Population-based Algorithm Portfolios with automated constituent algorithms selection

Tang, Ke; Peng, Fei; Chen, Guoliang; Yao, Xin

DOI:
10.1016/j.ins.2014.03.105

Information Sciences 279 (2014) 94–104

Contents lists available at ScienceDirect

Information Sciences

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

Population-based Algorithm Portfolios with automated constituent algorithms selection

Ke Tang [a,*], Fei Peng [a], Guoliang Chen [a], Xin Yao [a,b]

[a] USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications (UBRI), School of Computer Science and Technology, University of Science and Technology of China, China
[b] The Center of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, University of Birmingham, United Kingdom

ARTICLE INFO

Article history:
Received 11 May 2013
Received in revised form 23 March 2014
Accepted 26 March 2014
Available online 3 April 2014

Keywords:
Population-based Algorithm Portfolios
Algorithm subset selection
Evolutionary optimization
Global optimization

ABSTRACT

Population-based Algorithm Portfolios (PAP) is an appealing framework for integrating different Evolutionary Algorithms (EAs) to solve challenging numerical optimization problems. Particularly, PAP has shown significant advantages to single EAs when a number of problems need to be solved simultaneously. Previous investigation on PAP reveals that choosing appropriate constituent algorithms is crucial to the success of PAP. However, no method has been developed for this purpose. In this paper, an extended version of PAP, namely PAP based on Estimated Performance Matrix (EPM-PAP) is proposed. EPM-PAP is equipped with a novel constituent algorithms selection module, which is based on the EPM of each candidate EAs. Empirical studies demonstrate that the EPM-based selection method can successfully identify appropriate constituent EAs, and thus EPM-PAP outperformed all single EAs considered in this work.

1. Introduction

In the past decades, population-based algorithms, e.g., Evolutionary Algorithms (EAs), have been shown to be powerful optimization techniques for various real-world problems. However, since the performance of EAs may vary greatly from one problem to another, choosing the most appropriate EA is usually a non-trivial task. Algorithm selection will be even more challenging when a practitioner needs to address multiple problems simultaneously, as it might be too tedious to figure out the best algorithm for each single problem. Hence, an algorithm that performs generally well on all these problems is usually desirable. Motivated by this consideration, a general framework Population-based Algorithm Portfolios (PAP) has been proposed [17].

Basically, PAP can be viewed as a combination of multiple EAs. When solving a single problem, PAP "invests" computational time to its constituent EAs, runs them in parallel and maintains interactions between them via a simple migration scheme. In case of solving a set of problems, PAP utilizes the same settings for all problems considered, rather than trying to identify the best algorithm for each single problem. Empirical studies showed that, by integrating the advantages of different EAs into one framework, PAP not only provides practitioners a unified approach for solving his/her problem set, but also may lead to better performance than a single EA [17].

* Corresponding author. Tel.: +86 551 63600547.
E-mail address: ketang@ustc.edu.cn (K. Tang).

http://dx.doi.org/10.1016/j.ins.2014.03.105
0020-0255/

Despite the promising preliminary results, both empirical and theoretical analysis revealed that the performance of PAP is sensitive to its constituent EAs [17]. In some cases, integrating multiple EAs even led to inferior performance in comparison to a single EA. This phenomenon raises another research question. That is, how to select a few constituent algorithms for PAP from those off-the-shelf EAs. Although some analyses have been conducted in [17] to give guidelines along this direction, no approach has been developed.

This paper aims at further advancing the research on PAP by enabling it to select constituent algorithms automatically. An extended version of PAP, named PAP based on Estimated Performance Matrix (EPM-PAP), is proposed. EPM-PAP incorporates a novel and efficient approach for choosing constituent EAs. Given a problem set to address in a fixed time budget and a set of candidate EAs, EPM-PAP first utilizes a portion of the time budget to establish an EPM, based on which the constituent EAs are selected. After that, the constituent EAs will be used to construct a PAP instantiation to solve the problems with the remaining time budget.

The rest of this paper is organized as follows. Section 2 introduces PAP and the constituent algorithms selection problem associated to it. After that, details of EPM-PAP are described with discussion on related work in Section 3. Section 4 presents the experimental studies that compare EPM-PAP to other relevant approaches. Finally, conclusions and discussions are given in Section 5.

2. Preliminary background

2.1. Population-based Algorithm Portfolio

Generally speaking, the term algorithm portfolio refers to the idea of "investing" the entire time budget in multiple algorithms rather than allocating it to a single algorithm. It concerns fully utilizing the advantages of these algorithms in order to benefit a problem-solving episode. This idea has been explored for more than ten years, and early investigations are mainly dedicated to combinatorial problems [10,11]. Recently, PAP was proposed to generalize this idea to numerical optimization, with a focus on EAs [17]. More important, while the aim of most previous work is to boost performance on a single optimization problem, PAP mainly concerns the overall performance on a set of problems.

A typical instantiation of PAP consists of l EAs as its constituent algorithms, denoted as $\{a_i | i = 1, 2, \cdots, l\}$. Given a budget of computation time (e.g., a total number of fitness evaluations), a PAP instantiation allocates the time budget to its constituent EAs by holding m separate subpopulations and applying each constituent EA to one of them. During the evolution process, information sharing is implemented by migrating promising individuals among subpopulations. PAP terminates when the given fitness evaluations (FEs) are used up. The pseudo-code of PAP is shown in Fig. 1.

From the perspective of heuristic search, PAP can be viewed as a generic framework that utilizes different search biases. This concept has been intensively investigated in the evolutionary computation community since 1990s, but most early research focused on integrating multiple search operators (e.g., different mutation operators) into an EA, e.g., [1,13,14,18]. It is only recently that the integration of multiple EAs attracted more attention. For example, Vrugt et al. [25] proposed a multi-algorithm genetically adaptive method (AMALGAM), which combines multiple EAs together. Besides, the emerging research on Ensemble of Evolutionary Algorithms (EEAs) also employs the idea of combining multiple search techniques [16,30]. Although these efforts might be conceptually similar to PAP in the aspect that they also involve multiple algorithms/operators/parameters, like different ensemble approaches in machine learning, PAP was developed with a different aim. To be specific, the motivation of PAP is to achieve good overall performance on *a set of problem instances* and *reduce the risk of failing* to solve any single problem instance, while both AMALGAM and EEAs concern more about performance on sin-

```
migration_size: the number of emigrants in each migration
migration_interval: the generation numbers between two migrations
Procedure PAP
1  Initialization: Generate l initial subpopulations pop_1, ..., pop_l and evaluate all individuals in them
2  While stopping conditions are not satisfied do
3     For i=1 to l
4        Evolve pop_i using algorithm a_i
5     End for
6     If migration_interval generations is reached then activate the migration procedure
7        For i=1 to l
8           Identify migration_size best individuals from the l-1 subpopulations except pop_i
9           Duplicate the migration_size best individuals to form the set of emigrants
10          Set pop' as the union of pop_i and emigrants
11          Discard the migration_size worst individuals in pop'
12          Set pop_i to pop'
13       End for
14    End if
15 End while
```

Fig. 1. The general framework of a PAP instantiation.

