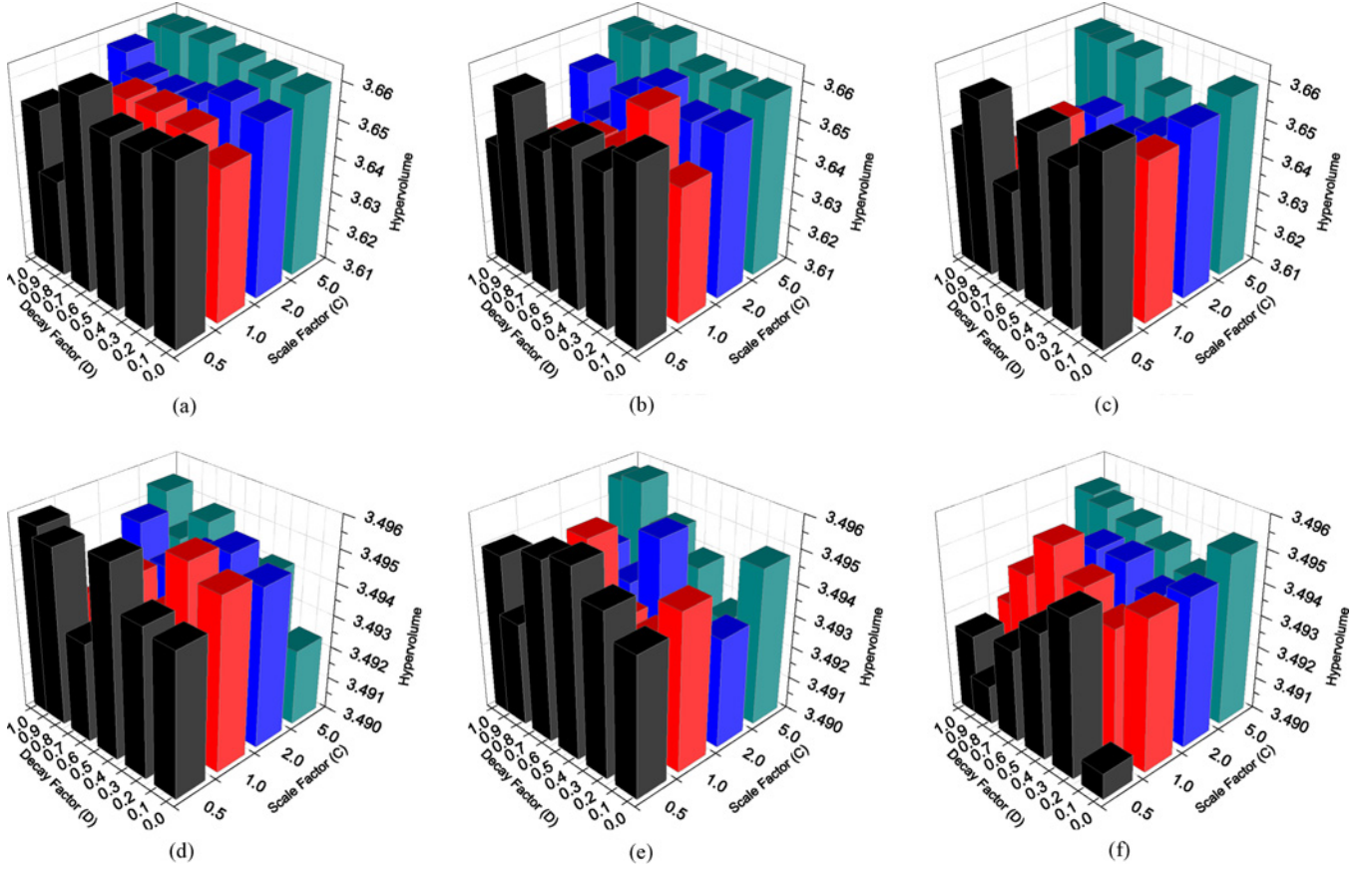
Fig. 12. Median I_H values found by MOEA/D-FRRMAB with 72 different combinations of C, D, and W on UF3 (*a* to *c*) and UF7 (*d* to *f*). (a) $W = 0.5 \times NP$. (b) $W = NP$. (c) $W = 2 \times NP$. (d) $W = 0.5 \times NP$. (e) $W = NP$. (f) $W = 2 \times NP$.

