Integrated Computer-Aided Engineering -1 (2019) 1–17
DOI 10.3233/ICA-190723
IOS Press

An optimizer ensemble algorithm and its application to image registration

Xiaohu Yan[a], Fazhi He[b,*], Yongjun Zhang[a] and Xunwei Xie[a]
[a] *School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, China*
[b] *School of Computer Science, Wuhan University, Wuhan, Hubei, China*

Abstract. The design of effective optimization algorithms is always a hot research topic. An optimizer ensemble where any population-based optimization algorithm can be integrated is proposed in this study. First, the optimizer ensemble framework based on ensemble learning is presented. The learning table consisting of the population members of all optimizers is constructed to share information. The maximum number of iterations is divided into several exchange iterations. Each optimizer exchanges individuals with the learning table in exchange iterations and runs independently in the other iterations. Exchange individuals are generated by a bootstrap sample from the learning table. To maintain a balance between exchange individuals and preserved individuals, the exchange number of each optimizer is adaptively assigned according to its fitness. The output is obtained by the voting approach that selects the highest ranked solution. Second, an optimizer ensemble algorithm (OEA) which combines multiple population-based optimization algorithms is proposed. The computational complexity, convergence, and diversity of OEA are analyzed. Finally, extensive experiments on benchmark functions demonstrate that OEA outperforms several state-of-the-art algorithms. OEA is used to search the maximum mutual information in image registration. The high performance of OEA is further verified by a large number of registration results on real remote sensing images.

Keywords: Optimizer ensemble, ensemble learning, population-based optimization algorithm, image registration

1. Introduction

The design of effective optimization algorithms is a hot topic in the field of scientific research and engineering applications [1–3]. Many population-based optimization algorithms have been explored to solve optimization problems over the last few decades, such as genetic algorithm (GA) [4], particle swarm optimization (PSO) [5], and ant colony optimization (ACO) [6].

In general, population-based optimization algorithm can be divided into three categories: evolution-based algorithm, swarm-based algorithm, and physics-based algorithm [7,8]. Evolution-based algorithm is inspired by the concepts of evolution in nature [9,10]. The most famous evolution-based algorithms are GA [11–14], differential evolution (DE) [15,16], genetic programming (GP) [17], and evolutionary programing (EP) [18]. Swarm-based algorithm simulates the intelligent behavior of biology. The most popular swarm-based algorithms are PSO [19,20], ACO [21], artificial bee colony (ABC) algorithm [22], invasive weed optimization (IWO) [23], cuckoo search (CS) [24], fruit fly optimization algorithm (FOA) [25], harmony search algorithm (HSA) [26], and bat algorithm (BA) [27, 28]. Physics-based algorithm simulates the physical rules in the universe. The most well-known physics-based algorithms are gravitational search algorithm (GSA) [29], ray optimization (RO) [30], black hole (BH) [31], charged system search (CSS) [32], spiral dynamics algorithm (SpDO) [33], water drop algorithm (WDA) [34], and artificial chemical reaction optimization algorithm (ACROA) [35].

However, according to the no-free-lunch (NFL) theorem [36], no single algorithm can outperform all others on every optimization problem. Efficiently designed algorithms should specifically address the features of the problems to optimize [37]. This study aims to construct an ensemble of multiple population-based

*Corresponding author: Fazhi He, School of Computer Science, Wuhan University, Wuhan, Hubei, China. E-mail: fzhe@whu.edu.cn.

optimization algorithms, which can address reasonable ranges of problem features and adapt to solve a wide range of optimization problems.

Ensemble learning is a machine learning paradigm [38]. There are numerous studies for constructing the ensemble which consists of a set of individually trained classifiers, such as neural networks and decision trees [39]. Researchers have demonstrated that ensembles can often perform better than any single classifier [40]. The reason is that ensemble methods combine multiple models to improve overall performance [41].

Using the combination strategies in ensemble learning, this paper proposes an optimizer ensemble where any population-based optimization algorithm can be integrated. First, the population of an optimizer might not provide sufficient information for searching the global optimum. The learning table that consists of the population members of all optimizers is constructed to share information. Second, a single optimizer might not be able to solve complex optimization problems. The search mechanism simulating the natural phenomenon might be imperfect, which results in the local optimum entrapment. An optimizer ensemble algorithm (OEA) that combines different search mechanisms is presented to compensate for the imperfection. Third, the search space of an optimizer might not contain the global optimum. The maximum number of iterations is divided into several exchange iterations when optimizers exchange individuals with the learning table.

This paper is organized as follows. Section 2 is devoted to an introduction of related works. In Section 3, the optimizer ensemble framework is provided. In Section 4, OEA is introduced. In Section 5, experimental results are analyzed. The conclusions and future works are presented in Section 6.

2. Related works

2.1. Ensemble of algorithms/strategies

In real-word applications, each problem is characterized by its features, such as problem dimensionality, multi-modality, ill-conditioning, and dynamic behavior. A single optimizer may easily fall into local optima when solving complicated optimization problems [42, 43]. To solve a wide range of optimization problems, researchers have proposed hybrid algorithms which combine multiple algorithms/strategies [44,45]. Memetic computing algorithm is a structure that contains a main optimizer and one or more local search algorithms [46–48]. In hyper-heuristics and portfolio algorithms, a list of multiple optimizers is coordinated by means of a heuristic rule or supervisory/adaptive scheme [49].