Table 1
Main notations used in this paper.

A	the set of candidates EAs from which constituent EAs are selected
$\widetilde{A}$	a set of constituent EAs, or the corresponding PAP instantiation
m	number of candidate EAs
l	number of constituent EAs in a PAP instantiation
i	index of constituent EAs of an PAP instantiation
j	index of candidate EAs
a_i and a_j	a constituent EA and a candidate EA, respectively
F	the set of problems to solve
n	number of problems to solve
k	index of problems
f_k	a problem in set F
T	total time budget for solving a set of problems
EPM	estimated performance matrix

gle instances. The objective of PAP is more realistic than seeking an approach that performs the best for all problem instances, since such an approach may not even exists because of the no-free-lunch theorem [26].

2.2. Formulation of the constituent algorithms selection problem

Given a time budget T, PAP can be employed to solve a set of problems through two major steps. That is, determining the constituent EAs, and then applying the obtained PAP instantiation[1] to the problems of interest. Suppose $F = \{f_k | k = 1, 2, \ldots, n\}$ is a set of problems to solve. Let $A = \{a_j | j = 1, 2, \ldots, m\}$ be a set of candidate EAs, from which a number of EAs will be selected as the constituent EAs of PAP. The optimal set of EAs for the PAP can be written as:

$$\widetilde{A}_{opt} = \arg\max_{\widetilde{A} \subseteq A} U(\widetilde{A}, F, T) \tag{1}$$

where $U(\widetilde{A}, F, T)$ measures the overall performance of a PAP instantiation $\widetilde{A}$ on F within T. Following [17], $U(\widetilde{A}, F, T)$ is defined based on the pair-wise comparison of PAP instantiations, as given in Eq. (2):

$$P(\widetilde{A} > \widetilde{A}'|F) = \frac{1}{n}\sum_{k=1}^{n} P(\widetilde{A}_k > \widetilde{A}'_k | f_k);\ f_k \in F \tag{2}$$

where $\widetilde{A}$ and $\widetilde{A}'$ are different subsets of A and represent the corresponding PAP instantiations. $\widetilde{A}_k > \widetilde{A}'_k$ means $\widetilde{A}$ outperforms $\widetilde{A}'$ on f_k (i.e., $\widetilde{A}$ obtained a better final solution than $\widetilde{A}'$). P denotes the probability of the event $\widetilde{A}_k > \widetilde{A}k'$, which can be estimated by running $\widetilde{A}$ and $\widetilde{A}'$ on f_k for multiple times. For each $\widetilde{A} \subseteq A$, $U(\widetilde{A}, F, T)$ can be calculated by summing up Eq. (2) over all $\widetilde{A}' \subseteq A$.

To summarize, the constituent algorithms selection problem aims to choose from A a number of algorithms a_i, with which a good PAP instantiation can be established using the framework given in Fig. 1. Moreover, from the viewpoint of practice, the time budget T is provided for solving the whole problem set F, which consists of both the time for constituent algorithms selection (a "set-up" procedure) and the time for executing the resultant PAP instantiation. Hence, the constituent algorithms need to be selected in an efficient way in order to reserve sufficient time for solving the problems. For the sake of clarity, the major notations used in this paper are listed in Table 1.

3. PAP based on estimated performance matrix

Although the constituent algorithms selection problems are defined in the form of a standard optimization problem in Eqs. (1) and (2), it cannot be solved trivially. To be specific, given m candidate EAs, there exist 2^m candidate constituent EAs subsets (and thus the same number of different PAP instantiations). Hence, calculating Eq. (2) for all pairs of PAP instantiations will be computationally prohibitive even when m is of moderate value. In this section, we first present a novel approach for constituent EAs selection, which is based on Estimated Performance Matrix (EPM) and does not require exhaustive comparison between PAP instantiations. After that, the EPM-PAP algorithm is summarized with a discussion on relevant work.

3.1. Estimated performance matrix

Let $A = \{a_j | j = 1, 2, \ldots, m\}$ and $F = \{f_k | k = 1, 2, \ldots, n\}$ be the set of candidate EAs and the set of problems to solve, suppose the total time consumed by the constituent selection procedure is t ($t < T$). The EPM is a matrix that records the performance of each candidate EA $a_j \in A$. For each a_j, the corresponding EPM, denoted by EPM_j, is an r-by-n matrix. This matrix can be

[1] Since a PAP instantiation is built on its constituent algorithms set, these two terms will be used interchangeably in the context of this paper.

obtained by running a_j on each of the n problems for r times. Each element of EPM_j is the objective value of the best solution that a_j obtained on a problem in a single run. Since each element of EPM_j is obtained with a small portion of T,[2] it can be viewed as a conservative estimate of the solution quality achieved by running a_j with T on the same problem. Although the accuracy of such an estimate is by no means guaranteed, it has been commonly employed in many other scenarios, e.g., fine-tuning the parameters of an EA [4], where the performance of an EA needs to be estimated.

3.2. EPM-PAP

The definition of EPM can be extended to PAP instantiations easily, i.e., by constructing an EPM for each candidate PAP instantiation. However, this straightforward extension suffers from the fact that the number of potential PAP instantiations increases exponentially with the number of candidate EAs. Consider the scenario that a time budget t is available for selecting the constituent algorithms of PAP from m candidate EAs. The average time available for constructing an EPM for each possible subset of candidate EAs will be much less than that for constructing an EPM for a single candidate EA. As a result, the EPMs in the former case will provide a much less accurate estimate on an algorithms' performance and may lead to the selection of an inappropriate constituent algorithms set. Hence, a more cost-effective way needs to be developed.