Fig. 13. Box plots for the comparison of MOEA/D-DE, MOEA/D-DRA, ENS-MOEA/D, and MOEA/D-FRRMAB on *IGD*. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean *IGD* metric value is underlined.)
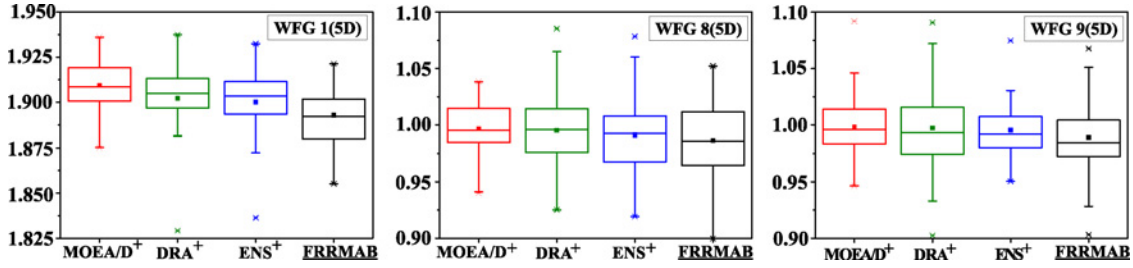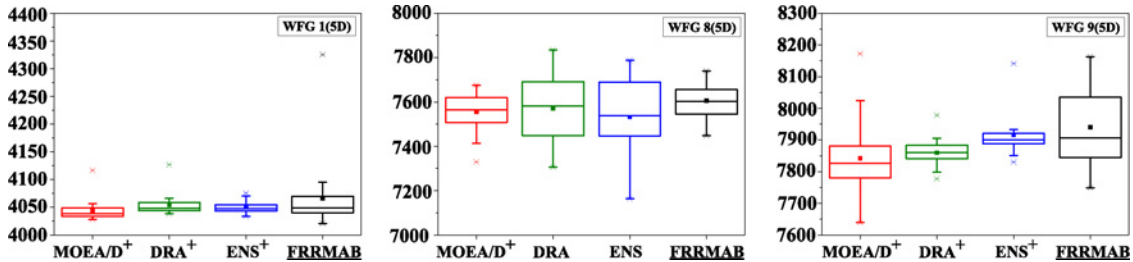

Fig. 14. Box plots for the comparison of MOEA/D-DE, MOEA/D-DRA, ENS-MOEA/D, and MOEA/D-FRRMAB on I_H. (Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-FRRMAB and each of MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D. "+" and "++" denote that the performance of the corresponding algorithm is significantly worse than or better than that of the proposed MOEA/D-FRRMAB, respectively. The algorithm with the best mean I_H metric value is underlined.)

TABLE V
STATISTICS OF PERFORMANCE COMPARISONS WITH THE OTHER
MOEA/D VARIANTS ON MANY-OBJECTIVE PROBLEMS

		MOEA/D	DRA	ENS
	−	3	3	3
IGD	+	0	0	0
	≈	0	0	0
	−	3	2	2
I_H	+	0	0	0
	≈	0	1	1

"−," "+," and "≈" denote the number of times the performance of the corresponding algorithm is worse than, better than, and similar to that of the proposed MOEA/D-FRRMAB, respectively, according to the Wilcoxon rank sum test at a 0.05 significance level.

attracted growing attention from the evolutionary computation community. Although the major purpose of this paper is not to address many-objective problems, we still want to investigate whether FRRMAB can improve the abilities of MOEA/D for dealing with many-objective problems. To this end, we have taken three five-objective instances from [48] (WFG 1, WFG 8, and WFG 9) to test algorithms' abilities for dealing with many objectives. These instances were used in the CEC 2007 MOEA contest [54]. We have compared our proposed MOEA/D-FRRMAB with MOEA/D-DE, MOEA/D-DRA, and ENS-MOEA/D on these three instances. The parameter settings are the same as in Section V-B, except for $N = 1200$, and both of IGD and I_H are employed as the performance metrics here. Wilcoxon's rank sum test at a 0.05 significance level is used to compare the statistical significance between every two algorithms.