In recent years, the ensembles of algorithms/ strategies have been studied. Mallipeddi et al. [50] proposed ensemble strategies with adaptive evolutionary programming. Wang and Li [51] designed a two-stage based ensemble optimization evolutionary algorithm to solve large-scale global optimization problems. Qu and Suganthan [52] constructed an ensemble of constraint handling methods to tackle constrained multi-objective optimization problems. Zhao et al. [53] proposed a decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes. Yu and Suganthan [54] constructed an ensemble of niching algorithms. Tasgetiren et al. [55] constructed an ensemble of discrete differential evolution algorithms. Mallipeddi and Suganthan [56] presented a differential evolution algorithm with ensemble of population members. Mallipeddi and Suganthan proposed a DE with an ensemble of mutation and crossover strategies and their associated control parameters [57]. Zhang et al. [58] proposed a novel way to design a P system for directly obtaining the approximate solutions of combinatorial optimization problems. Iacca et al. [59] presented a novel population-based algorithm combining two components with complementary algorithm logics. These ensembles mostly consist of multiple evolution-based algorithms. More algorithms/strategies cannot be integrated in the ensembles. Furthermore, the combination strategies in most ensembles are excessively complex, which results in a significant increase in extra calculation.

According to NFL theorem [36], there is no algorithm for solving all optimization problems. This is the motivation of this study, in which an ensemble of multiple population-based optimization algorithms is presented to solve a diverse array of optimization problems. To the best of our knowledge, there is no literature which presents the ensemble of population-based optimization algorithms. This study is the first work to construct an optimizer ensemble where any population-based optimization algorithm can be integrated.

2.2. Ensemble learning

Ensemble learning methods train multiple learners to solve a machine learning task. An ensemble contains a lot of learners called base learners. Base learners are generated by a base learning algorithm that may be decision tree or neural network. Ensemble learning methods have gained popularity because researchers have demonstrated that the prediction performance of the ensemble is usually better than that of a single learner on a variety of problems.

Ensemble learning algorithms can generally be divided into two frameworks: the dependent framework and the independent framework. In the dependent framework, the output of each learner affects the construction of the next learner. In the independent framework, each learner is built independently from other learners [60].

The most influential dependent algorithm for building an ensemble is boosting algorithm [61]. Boosting algorithm generates a set of learners sequentially [62]. The later learners focus more on the mistakes of the earlier learners. The level of focus is determined by a weight that is assigned to each training instance.

The most well-known independent algorithm is bagging algorithm [63]. Bagging algorithm adopts bootstrap sampling to obtain the data subsets for training base learners. Each data subset is used to train a different base learner of the same type [64]. The base learners' combination strategy is majority vote [65].

In this study, bagging algorithm will be employed to combine multiple optimizers in OEA. However, different from bagging algorithm, the type of each base optimizer is different, and the base optimizers are combined by the highest ranked solution in OEA.

3. Optimizer ensemble framework

To construct an ensemble of multiple optimizers, the related concepts are defined. A population-based optimization algorithm is an optimizer. The ensemble is homogeneous when the type of each base optimizer is the same. Otherwise, the ensemble is heterogeneous.

Without loss of generality, this paper will refer to the minimization problem of an objective function, which is defined as

$$\min f(x), x = [x_1, x_2, \ldots, x_D]^{\mathrm{T}} \tag{1}$$

where D is the dimension of the search space. In an iteration, individuals from other optimizers may have unexploited and unexplored positions that can help an optimizer to search the global optimum, which leads to the scope of individual exchange among optimizers.

3.1. Exchange iteration

The maximum number of iterations *maxIter* is divided into l blocks of iterations; the last of these iterations is an exchange iteration when an optimizer exchanges individuals with the other optimizers. All iterations are expressed by

$$\mathit{iter} = [1, 2, \ldots, E_1, 1, 2, \ldots, E_2, \ldots, E_l] \tag{2}$$

where E_i is the ith exchange iteration. The sum of all exchange iterations is equal to the maximum number of iterations *maxIter*. The relationship between E_i and *maxIter* is as follows

$$\mathit{maxIter} = \sum_{i=1}^{l} E_i \tag{3}$$

where l is the exchange frequency. Note that the setting of l impacts the information exchange and computational cost. When l is large, there are lots of exchange iterations for information sharing. Nevertheless, the computational cost is high due to the extra calculation in exchange iterations.

It is worth mentioning that the values of exchange iterations affect information exchange. In early iterations, optimizers have not obtained good solutions, which may lead to negative exchange. Meanwhile, the search mechanism of each optimizer may be disturbed when individuals are exchanged too early. In late iterations, optimizers may get trapped into local optima, and then the frequent exchange is helpful to avoid the local optimum and premature convergence. Thus, the exchange iteration E_i and exchange frequency l are dynamically adjusted according to the maximum number of iterations *maxIter* in this study, which is presented in Algorithm 1.

Algorithm 1: Calculation of the exchange iteration and exchange frequency.