Recall that the basic idea of PAP is to combine different EAs to achieve superior performance than employing any single candidate EA. Hence, it is natural to expect that a candidate EA, say a_j, outperforms a good PAP instantiation with a small probability. Under the assumption that constituent EAs of a PAP instantiation is run independently, the probability that a PAP instantiation under-performs a_j, on problem f_k can be stated as Eq. (3):

$$R_{k,j} = \prod_{i=1}^{l}\left(1 - P_{i,j}^{k}\right) \tag{3}$$

where l is the number of constituent EAs of the PAP instantiation and $P_{i,j}^{k}$ denotes the probability that the ith constituent EA of the PAP instantiation outperforms a_j. In practice, as constituent EAs of a PAP instantiation interact with one another via migration, they are not truly independent. In this case, Eq. (3) does not hold and $R_{k,j}$ can hardly be written explicitly due to the difficulty of theoretically modeling the interactions between different constituent EAs. However, PAP only migrates good solutions among constituent algorithms. Such migrations, as demonstrated by previous study on PAP as well as on distributed EAs, can enhance the quality of the final solution obtained. Hence, it is highly likely that $R_{k,j}$ will decrease when introducing the migration scheme into PAP. In other words, Eq. (3) can be viewed as an upper bound (though not a rigorous one) of the true probability of the PAP instantiation being outperformed.

By averaging Eq. (3) over the candidate EA set $A = \{a_j | j = 1, 2, \cdots, m\}$ and the problem set $F = \{f_k | k = 1, 2, \ldots, n\}$, we can reformulate the selection of constituent EAs as a minimization problem with the objective function given in Eq. (4):

$$R = \frac{1}{mn}\sum_{j=1}^{m}\sum_{k=1}^{n}\prod_{i=1}^{l}\left(1 - P_{i,j}^{k}\right) \tag{4}$$

The minimization problem defined by Eq. (4) aims to identify a set of constituent EAs that, when employed to construct a PAP instantiation, are the least likely to be outperformed by any of the candidate EAs available for PAP. Suppose m EPMs have been obtained for m candidate EAs and Eq. (4) needs to be computed for a PAP instantiation that employ l constituent EAs. According to the definition of EPM, the k column of EPM_j consists of r elements that are the best solution quality obtained in r runs of a_j on f_k. Hence, $P_{i,j}^{k}$ can be calculated based on the kth column of the EPMs corresponding to a_i and a_j. First, pair-wise comparisons of the elements in the two columns are conducted (i.e., there are r^2 comparisons in total). Then, $P_{i,j}^{k}$ can be estimated by dividing the times that a solution of a_i beats a solution of a_j with r^2. By calculating $P_{i,j}^{k}$ for all constituent algorithms (a_i) and candidate EAs (a_j) on all the n problems, Eq. (4) can be obtained.

From the steps for calculating Eq. (4), it can be observed that the most important advantage of Eq. (4) is that it does not require estimating the performance of any PAP instantiation, but can be calculated solely based on the EPMs of the m candidate EAs. In other words, it avoids constructing EPM for each potential PAP instantiation. When the set of candidate EAs is of medium size (say 10), this advantage makes it possible to enumerate all potential PAP instantiations and identifying the best one according to Eq. (4). In case that the number of candidate EAs is huge and direct enumeration becomes prohibitive, Eq. (4) can still be integrated with some existing search method, say forward selection [12], to yield at least a sub-optimal set of constituent algorithms within an acceptable time period.

In addition to its computational advantages, Eq. (4) also bears some interesting interpretations. Consider the case of a PAP with two constituent EAs, say a_1 and a_2, we may find that

$$\begin{aligned} R &= \frac{1}{mn}\sum_{j=1}^{m}\sum_{k=1}^{n}\left(1 - P_{1,j}^{k}\right)\left(1 - P_{2,j}^{k}\right) \\ &= \frac{1}{m}\sum_{j=1}^{m}\left(1 + \frac{1}{n}\sum_{k=1}^{n}P_{1,j}^{k}P_{2,j}^{k} - \frac{1}{n}\sum_{n}^{1}P_{1,j}^{k} - \frac{1}{n}\sum_{n}^{1}P_{1,j}^{k}\right) = \frac{1}{m}\sum_{j=1}^{m}\left[\left(1 - \overline{P}_{1,j}\right)\left(1 - \overline{P}_{2,j}\right) + \frac{1}{n}\sum_{k=1}^{n}\left(P_{1,j}^{k} - \overline{P}_{1,j}\right)\left(P_{2,j}^{k} - \overline{P}_{2,j}\right)\right] \end{aligned} \tag{5}$$

[2] t/mnr if the time budget for selection procedure is allocated to each run evenly.

where $\overline{P}_{1,j} = \frac{1}{n}\sum_{k=1}^{n} P_{1,j}^k$ and $\overline{P}_{2,j} = \frac{1}{n}\sum_{k=1}^{n} P_{2,j}^k$. From Eq. (5), it can be observed that our selection method favors candidate EAs with large $\overline{P}_{1,j}$ and $\overline{P}_{2,j}$. That means, each constituent EA should be competitive in comparison to other candidate EAs. More important, the second term, i.e., $\frac{1}{n}\sum_{k=1}^{n}\left(P_{1,j}^k - \overline{P}_{1,j}\right)\left(P_{2,j}^k - \overline{P}_{2,j}\right)$ requires the two constituent EAs to be complementary. More precisely, the performance of a_1 on a problem is desired to be above its "average" performance over the problem set when the performance of a_2 is below its "average" performance and vice versa. Due to this property, our constituent EAs selection method intends to select candidate EAs that behave differently, and thus leads to overall good performance over the whole problem set.

With the above-described approach for selecting constituent EAs, EPM-PAP solves a set of problems following the major steps described below:

1. First, each candidate EA a_j is applied to each problem for r independent runs. The final population obtained in each run is stored. An EPM is constructed for each a_j based on the quality of the best solution it obtained in each run. This step consumes a part of the total time budget T (in terms of fitness evaluations).
2. All possible subset of A is enumerated and the corresponding R is calculated using Eq. (4) and the EPMs. The subset with the smallest R is selected as the constituent algorithms for PAP.
3. A PAP instantiation is constructed by embedding the selected constituent algorithms into the framework given in Fig. 1. When initializing the population for each constituent algorithm, the best solution obtained in Step 1 is first inserted into the population. Then, the other initial individuals are randomly picked from the r populations stored during Step 1.
4. Finally, the PAP instantiation is run on all the problems to solve until the remaining total time budget is used up.

3.3. Related work on algorithm selection

To the best of our knowledge, the constituent algorithms selection problem has not been investigated in the literature. Nevertheless, it is closely related to the algorithm selection problem, which has attracted a lot of attentions in the past few years. Hence, we review the related work on algorithm selection in this sub-section. It is interesting to note that almost all algorithm selection methods [4,9,15,16,19,25,30] could be adapted easily to select operators or parameters as well, although not every paper has stated this explicitly.