From the experimental results in Figs. 13, 14, and Table V, we can conclude that MOEA/D-FRRMAB is the most competitive algorithm among all these MOEA/D variants. To be specific, WFG 1 can measure an MOEA for coping with bias from the experimental results shown in Figs. 13 and 14; we can clearly find the superior performance of our proposed MOEA/D-FRRMAB. Similarly to WFG 1, WFG 8 and WFG 9 are also featured with significant bias. In addition, the distance-related parameters are dependent on position-related parameters. The outperformance of MOEA/D-FRRMAB indicates that the AOS method can be of benefit for finding a promising set of distance parameters. As for WFG 9, although the dependency between position-related and distance-related parameters is not as difficult like that of WFG 8, it has multimodal and nonseparable reduction properties. The performance of MOEA/D-DE is slightly better than that of MOEA/D-DRA, but both perform worse than MOEA/D-FRRMAB. Based on these empirical results, we can conclude that AOS methods are worth considering in MOEAs for many-objective problems.

VI. CONCLUSION

In this paper, we proposed a new bandit-based AOS method, FRRMAB, to automatically select appropriate operators in an online manner. In FRRMAB, the fitness improvement rate caused by an operator is used to measure its quality. A method with a decaying mechanism was used to calculate the final reward value for each operator based on all the recent fitness improvement rates that were stored in a sliding window. Then, guided by these reward values, a simple bandit-based scheme selected the operator to be used next.

Since it was not easy to quantify the improvement caused by an operator in most Pareto dominance-based MOEAs, the use of AOS in these algorithms posed a big challenge. The decomposition nature of MOEA/D made it very suitable for using AOS. This paper studied several issues in incorporating our proposed FRRMAB into MOEA/D. We proposed a simple way for measuring the improvement caused by an operator for each subproblem within the MOEA/D framework. We took four commonly used DE mutation operators as candidate operators and conducted extensive experimental studies on some test instances. We showed that FRRMAB was robust, and its operator selection was reasonable in MOEA/D. The comparison experiments also indicated that FRRMAB can significantly improve the performance of MOEA/D.

Although FRRMAB was used to select operators for static MOPs in this paper, this approach could be generalized in a straightforward way for dealing with dynamic MOPs. It could also be useful to many other online selection issues in evolutionary computation, such as allocating computational efforts to different subproblems in decomposition-based multi-objective optimization, and selecting surrogate models in model-based EAs. Moreover, in addition to the decomposition-based MOEAs, a similar idea can also be applied to the indicator-based MOEAs, in which the fitness improvements caused by an operator can be measured with respect to the improvements evaluated by the underlying indicator. These are possible research issues for the follow-up to this paper.

The source code of our proposed MOEA/D-FRRMAB, the numerical values of the simulations, and more comprehensive parameter sensitivity studies can be found on K. Li's homepage: a nd Q. Zhang's homepage: .

REFERENCES

[1] K. Miettinen, *Nonlinear Multiobjective Optimization*, vol. 12. Dordrecht, The Netherlands: Kluwer Academic, 1999.

[2] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. 1st Int. Conf. Genet. Algorithms*, 1985, pp. 93–100.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[4] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Proc. Evol. Methods Des. Optimization Control Appl. Ind. Problems*, 2002, pp. 95–100.

[5] T. Chan, K. Man, K. Tang, and S. Kwong, "A jumping gene paradigm for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 2, pp. 143–159, Apr. 2008.

[6] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[7] K. Li, S. Kwong, R. Wang, K.-S. Tang, and K.-F. Man, "Learning paradigm based on jumping genes: A general framework for enhancing exploration in evolutionary multiobjective optimization," *Inform. Sci.*, vol. 226, pp. 1–22, Mar. 2013.

[8] K. Li, S. Kwong, J. Cao, M. Li, J. Zheng, and R. Shen, "Achieving balance between proximity and diversity in multiobjective evolutionary algorithm," *Inform. Sci.*, vol. 182, no. 1, pp. 220–242, 2012.

[9] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York: Wiley, 2001.

[10] C. A. Coello Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems* (Genetic and Evolutionary Computation), 2nd ed. New York: Springer, 2007.

[11] F. G. Lobo, C. F. Lima, and Z. Michalewicz, *Parameter Setting in Evolutionary Algorithms* (Studies in Computational Intelligence), F. Lobo, C. Lima, and Z. Michalewicz, Eds. Berlin/Heidelberg, Germany: Springer, 2007.

[12] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learning*, vol. 47, nos. 2–3, pp. 235–256, May 2002.