Input: t, the threshold;
maxIter, the maximum number of iterations.
Output: E, the exchange iterations;
l, the exchange frequency.
$i = 1$;
$E_1 = \mathit{maxIter}/2$;
$s = E_1$;
while $E_i > t$ **do**
  $i = i + 1$;
  $E_i = \mathit{maxIter}/(2 \times i)$;
  $s = s + E_i$;
end
$l = i$;
$E_l = \mathit{maxIter} - s$.

As shown in Algorithm 1, the first exchange iteration is *maxIter*/2. Thus, each optimizer exchanges individuals in the late iterations when the iterations are equal to or greater than *maxIter*/2. Since optimizers may get trapped into local optima in late iterations, the individual exchange can increase the population diversity and enhance the search ability. It is unnecessary to exchange individuals with the learning table when E_i is small. As a result, the threshold t is set to ten.

3.2. Learning table

In an exchange iteration, multiple optimizers share information and knowledge via the learning table which consists of the population members of all optimizers. Suppose that the ensemble consists of m optimizers. In the ith exchange iteration E_i, the population of the jth optimizer is P_{ij}, then the learning table Lt_i is defined as

$$Lt_i = [P_{i1}, P_{i2}, \ldots, P_{im}]^{\mathrm{T}} \tag{4}$$

In an exchange iteration, each optimizer exchanges its individuals with the learning table. The exchange number of individuals significantly affects the information communication of each optimizer. To keep the convergence and search mechanism, more individuals in the population should be preserved. In contrast, to enhance the global search ability, an optimizer should exchange more individuals with the other optimizers that have better individuals. To maintain a balance between exchange individuals and preserved individuals, the exchange number of each optimizer is adaptively assigned according to its fitness.

Suppose that f_i is the best fitness of the ith optimizer in an exchange iteration, and N is the population size of an optimizer in the ensemble. Note that the population size of each optimizer in the ensemble is the same. Since the fitness difference among optimizers is large, the best fitness of each optimizer is normalized as

$$h_i = \frac{f_i - f_{\min}}{\sum\limits_{j=1}^{m} (f_j - f_{\min})} \tag{5}$$

where h_i is the normalized value of f_i, and $f_{\min}$ is the minimum value of f. The optimization problem is assumed to be a minimization problem in this paper. Thus, to obtain more good individuals, the optimizer with larger fitness should exchange more individuals with the learning table. To preserve more good individuals, the optimizer with smaller fitness should exchange fewer individuals with the learning table. Hence, the adaptive exchange number of the ith optimizer is expressed by

$$n_i = \mathit{round}(ce^{h_i}) \tag{6}$$

where e is the natural logarithm base; c is the exchange factor; $\mathit{round}(\cdot)$ is the rounding function. Since the population size of an optimizer is greater than or equal to its exchange number, the exchange factor c should be less than or equal to N/e. To share information sufficiently, the exchange factor is set to N/e. The denominator in Eq. (5) is zero when the best fitness of each optimizer is the same. In this case, the exchange number of each optimizer is set to $\mathit{round}(N/e)$.

3.3. Voting approach

Voting approach concerns how the best solutions of all optimizers are used in exchange iterations. In bagging algorithm, the combination strategy is a simple majority voting. Every learner has the same weight on the overall decision in majority voting.

Since the best fitness of each optimizer is different, the weight should not be the same in the optimizer ensemble. In the optimizer ensemble, the best solutions of all optimizers are sorted by their fitness values, and the highest ranked solution is considered to be the overall decision. The proposed voting approach can reduce the variance and output the global best solution obtained by all optimizers in the worst case.

3.4. Multi-optimizer combination

In an exchange iteration, a base optimizer in the ensemble interacts with the other optimizers via the learning table. The multi-optimizer combination based on ensemble learning is shown in Fig. 1.

It is clearly shown in Fig. 1 that multiple optimizers share information by exchanging individuals with the learning table that consists of the population members of all optimizers. Each optimizer exchanges individuals with the learning table in exchange iterations and runs independently in the other iterations, which can reduce the computational cost and make the combination simple. A new population for each optimizer is composed of a part of the current population and a bootstrap sample from the learning table. The output of all optimizers is obtained by the voting approach that selects the highest ranked solution.

As shown in Fig. 1, the best individual of each optimizer is added to its population after the exchange with the learning table. Thus, the best solution of each