Generally speaking, algorithm selection aims to identify the best-performing algorithm from a set of candidate algorithms. It is thus different from the constituent algorithms selection problem, which aims to select multiple algorithms to form a PAP instantiation. Existing approaches for algorithm selection can be divided into two main categories, i.e., the so-called inter-problem methods and the intra-problem ones.

An inter-problem approach usually focuses on selecting algorithm for a given problem class. For example, statistical racing [4,29] is a general-purpose tool to find an algorithm that performs as well as possible on a problem class. First, a number of problem instances are sampled and used as the training instances. Then, candidate algorithms are evaluated on the training instances. The algorithms that perform poorly will be discarded sequentially as soon as statistically sufficient evidence is gathered against them. Some recently developed approaches, such as Meta-Learning techniques [9,24], Mapping method introduced in [23] and Empirical Hardness Models [15], also utilize a set of problem instances for training. By applying all candidate algorithms to the training instances, these approaches establish models to characterize the relationship between problem features and algorithms' performance. Given a new coming problem instance, the models are then employed to predict which candidate algorithm will perform the best on the new problem instance. Since all the above-mentioned approaches make use of a training set of problem instances, they implicitly assume that the knowledge (e.g., the best algorithm or the model built) obtained from the training instances can "generalize" well to other problem instances in the same class. Such an assumption holds only in case that the training instances are related to other problem instances to solve. As PAP does not assume any relationship between problems (or problem instances), the inter-problem approaches are not suitable for the constituent algorithms selection.

Differently from inter-problem approaches, a typical intra-problem method aims to select the best algorithm for a single problem instance rather than a problem class. A representative method in this category is the so-called "racing multiple algorithms on a single problem" approach proposed by Yuan and Gallagher [29], which is an extension of statistical racing. For a given problem, this approach first executes all the candidate algorithms on the problem and compares the different algorithms with a pre-defined statistical test. Candidate algorithms that perform significantly poorly (in statistical sense) will be eliminated. Then, the remaining candidate algorithms are applied to the problem again to further eliminate some candidate algorithms. This procedure is repeated until only one candidate is left or the time budget is used up. Another intra-problem method that is worthy of mention is the intra-problem Adaptive Online Time Allocation (intra-AOTA) approach [8]. As shown by its name, intra-AOTA does not directly select the best algorithm, but iteratively allocates the time budget to a set of candidate algorithms. To be specific, intra-AOTA divides the given time budget into several slots. Each slot corresponds to an iteration. At the first iteration, the time slot is allocated to all the candidate algorithms according to some prior distributions or rules. Then, the average fitness of solutions obtained by each algorithm on the problem is recorded. Then, a linear model that maps the time allocation scheme to expected performance improvements is built and employed to

determine the time allocation scheme for the next iteration. This procedure is repeated until the total time budget is used up. In the extreme case, intra-AOTA can also be used for algorithm selection by allocating all the time to a single candidate algorithm at each iteration. AMALGAM [25] and EEAs [16,30] also adopted a similar adaptation strategy as intra-AOTA. Since the motivation of PAP framework is to establish a combination of EAs for a set of diverse problems, while the intra-problem methods tend to select different algorithms for different problems, they are not directly applicable for constituent algorithms selection.

To summarize, although more or less related, previous work on algorithm selection do not address the constituent algorithms selection task. Hence, the existing approaches cannot be employed in the context of PAP directly.

4. Experimental studies

To evaluate the effectiveness of EPM-PAP, experimental studies have been carried out. Four existing EAs, including self-adaptive differential evolution with neighborhood search (SaNSDE) [27], particle swarm optimizer with inertia weight (wPSO) [19], generalized generation gap (G3) model with generic parent-centric recombination (PCX) operator (G3PCX) [6] and covariance matrix adaptation evolution strategy (CMA-ES) [2], were chosen as the candidate EAs. These candidate EAs can be used to implement 6 instantiations of PAP with two distinct constituent algorithms and 4 instantiations of PAP with three constituent algorithms. Accordingly, our experiments involved two variants of EPM-PAP, referred to as EPM-PAP-2 and EPM-PAP-3, respectively. In EPM-PAP-2, two constituent algorithms were selected to construct a PAP instantiation, while 3 constituent algorithms were selected for EPM-PAP-3. The purposes of our experimental studies are twofold:

(1) To assess whether EPM-PAP provides a competitive approach for solving a set of problems within a given time budget.
(2) To verify whether the constituent algorithm selection method proposed in this paper can truly identify the best constituent algorithms for constructing a PAP instantiation.

4.1. Problem set and compared approaches

All the experimental studies were conducted on 27 benchmark functions. The first 13 functions were selected from the classical benchmark functions used in [28], denoted as $f_1 \sim f_{13}$. The other 14 functions were selected from the benchmark functions of the special session on real-parameter optimization of the 2005 IEEE Congress on Evolutionary Computation (CEC2005) [21], denoted as $f_{cec1} \sim f_{cec14}$. These 27 functions span a diverse set of problem features, such as multi-modality, ruggedness, ill-conditioning, and interdependency. More details of these functions can be found in [21,28]. In our experiments, all the functions were solved in 30-dimensions.

In [17], it has been shown that the best PAP instantiation can outperform any single EA in the candidate set. However, this observation is obtained by implicitly assuming that the constituent algorithms of the PAP instantiation are selected in advance and does not consume any computational time. Such an assumption can hardly hold in practice. For EPM-PAP, the constituent algorithms selection procedure is integrated in the problem-solving process and consumes part of the computational time. In other words, after the constituent algorithms are chosen, the corresponding PAP instantiation will have less time to search for good solutions. Although this is a more realistic setting, the quality of the final solution obtained by EPM-PAP may not be as good as the best results presented in [17]. Therefore, we compare EPM-PAP with the four candidate EAs, i.e., SaNSDE, wPSO, CMA-ES and G3PCX, in our experiments to verify whether EPM-PAP is still advantageous to single EAs. Furthermore, EPM-PAP is also compared to the "racing multiple algorithms on a single problem" approach and the intra-AOTA approach introduced in Section 3.3. This comparison is to answer a more general research question. That is, given a number of candidate EAs and a fixed time budget for solving a set of problems, is EPM-PAP a promising approach that performs overall well on the problem set? Although the racing approach and intra-AOTA approach cannot be employed to select constituent algorithms for PAP, they can be utilized in the above scenario as well and were hence involved. Concretely, both the approaches can be employed by addressing the problems one by one. When solving each problem, the intra-AOTA approach assigns computational time dynamically to the above-mentioned 4 candidate EAs, while the racing approach first identifies the best candidate EA for the problem and then runs the best EA with the remaining time budget. In our experiments, the racing approach employs the Friedman two-way analysis of variance by ranks [5,20] with significance level 0.05 as its statistical test, and will be referred to as F-Race hereafter.