[13] L. Da Costa, A. Fialho, M. Schoenauer, and M. Sebag, "Adaptive operator selection with dynamic multiarmed bandits," in *Proc. Annu. Conf. Genet. Evol. Comput.*, Jul. 2008, pp. 913–920.

[14] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag, "Analyzing bandit-based adaptive operator selection mechanisms," *Ann. Math. Artif. Intell.*, vol. 60, nos. 1–2, pp. 25–64, Oct. 2010.

[15] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag, "Dynamic multiarmed bandits and extreme value-based rewards for adaptive operator selection in evolutionary algorithms," in *Proc. 3rd Int. Conf. Learning Intell. Optimization*, LNCS 5851. 2009, pp. 176–190.

[16] A. Fialho, M. Schoenauer, and M. Sebag, "Toward comparison-based adaptive operator selection," in *Proc. Annu. Conf. Genet. Evol. Comput.*, Jul. 2010, pp. 767–774.

[17] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[18] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[19] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[20] J. Zhang and A. C. Sanderson, "Self-adaptive multiobjective differential evolution with direction information provided by archived inferior solutions," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2008, pp. 2806–2815.

[21] V. L. Huang, A. K. Qin, P. N. Suganthan, and M. F. Tasgetiren, "Multiobjective optimization based on self-adaptive differential evolution algorithm," in *Proc. IEEE Congr. Evol. Comput.*, May. 2007, pp. 3601–3608.

[22] V. L. Huang, S. Z. Zhao, R. Mallipeddi, and P. N. Suganthan, "Multiobjective optimization using self-adaptive differential evolution algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2009, pp. 190–194.

[23] K. Li, A. Fialho, and S. Kwong, "Multiobjective differential evolution with adaptive control of parameters and operators," in *Proc. 5th Int. Conf. Learning Intell. Optimization (LION)*, LNCS. 2011, pp. 473–487.

[24] K. Li, S. Kwong, R. Wang, J. Cao, and I. J. Rudas, "Multiobjective differential evolution with self-navigation," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Oct. 2012, pp. 508–513.

[25] R. Mallipeddi, P. N. Suganthan, Q. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, 2011.

[26] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, Feb. 2011.

[27] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 203–208.

[28] A. Eiben, Z. Michalewicz, M. Schoenauer, and J. Smith, "Parameter control in evolutionary algorithms," in *Parameter Setting in Evolutionary Algorithms* (Studies in Computational Intelligence, vol. 54), F. Lobo, C. Lima, and Z. Michalewicz, Eds. Berlin/Heidelberg, Germany: Springer, 2007, pp. 19–46.

[29] L. Davis, "Adapting operator probabilities in genetic algorithms," in *Proc. ICGA*, 1989, pp. 61–69.

[30] W. Gong, A. Fialho, and Z. Cai, "Adaptive strategy selection in differential evolution," in *Proc. Annu. Conf. Genet. Evol. Comput.*, Jul. 2010, pp. 409–416.

[31] J. Whitacre, T. Pham, and R. Sarker, "Use of statistical outlier detection method in adaptive evolutionary algorithms," in *Proc. Annu. Conf. Genet. Evol. Comput.*, 2006, pp. 1345–1352.

[32] A. Fialho, R. Ros, M. Schoenauer, and M. Sebag, "Comparison-based adaptive strategy selection in differential evolution," in *Proc. 11th Int. Conf. Parallel Problem Solving From Nature*, 2010, pp. 194–203.

[33] J. Maturana, F. Lardeux, and F. Saubion, "Autonomous operator management for evolutionary algorithms," *J. Heuristics*, vol. 16, no. 6, pp. 881–909, 2010.

[34] D. E. Goldberg, "Probability matching, the magnitude of reinforcement, and classifier system bidding," *Mach. Learning*, vol. 5, no. 4, pp. 407–425, Oct. 1990.

[35] D. Thierens, "An adaptive pursuit strategy for allocating operator probabilities," in *Proc. Annu. Conf. Genet. Evol. Comput.*, 2005, pp. 1539–1546.

[36] A. Fialho, "Adaptive operator selection for optimization," Ph.D. dissertation, Comput. Sci. Dept., Univ. Paris-Sud XI, Orsay, France, Dec. 2010.

[37] J. Knowles and D. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evol. Comput.*, vol. 8, no. 2, pp. 149–172, Jun. 2000.