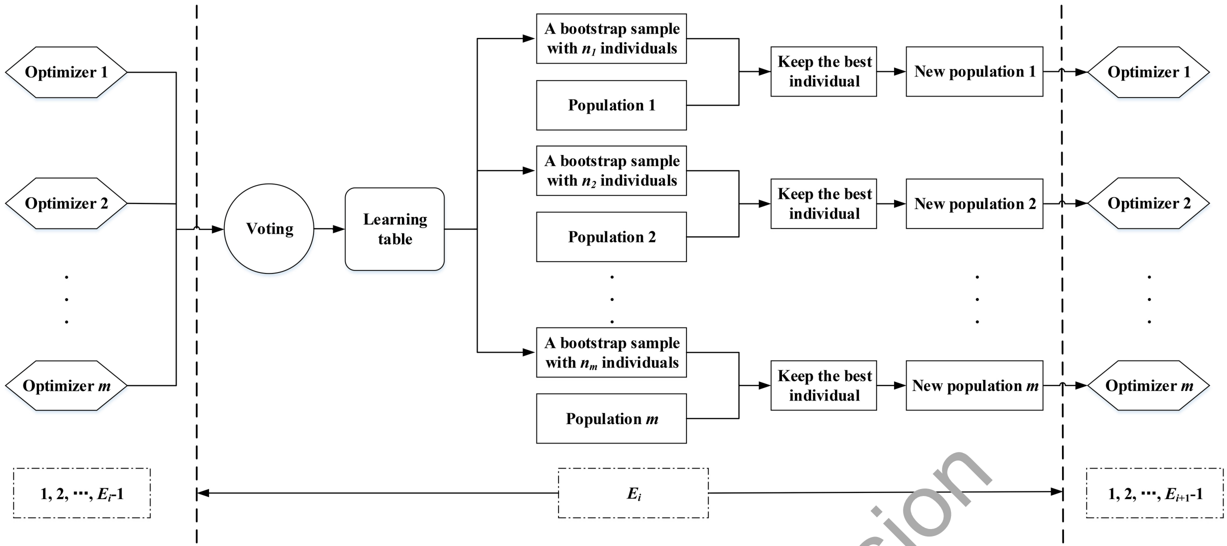

Fig. 1. Multi-optimizer combination in the optimizer ensemble.

optimizer is kept in the exchange iteration, which can help to enhance the search ability. Different from the crossover operation between two individuals [66], the individual exchange with the learning table is a master-slave mode that is more suitable for multiple optimizers to share information.

3.5. Ensemble construction

How to select an appropriate optimizer according to the optimization problem is an important step for constructing an effective ensemble. It is worthwhile to mention that the global search ability of an ensemble can be stronger than those of its base optimizers only if optimizers in the ensemble are different.

If all optimizers are identical, when an optimizer gets trapped into local optima, it is hard for the other optimizers to obtain the global optimum because their search mechanisms are identical. Therefore, to enhance the global search ability, the type of each optimizer is different, and the ensemble is heterogeneous in this study.

In optimization algorithms, the search process is focused on a balance between exploration and exploitation. Hence, it is wise to combine the optimizer that is good at exploitation with the optimizer that is good at exploration. It is also conducive to select optimizers with different categories of population-based optimization algorithms or optimizers with distinct characteristics. In summary, to construct an efficient ensemble, it is a good way to combine optimizers that are competitive, distinct, and complementary.

4. Optimizer ensemble algorithm

4.1. OEA

In the proposed optimizer ensemble, each optimizer exchanges individuals with the learning table in exchange iterations. Exchange individuals are generated by a bootstrap sample from the learning table. The exchange number is adaptively assigned to each optimizer. Thus, the resulting algorithm is presented in Algorithm 2.

In OEA, the maximum number of iterations is divided into l exchange iterations. First, m optimizers are initialized by a set of random solutions. Second, each optimizer runs independently when the current iteration is less than the exchange iteration. Each optimizer exchanges individuals with the learning table when the current iteration is equal to the exchange iteration. The exchange number n_i is adaptively assigned according to Eq. (6). The population of the ith optimizer in the exchange iteration is its initial population in the next iteration. Finally, the best fitness g and its corresponding position gx obtained by m optimizers are updated according to f and fx. The best solution obtained by all optimizers is the overall output of OEA.

The ensemble strategy in OEA differs from bagging algorithm. In bagging algorithm, a bootstrap sample with a fixed number is generated from the training set, and base learners are combined by majority voting. Nevertheless, in OEA, a bootstrap sample with adaptive number is generated from the learning table, and

Algorithm 2: The pseudo-code of OEA.

Input: E, the exchange iterations;
D, the dimension of the search space;
N, the population size of an optimizer;
l, the exchange frequency;
m, the number of optimizers.
Output: The best fitness g and its corresponding position gx obtained by m optimizers.
for $i = 1 : m$ **do**
  Randomly generate N individuals to initialize the ith optimizer;
end
for $k = 1 : l$ **do**
  for $i = 1 : m$ **do**
    for $j = 1 : E_k - 1$ **do**
      The ith optimizer runs independently in the jth iteration;
      Compute the fitness of each individual in the population;
      Update the best fitness f_i and its corresponding position fx_i of the ith optimizer;
    end
  end
  for $i = 1 : m$ **do**
    Normalize the best fitness f_i using Eq. (5);
    Compute the exchange number n_i using Eq. (6);
    The ith optimizer exchanges n_i individuals with the learning table;
  end
  Update the best fitness g and its corresponding position gx obtained by m optimizers;
end

base optimizers are combined by the highest ranked solution. Moreover, the type of each base learner is usually the same in bagging algorithm, while the ensemble is heterogeneous in OEA.

4.2. Computational complexity

It is difficult to solve large-scale optimization problems when the computational cost of an algorithm is too high. The computational complexity of OEA can be defined based on its implementation in Algorithm 2.

In OEA, the population size of an optimizer is N, and the dimension of the search space is D. It takes $O(N \times D)$ time to run an optimizer independently in an iteration. In an exchange iteration, the calculation of the exchange number can be implemented in $O(N \times D)$ time. Hence, the computational complexity of m optimizers in each iteration is $O(m \times N \times D)$. According to Eq. (3), the sum of all exchange iterations is equal to the maximum number of iterations *maxIter*. In other words, there are *maxIter* iterations in OEA. Therefore, the computational complexity of OEA is $O(\mathit{maxIter} \times m \times N \times D)$, which is equal to that of an optimizer with the population size of $m \times N$.