4.2. Experimental settings

All the algorithms were compared under three different settings of total time budget, i.e., T_1 = 27*4e05 FEs, T_2 = 27*8e05 FEs and T_3 = 27*1.2e06 FEs. Accordingly, the computational time for solving each benchmark problem was set to T_1 = 4e05 FEs, T_2 = 8e05 FEs and T_3 = 1.2e06 FEs, respectively.

EPM-PAP consists of three types of parameters. That is, the parameters associated with the construction of EPMs, the migration parameters (i.e., *migration_size* and *migration_interval*), and the control parameters of the constituent algorithms. To obtain the EPMs for candidate EAs, EPM-PAP requires running each candidate EA on each problem for r independent runs. It should be noted that the construction of EPMs is a part EPM-PAP. Hence, the FEs consumed for constructing EPMs are considered in the total FEs used by EPM-PAP. That is, part of the above-mentioned computational time was actually used for

algorithm subset selection, and the remaining time was used to search for solutions to each problem. In our experiments, r was set to 8. For the three settings of total time budget, each independent run is assigned with 25,000, 50,000 and 75,000 FEs, respectively. The migration parameters *migration_size* and *migration_interval* were set to 1 and *MAX_GEN*/20, respectively. Here, *MAX_GEN* stands for the maximum generation number of the PAP instantiation.

To make a fair comparison, the control parameters of each candidate EA were not fine-tuned to favor the PAP framework, but were directly set as suggested in the original publications. Concretely, we used all the parameter settings suggested in [27] when implementing SaNSDE. According to [19], a linearly decreasing inertia weight over the course of the search is employed in wPSO. The two coefficients of wPSO were both set to 1.49445. For G3PCX and CMA-ES, we simply used the source code of G3PCX and CMA-ES provided by their inventors (the codes are available online) and the parameters were set according to [2,3,6]. There exist a few variants of PCX operator. As suggested in [19], the variant that employs the best individual in the population as the main parent for generating offspring was used. Furthermore, restart strategy has been used in G3PCX and CMA-ES to prevent them from terminating before the total time budget was used up. To make a fair comparison, the settings of control parameters of all four candidate EAs remained unchanged throughout all experiments.

Similar to EPM-PAP, F-Race also requires running the candidate EAs on each benchmark problem for several times. The time consumed by an algorithm on a problem for a single independent run was set to 10,000 FEs, 20,000 FEs, 30,000 FEs, in accordance with T_1, T_2 and T_3, respectively. For intra-AOTA, the size of the shifting window used for the regression procedure was set to 4. The time consumed in each cycle ΔT was set to 10,000 FEs, 20,000 FEs, 30,000 FEs, in accordance with T_1, T_2 and T_3, respectively. Furthermore, intra-AOTA was initialized to first allocate the time equally among all the candidate algorithms.

4.3. Comparison between EPM-PAP and the other approaches

All the results presented in this paper were obtained by executing 25 independent runs for each compared approaches, i.e., EPM-PAP2, EPM-PAP3, SaNSDE, wPSO, CMA-ES, G3PCX, F-Race and intra-AOTA, on the benchmark problems. These approaches are compared from the perspective of solution quality, i.e., the quality of the best solution obtained by an approach in each run. Since all test functions used in this paper are minimization problems, the quality of a solution was measured by its function error value, and the function error value for the optimal solution is 0.

Non-parametric multiple-comparison statistical test described in [7] has been conducted to analyze the performance of all the compared algorithms. Specifically, two sets of tests have been carried out. In the first set of statistical tests, the average performance (i.e., solution quality) of the algorithms on each function was firstly obtained. Then, statistical tests were conducted to check whether EPM-PAP performed significantly different from the other algorithms on the all the 27 test functions in general. The Friedman test with significance level 0.05 was first employed to compare the general performance of all the algorithms. If the null hypothesis that all the methods performed the same was rejected, the Nemenyi's test with significance level 0.05 was employed to identify the pairwise differences between algorithms. The rankings of each algorithm, the Friedman p-value, and the Nemenyi's critical differences between rankings are presented in Tables 2 and 3. It can be observed that the Friedman test clearly indicated significant differences between the compared algorithms. Although the Nemenyi's critical difference showed that neither EPM-PAP-2 nor EPM-PAP-3 significantly outperformed all the other algorithms, EPM-PAP achieved the highest ranking in most of the cases. The only exception is the case for EPM-PAP-2 with T_1, where EPM-PAP is slightly worse than Intra-AOTA. This observation is consistent with our expectation that EPM-PAP's performance will degrade when less time budget is allocated for algorithm subset selection. Nevertheless, the superiority of PAP

Table 2
Comparison between EPM-PAP-2 (with different time budgets) and the other algorithms on the 27 test functions (Friedman test followed by the Nemenyi's test was employed). The test was conducted on the whole test function set based on the average performance of each algorithm. A smaller ranking indicates better performance.

Time budget	EPM-PAP-2	SaNSDE	wPSO	G3PCX	CMA-ES	F-Race	Intra-AOTA	p-Value	Critical difference
T_1	3.185	3.370	5.130	5.926	3.722	3.537	3.130	4.59E−09	1.733
T_2	3.278	3.407	4.907	5.889	3.648	3.481	3.389	3.44E−08	1.733
T_3	3.204	3.389	4.944	5.759	3.852	3.407	3.444	9.05E−08	1.733

Table 3
Comparison between EPM-PAP-3 (with different time budgets) and the other algorithms on the 27 test functions (Friedman test followed by the Nemenyi's test was employed). The test was conducted on the whole test function set based on the average performance of each algorithm. A smaller ranking indicates better performance.