[38] Y. Mei, K. Tang, and X. Yao, "Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem," *IEEE Trans. Evol. Comput.*, vol. 15, no. 2, pp. 151–165, Apr. 2011.

[39] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Adaptation of scalarizing functions in MOEA/D: An adaptive scalarizing function-based multiobjective evolutionary algorithm," in *Proc. 5th Int. Conf. Evol. Multi-Criterion Optimization*, 2009, pp. 438–452.

[40] V. A. Shim, K. C. Tan, and C. Y. Cheong, "A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem," *IEEE Trans. Syst., Man, Cybern., C Appl. Rev.*, vol. 42, no. 5, pp. 682–691, Sep. 2012.

[41] S. Martinez and C. Coello, "A direct local search mechanism for decomposition-based multiobjective evolutionary algorithms," in *Proc. IEEE CEC*, Jun. 2012, pp. 1–8.

[42] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, to be published.

[43] R. Storn and K. Price, "Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimization*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[44] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," *Comput. Sci. Informatics*, vol. 26, no. 4, pp. 30–45, 1996.

[45] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 12, no. 2, pp. 284–302, Apr. 2009.

[46] S. Zhao, P. N. Suganthan, and Q. Zhang, "Decomposition based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes," *IEEE Trans. Evol. Comput.*, vol. 16, no. 3, pp. 442–446, Jun. 2012.

[47] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," Univ. Essex and Nanyang Technological University, Essex, U.K./Singapore, Tech. Rep. CES-487, 2008.

[48] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.

[49] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[50] O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello, "Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 4, pp. 504–522, Aug. 2012.

[51] P. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 174–188, Apr. 2003.

[52] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[53] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *Proc. 4th Int. Conf. Evol. Multi-Criterion Optimization*, 2007, pp. 742–756.

[54] V. L. Huang, A. K. Qin, K. Deb, E. Zitzler, P. N. Suganthan, J. J. Liang, M. Preuss, and S. Huband, "Problem definitions for performance assessment of multiobjective optimization algorithms," Nanyang Technol. Univ., Singapore, Tech. Rep., 2007.

Ke Li (S'10) was born in Hunan, China, in 1985. He received the B.Sc. and M.Sc. degrees in computer science and technology from Xiangtan University, Xiangtan, China, in 2007 and 2010, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong.

His current research interests include evolutionary multiobjective optimization, surrogate-assisted evolutionary algorithms, and statistical machine learning techniques.

Álvaro Fialho (M'13) was born in Campo Grande, Brazil, in 1982. He received the B.Sc. degree in computer engineering from Universidade Católica Dom Bosco, Campo Grande, in 2004, the M.Sc. degree in electrical engineering from the Escola Politécnica of Universidade de São Paulo, São Paulo, Brazil, in 2007, and the Ph.D. degree in computer science from Université Paris-Sud XI, Orsay, France, in 2010.

Since 2012, he has been a Computational Intelligence Researcher within GE Global Research, Rio de Janeiro, Brazil. His current research interests include extension and hybridization of computational intelligence tools for tackling real-world problems.

Sam Kwong (M'93–SM'04) received the B.S. and M.S. degrees in electrical engineering from the State University of New York at Buffalo, Buffalo, USA, in 1983, and the University of Waterloo, Waterloo, ON, Canada, in 1985, respectively, and the Ph.D. degree from the University of Hagen, Hagen, Germany, in 1996.

In 1990, he became a Lecturer with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong, where he is currently a Professor and the Head of the Department of Computer Science. His current research interests include evolutionary computation, video coding, and machine learning.

Dr. Kwong is an Associate Editor of the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and the *Information Sciences Journal*.

Qingfu Zhang (M'01–SM'06) received the B.Sc. degree in mathematics from Shanxi University, Shanxi, China, in 1984, and the M.Sc. degree in applied mathematics and the Ph.D. degree in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively.

He is currently a Professor with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong, and a Professor on leave from the School of Computer Science and Electronic Engineering, University of Essex, Colchester U.K. He holds two patents. His current research interests include evolutionary computation, optimization, neural networks, data analysis, and their applications.

Dr. Zhang is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS PART B. He is also an Editorial Board Member of three other international journals. MOEA/D, a multiobjective optimization algorithm developed in his group, won the Unconstrained Multiobjective Optimization Algorithm Competition at the Congress of Evolutionary Computation in 2009, and was a recipient of the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award.