4.3. Convergence and diversity

The convergence and diversity of OEA are enhanced by the following strategies:

1) The learning table consists of the population members of all optimizers. Each optimizer exchanges individuals with the learning table. Thus, OEA can decrease the risk of local optimum entrapment and premature convergence by sharing information among all optimizers.
2) The exchange number of each optimizer is adaptively assigned according to its fitness. The weak optimizer exchanges more individuals with the learning table, which can take more good solutions from the other optimizers. The strong optimizer exchanges fewer individuals with the learning table, which can preserve more good solutions. The adaptive exchange number can maintain a balance between exploration and exploitation.
3) Exchange individuals of each optimizer are selected randomly with replacement from the learning table. Hence, the diversity of exchange individuals is increased by injecting randomness. Heterogeneous search mechanisms can produce good solutions and various population members, which is beneficial for the local optimum avoidance and population diversity.
4) The voting approach that selects the highest ranked solution can reduce the risk of selecting the local optimum and enhance the search ability. The ensemble can output the best solution obtained by all optimizers in the worst situation.

5. Experiment

To construct an efficient ensemble, it is conducive to select optimizers with different categories of population-based optimization algorithms. DE, PSO, and GSA belong to evolution-based algorithm, swarm-based algorithm, and physics-based algorithm, respectively. Thus, DE, PSO, and GSA are employed in OEA (OEA-DPG). The algorithms have been tested on CEC2013 benchmark and image registration problem. The detailed description of CEC 2013 can be found in [67].

The experimental analysis has been structured as follows. First, OEA-DPG is compared with its base optimizers and EPSDE, which is a DE with an ensemble of mutation and crossover strategies and their associated control parameters [57]. The exploitation and exploration abilities of OEA are analyzed. Second, the runtime of OEA-DPG is compared with that of its base optimizers. Third, to investigate the construction of an efficient OEA, different ensemble strategies are compared. Finally, to further analyze the performance of OEA, the algorithm is applied to image registration problem which is a real-world application.

5.1. Experimental setup

In this study, the population size of each algorithm is 150. For the sake of fair comparisons, the population size of each algorithm is the same. Hence, the population size of each optimizer in two-optimizer ensemble is 75, and the population size of each optimizer in three-optimizer ensemble is 50. The maximum number of iterations of each algorithm is 1000. The stopping criteria used for terminating iterations is to stop iterating when the maximum number is reached. If the global best solution is not improved in 50 iterations, then the iteration is stopped as well. According to Algorithm 1, the exchange iterations are set to [500, 250, 125, 63, 32, 16, 14].

In PSO, the learning factors are 2, and the inertial weight is decreased linearly from 0.9 to 0.2 over iterations. In DE, the crossover rate is 0.9, and the mutation factor is 0.5. The mutation strategy is DE/rand/1. The parameters of GSA and EPSDE are set according to their original literature [29,57], respectively. All experiments are executed on an Intel(R) Core(TM) i7-8700 @3.2 GHz CPU with 8 GB memory. The algorithms are written in Matlab R2018a.

Without loss of generality, all of the algorithms are run 30 times on each function. The average fitness value (AVE) and standard deviation (STD) over the 30 available runs are compared. Moreover, for each function, a statistical pair-wise comparison has been performed by applying the Wilcoxon rank-sum test at the 5% significant level. In all the result tables reported in this study, the symbols of "+", "=" and "−" respectively represent that the performance of OEA-DPG is better than, similar to and worse than that of the corresponding algorithm. For each function, the first two decimal places are considered, and the best average fitness value is marked in bold.

5.2. Comparison with popular optimizers

There are 28 benchmark functions in CEC2013 testbed, and the search range is $[-100, 100]$. These functions are divided into three groups: unimodal functions (F1-F5), multi-modal functions (F6-F20), and composite functions (F21-F28). The unimodal function has only one global optimum, which makes it useful for evaluating the exploitation ability. In contrast, the multi-modal function has multiple local optima, which makes it suitable for evaluating the exploration capability. The composite function combines multiple functions into a complex landscape, which can assess the performance of optimization algorithms from different perspectives.

To analyze the exploitation and exploration abilities of OEA, OEA-DPG is compared with its base optimizers and the ensemble algorithm EPSDE. Tables 1–3 display the comparison results on CEC2013 testbed in 10, 30, and 50 dimensions, respectively. In each table, the average, standard deviation, and Wilcoxon rank-sum test obtained by DE, PSO, GSA, EPSDE, and OEA-DPG are compared.

It can be seen from Tables 1–3 that OEA-DPG outperforms the other optimizers on most functions, especially on the composite functions which are more challenging. Although OEA-DPG has not obtained the best solution on some functions, OEA-DPG provides the good solution that is competitive. The reason is that OEA-DPG can make use of multiple search mechanisms.