Time budget	EPM-PAP-3	SaNSDE	wPSO	G3PCX	CMA-ES	F-Race	Intra-AOTA	p-Value	Critical difference
T_1	3.019	3.407	5.148	5.926	3.741	3.574	3.185	2.57E−09	1.733
T_2	3.296	3.463	4.907	5.889	3.648	3.426	3.370	3.70E−08	1.733
T_3	2.870	3.537	5.019	5.759	3.870	3.426	3.519	1.01E−08	1.733

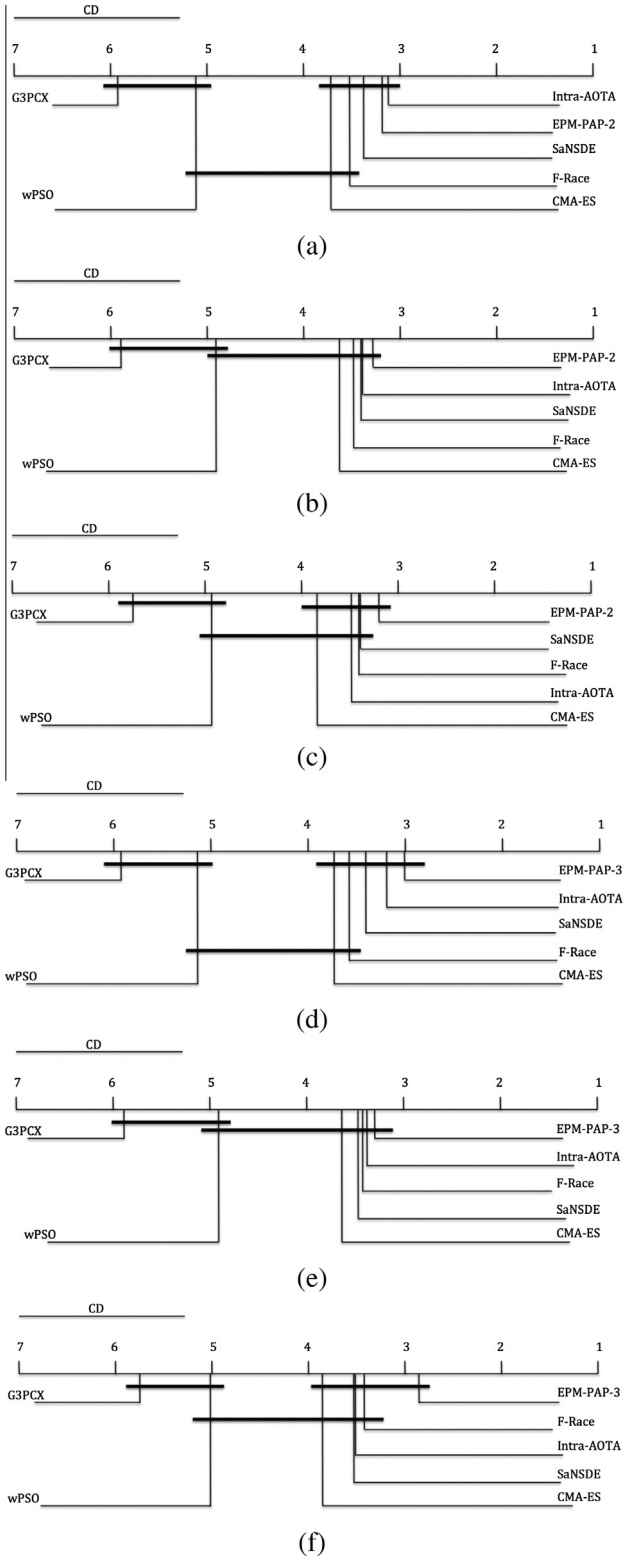

Fig. 2. Critical diagrams of the first set of multiple-comparison statistical tests. Figures (a–c) corresponds to EPM-PAP-2, with time budget T_1, T_2 and T_3, respectively. Figures (d–f) corresponds to EPM-PAP-2, with time budget T_1, T_2 and T_3, respectively.

over the 4 candidate EAs clearly demonstrates that effectiveness of the constituent algorithms selection procedure. In addition, the critical diagrams for each case of EPM-PAP (i.e., with different time budgets and different number of constituent algorithms) are presented in Fig. 2.

In the second set of tests, we separately compared all the algorithms' performance on each test function. That means, the test was repeated for 27 times for all the 27 test functions. Such tests allowed us to evaluate the performance of EPM-PAP from a different perspective. For each test function, the test is not conducted based on the performance of each algorithm in each run. Similar to the first set of tests, the Friedman test with significance level 0.05 was first employed to compare the general performance of all the algorithms. If the null hypothesis that all the methods performed the same was rejected, the Nemenyi's test with significance level 0.05 was employed as the post hoc test to identify the algorithms that performed the best on this function. For each compared algorithm, the number of functions on which it performed the best is given in Tables 4 and 5. It can be observed from the table that both EPM-PAP-2 and EPM-PAP-3 outperformed all the compared algorithms for all three settings of total time budget. Therefore, the results of this set of tests further confirmed EPM-PAP's advantages over the compared algorithms.

In addition to the non-parametric multiple-comparison statistical test that supports the overall superior performance of EPM-PAP, two-sided Wilcoxon rank-sum tests with significance level 0.05 have also been conducted to compare EPM-PAP-2 and EPM-PAP-3 with the other algorithms separately. Table 6 summarizes the results of the Wilcoxon tests over the 27 test functions. Here, the results are presented in form of win-draw-lose, standing for the numbers of functions on which EPM-PAP is superior, comparable (i.e., statistical insignificant), and inferior to the compared approach.

The results in Table 6 are generally consistent with those presented in Tables 2–5. It can be found that the advantage of EPM-PAP over wPSO and G3PCX is quite obvious. When compared to SaNSDE and CMA-ES, the advantage of EPM-PAP is relatively marginal, especially in the case of EPM-PAP-2 (i.e., only two candidate EAs were selected for constructing the PAP instantiation). The reason is that a PAP instantiation will behave similarly to its best constituent algorithm when it only consists of 2 constituent algorithms [17]. In addition, SaNSDE and CMA-ES are in general more powerful than wPSO and G3PCX, and thus are more likely to be selected in our experiments. Hence, EPM-PAP-2 may perform similarly to these two algorithms.

Second, both EPM-PAP-2 and EPM-PAP-3 performed better than F-Race for all three settings of total time budget. It is interesting to mention that F-Race and EPM-PAP represent two distinct strategies for solving a set of problems. Given a pool of candidate EAs, the former tries to find the best algorithm for each problem. The latter, on the other hand, resorts to identifying a combination of candidate EAs that can overall perform well on all the problems. Despite the popularity of racing approaches in the literature, the promising results of EPM-PAP suggest that the latter might be a better alternative.

Table 4
Comparison between EPM-PAP-2 and the other algorithms on the 27 test functions (Friedman test followed by the Nemenyi's test was employed). The number in each cell stands for the number of functions on which the corresponding algorithm performed the best.