Numerical results show that DE obtains good solutions on the majority of the unimodal functions, and PSO and GSA perform well on the multi-modal functions. Hence, the exploitation ability of DE is strong, and the exploration abilities of PSO and GSA are strong. OEA-DPG can take advantage of the algorithms whose search mechanisms are distinct and complementary, and hence OEA-DPG performs better on most functions.

By employing Wilcoxon's rank-sum test to analyze the experimental results, some findings are given as follows. OEA-DPG is better than DE, PSO, GSA and EPSDE on 17, 21, 24 and 17 functions in the case of $D = 10$, 22, 18, 24 and 24 functions in the case of $D = 30$, and 24, 18, 25 and 19 functions in the case of $D = 50$. In contrast, OEA-DPG is only worse than DE, PSO, GSA and EPSDE on 3, 0, 1 and 3 function(s) when $D = 10$, 2, 4, 2 and 2 functions when $D = 30$, and 1, 5, 2 and 5 functions when $D = 50$. Thus, the superiority of OEA-DPG is statistically significant, which confirms that the proposed ensemble framework is indeed effective.

Table 1
OEA-DPG against DE, PSO, GSA, and EPSDE on CEC2013 in 10 dimensions

Function	DE AVE	DE STD		PSO AVE	PSO STD		GSA AVE	GSA STD		EPSDE AVE	EPSDE STD		OEA-DPG AVE	OEA-DPG STD
F1	**−1.40E+03**	**0.00E+00**	=	**−1.40E+03**	**0.00E+00**	=	**−1.40E+03**	**0.00E+00**	=	**−1.40E+03**	**0.00E+00**	=	**−1.40E+03**	**0.00E+00**
F2	**−1.30E+03**	**1.27E−07**	−	3.49E+05	2.52E+05	+	5.06E+06	5.67E+05	+	−1.29E+03	3.41E+00	=	−6.01E+02	1.52E+03
F3	−1.20E+03	1.34E−01	=	1.03E+07	4.80E+07	+	1.10E+09	1.05E+09	+	−1.19E+03	5.72E+00	+	**−1.20E+03**	**1.58E+00**
F4	**−1.10E+03**	**1.24E−09**	−	−1.11E+02	5.65E+02	+	1.58E+04	1.70E+03	+	−1.10E+03	5.34E−02	−	−1.08E+03	2.79E+01
F5	**−1.00E+03**	**0.00E+00**	=	−1.00E+03	3.59E−14	+	−1.00E+03	1.07E−04	+	**−1.00E+03**	**0.00E+00**	=	**−1.00E+03**	**0.00E+00**
F6	**−9.00E+02**	**1.79E+00**	−	−8.97E+02	1.95E+00	+	−8.30E+02	2.89E+00	+	−8.96E+02	4.89E+00	=	−8.98E+02	2.96E+00
F7	−8.00E+02	9.18E−04	+	−7.96E+02	4.24E+00	+	−7.57E+02	2.68E+01	+	−8.00E+02	1.03E−01	+	**−8.00E+02**	**7.68E−04**
F8	−6.80E+02	7.15E−02	=	**−6.80E+02**	**6.46E−02**	=	−6.80E+02	7.87E−02	=	−6.80E+02	7.32E−02	=	−6.80E+02	8.72E−02
F9	−5.98E+02	1.14E+00	=	−5.97E+02	1.36E+00	+	−5.94E+02	1.31E+00	+	−5.94E+02	7.91E−01	+	**−5.99E+02**	**9.91E−01**
F10	−5.00E+02	7.52E−02	+	−5.00E+02	1.79E−01	+	−5.00E+02	1.24E−01	+	−5.00E+02	5.57E−02	+	**−5.00E+02**	**4.49E−02**
F11	−3.83E+02	2.79E+00	+	−3.99E+02	8.43E−01	+	−3.52E+02	7.10E+00	+	**−4.00E+02**	**3.70E−11**	−	−3.99E+02	9.23E−01
F12	−2.75E+02	3.21E+00	+	−2.86E+02	6.63E+00	+	−2.43E+02	7.76E+00	+	−2.86E+02	2.28E+00	+	**−2.92E+02**	**3.02E+00**
F13	−1.74E+02	3.16E+00	+	−1.79E+02	8.62E+00	+	−1.19E+02	1.19E+01	+	−1.84E+02	2.28E+00	+	**−1.90E+02**	**5.62E+00**
F14	9.61E+02	1.58E+02	+	2.15E+01	9.79E+01	+	8.55E+02	2.51E+02	+	−4.14E+01	1.63E+01	+	**−4.92E+01**	**6.50E+01**
F15	1.43E+03	1.49E+02	+	8.59E+02	2.99E+02	+	8.56E+02	1.82E+02	+	1.30E+03	1.23E+02	+	**7.28E+02**	**2.40E+02**
F16	2.01E+02	1.82E−01	+	2.01E+02	2.25E−01	+	**2.00E+02**	**2.52E−02**	=	2.01E+02	1.70E−01	+	2.00E+02	4.61E−02
F17	3.27E+02	3.40E+00	+	3.13E+02	1.68E+00	+	3.12E+02	1.00E+00	+	**3.10E+02**	**8.05E−02**	−	3.11E+02	1.51E+00
F18	4.36E+02	4.30E+00	+	4.26E+02	7.51E+00	+	**4.12E+02**	**3.54E−01**	−	4.32E+02	3.26E+00	+	4.15E+02	2.10E+00
F19	5.02E+02	3.37E−01	+	5.01E+02	2.08E−01	=	5.02E+02	3.66E−01	+	5.01E+02	7.85E−02	=	**5.01E+02**	**1.82E−01**
F20	6.03E+02	2.26E−01	+	6.03E+02	3.41E−01	+	6.04E+02	2.82E−01	+	6.03E+02	2.14E−01	+	**6.02E+02**	**3.93E−01**
F21	**1.04E+03**	**9.33E+01**	=	1.09E+03	3.66E+01	=	1.10E+03	4.63E−13	+	1.06E+03	8.14E+01	=	1.07E+03	7.59E+01
F22	1.47E+03	1.44E+02	+	1.00E+03	1.08E+02	=	3.04E+03	1.57E+02	+	9.85E+02	2.63E+01	+	**9.56E+02**	**1.03E+02**
F23	2.27E+03	1.44E+02	+	1.80E+03	3.42E+02	=	2.49E+03	2.33E+02	+	2.12E+03	1.27E+02	+	**1.75E+03**	**2.97E+02**
F24	1.20E+03	1.53E+01	=	1.21E+03	3.23E+00	+	1.23E+03	4.35E+00	+	1.21E+03	1.08E+01	+	**1.20E+03**	**1.69E+01**
F25	1.31E+03	3.83E+00	+	1.31E+03	1.77E+01	+	1.32E+03	3.77E+00	+	1.31E+03	4.87E+00	+	**1.31E+03**	**3.67E+00**
F26	1.40E+03	1.40E+01	+	1.37E+03	5.57E+01	+	1.49E+03	4.27E+01	+	1.39E+03	1.87E+01	+	**1.35E+03**	**4.31E+01**
F27	1.77E+03	4.64E+01	+	1.68E+03	3.13E+01	+	1.70E+03	1.74E−10	+	1.78E+03	3.73E+01	+	**1.64E+03**	**4.71E+01**
F28	1.67E+03	6.91E+01	=	1.70E+03	9.91E+01	=	2.16E+03	7.81E+01	+	1.69E+03	3.65E+01	=	**1.67E+03**	**7.58E+01**