Time budget	EPM-PAP-2	SaNSDE	wPSO	G3PCX	CMA-ES	F-Race	Intra-AOTA
T_1	25	20	9	6	18	20	20
T_2	25	21	11	5	17	19	21
T_3	26	21	11	5	16	20	20

Table 5
Comparison between EPM-PAP-3 and the other algorithms on the 27 test functions (Friedman test followed by the Nemenyi's test was employed). The number in each cell stands for the number of functions on which the corresponding algorithm performed the best.

Time budget	EPM-PAP-2	SaNSDE	wPSO	G3PCX	CMA-ES	F-Race	Intra-AOTA
T_1	24	20	9	6	19	20	20
T_2	24	20	11	5	17	19	20
T_3	26	21	11	5	17	19	19

Table 6
Comparison between EPM-PAP-2 and EPM-PAP-3 and all the other approaches (Two-sided Wilcoxon rank-sum tests with significance level 0.05 was employed). Results are presented in the form of WIN-DRAW-LOSE, standing for the numbers of functions on which the corresponding PAP instantiation is superior, comparable and worse than the compared algorithm.

	Time budget	SaNSDE	wPSO	G3PCX	CMA-ES	F-Race	Intra-AOTA
EPM-PAP-2	T_1	8-14-5	17-10-0	21-6-0	8-13-6	9-14-4	6-15-6
	T_2	7-14-6	16-10-1	20-7-0	9-14-4	7-15-5	5-18-4
	T_3	6-15-6	17-9-1	21-6-0	10-14-3	7-14-6	6-18-3
EPM-PAP-3	T_1	9-11-7	19-7-1	21-5-1	10-10-4	10-13-4	5-17-5
	T_2	8-17-2	17-9-1	20-7-0	9-12-6	9-12-6	5-20-2
	T_3	9-16-2	17-10-0	21-6-0	9-14-4	9-14-4	6-20-1

Table 7
The performance ranking of all instantiations of EPM-PAP-2 and EPM-PAP-3.

PAP	Rank	Time budget = T_1	Time budget = T_2	Time budget = T_3
With 2 constituent algorithms	1	SaNSDE + CMA-ES	SaNSDE + CMA-ES	SaNSDE + CMA-ES
	2	wPSO + CMA-ES	wPSO + CMA-ES	wPSO + CMA-ES
	3	SaNSDE + wPSO	SaNSDE + wPSO	SaNSDE + wPSO
	4	SaNSDE + G3PCX	SaNSDE + G3PCX	SaNSDE + G3PCX
	5	G3PCX + CMA-ES	G3PCX + CMA-ES	G3PCX + CMA-ES
	6	wPSO + G3PCX	wPSO + G3PCX	wPSO + G3PCX
wIth 3 constituent algorithms	1	SaNSDE + wPSO + CMA-ES	SaNSDE + wPSO + CMA-ES	SaNSDE + wPSO + CMA-ES
	2	SaNSDE + G3PCX + CMA-ES	SaNSDE + G3PCX + CMA-ES	SaNSDE + G3PCX + CMA-ES
	3	SaNSDE + wPSO + G3PCX	SaNSDE + wPSO + G3PCX	wPSO + G3PCX + CMA-ES
	4	wPSO + G3PCX + CMA-ES	wPSO + G3PCX + CMA-ES	SaNSDE + wPSO + G3PCX

Table 8
Accuracy of the constituent algorithms selection method.

	Time budget	SR_1 (%)	SR_2 (%)
EPM-PAP-2	T_1	40	88
	T_2	56	100
	T_3	72	100
EPM-PAP-3	T_1	16	84
	T_2	36	88
	T_3	56	100

Third, EPM-PAP performed comparably to intra-AOTA when the total time budget was set to T_1. However, the superiority of EPM-PAP-2 and EPM-PAP-3 became more and more visible when the total time budget was increased. The reason for this phenomenon is: when the total time budget is relatively small, the time available for constructing EPMs might be too limited to gain accurate estimations about the performance of candidate EAs, and thus inappropriate constituent algorithms may be selected. This phenomenon will be further elaborated in the next sub-section.

4.4. Analysis of the constituent algorithms selection method

As the main contribution of this work lies in the constituent algorithms selection method, it is important to verify whether the proposed method indeed selected the optimal or appropriate constituent algorithms for PAP. For this purpose, we executed all the 6 PAP instantiations with 2 constituent algorithms and 4 PAP instantiations with 3 constituent algorithms to the 27 benchmark functions, and ranked them according to the solution quality obtained. The detailed ranks are presented in Table 7, where the better PAP instantiation corresponds to a smaller rank.

Based on the ranks provided in Table 7, we recorded the two statistics SR_1 and SR_2 given in Eqs. (6) and (7) for EPM-PAP-2 and EPM-PAP-3, respectively.

$$SR_1 = \frac{N_{suc}}{25} \tag{6}$$

$$SR_2 = \frac{N_{suc2}}{25} \tag{7}$$

where N_{suc} is the number of runs (out of 25 runs) that the proposed selection method successfully selected the best constituent algorithms. N_{suc2} denotes the number of runs that the selected constituent algorithms set ranked at least 2 (i.e., either 1 or 2). According this definition, SR_1 measures the probability that the proposed selection method identifies the best constituent algorithms set, and SR_2 measures the probability that the selection method identifies good constituent algorithms set.

The SR_1 and SR_2 of EPM-PAP-2 and EPM-PAP-3 on three settings of total time budget are presented in Table 8. It can be found that, in 3 of the total 6 cases, the proposed selection methods achieved SR_1 higher than 50%. Specifically, SR_1 increased with the total time budget, which demonstrates that the proposed selection method is capable of identifying the best constituent algorithms when sufficient time if provided to construct the EPMs. More important, it can be found that SR_2 is larger than 80% in all 6 cases. Hence, even if EPM-PAP failed to identify the best constituent algorithms, the sub-optimal constituent algorithms can be selected, and lead to superior performance to the compared candidate EAs.

5. Conclusion and discussions

In this paper, a novel approach called EPM-PAP is proposed for solving optimization problems. Compared with the previous version of PAP, whose constituent algorithms need to be defined by human experts, EPM-PAP incorporates a method

for automatically choosing constituent algorithms. Empirical studies on numerical optimization problems demonstrate that the EPM-based selection method is capable of identifying appropriate constituent EAs within a short period. As a result, EPM-PAP outperformed all the EAs involved in the experiments. Furthermore, since EPM-PAP concerns solving multiple problems within a time budget, it is also compared to F-Race and intra-AOTA, which are applicable in this scenario. Experimental results clearly show the advantages of EPM-PAP over these two approaches.