Table 2
OEA-DPG against DE, PSO, GSA, and EPSDE on CEC2013 in 30 dimensions

Function	DE			PSO			GSA			EPSDE			OEA-DPG	
	AVE	STD		AVE	STD		AVE	STD		AVE	STD		AVE	STD
F1	−1.40E+03	3.60E−06	+	**−1.40E+03**	**1.93E−13**	−	−2.34E+02	4.80E+02	+	−1.40E+03	1.52E−09	+	−1.40E+03	4.76E−12
F2	7.21E+06	1.78E+06	+	1.55E+07	6.38E+06	+	3.98E+07	5.62E+06	+	9.43E+06	2.20E+06	+	**2.92E+06**	**1.40E+06**
F3	**7.88E+06**	**4.69E+06**	=	9.87E+08	1.09E+09	+	7.96E+12	2.59E+13	+	1.27E+07	5.85E+06	=	3.05E+07	6.05E+07
F4	4.55E+04	7.38E+03	+	**2.45E+04**	**6.94E+03**	=	6.76E+04	3.12E+03	+	4.73E+04	1.03E+04	+	2.54E+04	7.03E+03
F5	−1.00E+03	1.85E−04	+	**−1.00E+03**	**9.76E−09**	−	3.47E+02	1.33E+02	+	−1.00E+03	8.51E−07	+	−1.00E+03	5.60E−06
F6	**−8.85E+02**	**5.60E−01**	−	−8.72E+02	1.04E+01	+	−6.16E+02	5.31E+01	+	−8.84E+02	5.41E−01	−	−8.77E+02	4.69E+00
F7	−7.82E+02	4.63E+00	+	−7.12E+02	3.04E+01	+	2.92E+04	2.28E+04	+	−7.54E+02	6.08E+00	+	**−7.86E+02**	**9.24E+00**
F8	−6.79E+02	4.51E−02	=	−6.79E+02	5.81E−02	=	−6.79E+02	5.48E−02	=	−6.79E+02	3.84E−02	=	**−6.79E+02**	**5.84E−02**
F9	−5.60E+02	1.34E+00	+	**−5.75E+02**	**4.66E+00**	=	−5.60E+02	2.21E+00	+	−5.66E+02	1.24E+00	+	−5.75E+02	5.77E+00
F10	−4.99E+02	4.13E−02	+	−4.98E+02	1.11E+00	+	9.05E+01	8.99E+01	+	−4.99E+02	3.23E−02	+	**−5.00E+02**	**3.98E−01**
F11	−2.16E+02	1.46E+01	+	−3.66E+02	9.04E+00	+	6.88E+01	1.78E+01	+	−3.67E+02	2.98E+00	+	**−3.79E+02**	**8.59E+00**
F12	−9.64E+01	1.12E+01	+	−2.09E+02	2.62E+01	+	2.86E+02	4.06E+01	+	−1.45E+02	1.19E+01	+	**−2.70E+02**	**1.67E+01**
F13	3.80E+00	1.26E+01	+	−1.57E+01	3.35E+01	+	4.99E+02	4.39E+01	+	−3.07E+01	9.74E+00	+	**−1.16E+02**	**2.90E+01**
F14	5.88E+03	4.31E+02	+	**1.17E+03**	**3.12E+02**	−	4.28E+03	4.43E+02	+	2.34E+03	1.76E+02	+	1.97E+03	4.48E+02
F15	7.53E+03	3.19E+02	+	7.03E+03	6.19E+02	+	**4.10E+03**	**3.85E+02**	−	6.91E+03	3.75E+02	+	4.54E+03	4.90E+02
F16	2.03E+02	3.89E−01	+	2.02E+02	4.72E−01	+	**2.00E+02**	**1.22E−02**	−	2.03E+02	3.78E−01	+	2.00E+02	4.01E−02
F17	5.20E+02	8.39E+00	+	3.80E+02	1.34E+01	+	5.90E+02	2.81E+01	+	3.73E+02	3.77E+00	+	**3.46E+02**	**4.16E+00**