In addition to its appealing performance, the most interesting issue regarding EPM-PAP might be its connection with ensemble learning or committee machines, a sub-area of machine learning that has drawn intensive attention during the last decade. In particular, PAP and ensemble learning shares the same philosophy. That is, the constituent algorithms (of PAP) or base learners (of an ensemble) must be different in order to make an improvement over using them individually. In the context of ensemble learning, the term "different" is referred to as diversity and a lot of studies have been carried out to investigate how diversity can be quantitatively defined and be utilized to construct good ensembles [22]. In this work, the term "different" is introduced as "complementary". Analysis on a simple case (i.e., with only two constituent EAs) reveal that the EPM-based constituent EA selection method can be interpreted as explicitly requiring constituent EAs to be complementary to each other. However, the concept of complementary used in this work is rather vague and by no means a perfect one. Intuitively, two EAs can be said as complementary only if they behave in different ways that can benefit each other. Hence, a more precise modeling of EAs' behavior is required in order to have a clearer understanding of the concept of complementary and to make better use of it to construct a PAP. This issue is worthy of further in-depth study in the future.

Acknowledgements

This work was supported in part by the 973 Program of China under Grant 2011CB707006, the National Natural Science Foundation of China under Grants 61175065 and 61329302, the Program for New Century Excellent Talents in University under Grant NCET-12-0512, the Science and Technological Fund of Anhui Province for Outstanding Youth under Grant 1108085J16, the EPSRC under Grant No. EP/J017515/1, and the European Union Seventh Framework Programme under Grant 247619. Xin Yao was supported by a Royal Society Wolfson Research Merit Award.

References

[1] M.Z. Ali, N.H. Awad, A novel class of niche hybrid cultural algorithms for continuous engineering optimization, Inf. Sci. 267 (2014) 158–190.
[2] A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in: Proc. 2005 IEEE Congress on Evolutionary Computation, CEC'05, Edinburgh, UK, 2005, pp. 1769–1776.
[3] A. Auger, N. Hansen, Performance evaluation of an advanced local search evolutionary algorithm, in: Proc. 2005 IEEE Congress on Evolutionary Computation, CEC'05, Edinburgh, UK, 2005, pp. 1777–1784.
[4] M. Birattari, T. Stützle, L. Paquete, K. Varrentrapp, A racing algorithm for configuring metaheuristics, in: Proc. 4th annual Conference on Genetic and Evolutionary Computation, GECCO'02, New York, USA, 2002, pp. 11–18.
[5] W.J. Conover, Practical Nonparametric Statistics, third ed., John Wiley & Sons, New York, NY, USA, 1999.
[6] K. Deb, A. Anand, D. Joshi, A computationally efficient evolutionary algorithm for real-parameter optimization, Evol. Comput. 10 (4) (2002) 371–395.
[7] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[8] M. Gagliolo, V. Zhumatiy, J. Schmidhuber, Adaptive online time allocation to search algorithms, in: Proc. 15th European Conference on Machine Learning, ECML'04, Pisa, Italy, 2004, pp. 134–143.
[9] C. Giraud-Carrier, R. Vilalta, P. Brazdil, Introduction to the special issue on meta-learning, Mach. Learn. 54 (3) (2004) 187–193.
[10] C.P. Gomes, B. Selmon, Algorithm portfolios, Artif. Intell. 126 (1–2) (2001) 43–62.
[11] B.A. Huberman, R.M. Lukose, T. Hogg, An economics approach to hard computational problems, Science 275 (5296) (1997) 51–54.
[12] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artif. Intell. 97 (1–2) (1997) 273–324.
[13] B. Lacroix, D. Molina, F. Herrera, Region based memetic algorithm for real-parameter optimization, Inf. Sci. 262 (2014) 15–31.
[14] C.Y. Lee, X. Yao, Evolutionary programming using the mutations based on the Levy probability distribution, IEEE Trans. Evol. Comput. 8 (1) (2004) 1–13.
[15] K. Leyton-Brown, E. Nudelman, Y. Shoham, Empirical hardness models: methodology and a case study on combinatorial auctions, J. ACM 56 (4) (2009) 1–52.
[16] R. Mallipeddi, P.N. Suganthan, Ensemble of constraint handling techniques, IEEE Trans. Evol. Comput. 14 (4) (2010) 561–597.
[17] F. Peng, K. Tang, G. Chen, X. Yao, Population-based algorithm portfolios for numerical optimization, IEEE Trans. Evol. Comput. 14 (5) (2010) 782–800.
[18] A.P. Piotrowski, Adaptive memetic differential evolution with global and local neighborhood-based mutation operators, Inf. Sci. 241 (2013) 164–194.
[19] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proc. 1998 IEEE International Conference on Evolutionary Computation, ICEC'98, Anchorage, AK, USA, 1998, pp. 69–73.
[20] S. Siegel, Nonparametric Statistics for the Behavioral Sciences, McGraw-Hill, New York, NY, USA, 1956.
[21] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC-2005 special session on real-parameter optimization, Technical report, Nanyang Technological University, Singapore, 2005.
[22] E.K. Tang, P.N. Suganthan, X. Yao, An analysis of diversity measures 65 (2006) 247–271.
[23] E. Tsang, A. Kwan, Mapping Constraint Satisfaction Problems to Algorithms and Heuristics, Technical Report CSM-198, University of Essex, UK, 1993.
[24] R. Vilalta, Y. Drissi, A perspective view and survey of meta-learning, Artif. Intell. Rev. 18 (2) (2002) 77–95.
[25] J.A. Vrugt, B.A. Robinson, J.M. Hyman, Self-adaptive multimethod search for global optimization in real-parameter spaces, IEEE Trans. Evol. Comput. 13 (2) (2009) 243–259.
[26] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.
[27] Z. Yang, K. Tang, X. Yao, Self-adaptive differential evolution with neighborhood search, in: Proc. 2008 IEEE Congress on Evolutionary Computation, CEC'08, Hong Kong, China, 2008, pp. 1110–1116.
[28] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (2) (1999) 82–102.
[29] B. Yuan, M. Gallagher, Statistical racing techniques for improved empirical evaluation of evolutionary algorithms, in: Proc. 8th International Conference on Parallel Problem Solving From Nature, PPSN'04, Birmingham, UK, 2004, pp. 172–181.
[30] S.Z. Zhao, P.N. Suganthan, Q. Zhang, Decomposition based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes, IEEE Trans. Evol. Comput. 16 (3) (2012) 442–446.