F18	6.36E+02	9.29E+00	+	6.34E+02	3.06E+01	+	6.67E+02	2.06E+01	+	6.09E+02	1.03E+01	+	**4.68E+02**	**1.22E+01**
F19	5.17E+02	7.73E−01	+	**5.04E+02**	**9.33E−01**	=	2.43E+03	5.55E+02	+	5.06E+02	4.15E−01	+	5.05E+02	1.80E+00
F20	6.13E+02	2.51E−01	+	6.13E+02	3.24E−01	+	6.15E+02	1.04E−01	+	6.13E+02	2.24E−01	+	**6.12E+02**	**4.32E−01**
F21	**9.50E+02**	**5.08E+01**	=	9.91E+02	6.68E+01	=	2.25E+03	1.29E+02	+	9.93E+02	7.42E+01	+	9.63E+02	4.90E+01
F22	7.04E+03	3.97E+02	+	**2.12E+03**	**3.53E+02**	−	8.03E+03	4.57E+02	+	3.68E+03	2.15E+02	+	3.09E+03	5.31E+02
F23	8.38E+03	3.21E+02	+	8.01E+03	5.66E+02	+	**7.30E+03**	**3.34E+02**	=	7.81E+03	3.84E+02	+	7.37E+03	4.12E+02
F24	1.26E+03	1.75E+01	+	1.27E+03	9.82E+00	+	1.48E+03	5.99E+01	+	1.29E+03	4.29E+00	+	**1.22E+03**	**1.14E+01**
F25	1.35E+03	6.13E+00	=	1.37E+03	8.85E+00	+	1.52E+03	1.02E+01	+	1.40E+03	2.54E+00	+	**1.35E+03**	**7.49E+00**
F26	**1.40E+03**	**4.04E−01**	−	1.51E+03	7.91E+01	+	1.59E+03	3.40E+01	+	1.40E+03	2.17E−01	−	1.44E+03	6.28E+01
F27	2.55E+03	1.22E+02	+	2.26E+03	1.18E+02	+	2.54E+03	7.89E+01	+	2.47E+03	3.40E+01	+	**1.96E+03**	**1.12E+02**
F28	1.70E+03	3.41E−02	+	1.78E+03	2.99E+02	=	5.84E+03	3.23E+02	+	1.70E+03	1.43E−03	+	**1.70E+03**	**3.02E−04**

Table 3
OEA-DPG against DE, PSO, GSA, and EPSDE on CEC2013 in 50 dimensions

Function	DE			PSO			GSA			EPSDE			OEA-DPG	
	AVE	STD		AVE	STD		AVE	STD		AVE	STD		AVE	STD
F1	−1.40E+03	2.33E−02	+	**−1.40E+03**	**7.42E−05**	−	1.26E+04	1.39E+03	+	−1.40E+03	5.05E−05	+	−1.40E+03	2.11E−05
F2	1.23E+08	2.28E+07	+	4.33E+07	1.89E+07	+	1.33E+08	2.20E+07	+	6.56E+07	9.12E+06	+	**1.99E+07**	**7.49E+06**
F3	4.94E+09	2.24E+09	+	1.56E+10	9.53E+09	+	7.17E+11	5.17E+11	+	4.85E+09	1.24E+09	+	**7.84E+08**	**5.69E+08**
F4	1.33E+05	1.51E+04	+	6.60E+04	1.06E+04	+	9.02E+04	2.60E+03	+	1.18E+05	1.61E+04	+	**6.04E+04**	**6.38E+03**
F5	−1.00E+03	7.54E−02	+	−1.00E+03	3.29E−03	−	1.05E+03	2.65E+02	+	**−1.00E+03**	**1.45E−03**	−	−1.00E+03	4.68E−02
F6	−8.53E+02	5.72E−01	=	−8.51E+02	9.89E+00	=	−7.03E+01	1.14E+02	+	**−8.54E+02**	**6.98E−01**	−	−8.51E+02	9.99E+00
F7	−7.20E+02	1.15E+01	+	−6.58E+02	2.04E+01	+	1.59E+03	1.13E+03	+	−6.82E+02	1.05E+01	+	**−7.30E+02**	**1.77E+01**
F8	−6.79E+02	3.15E−02	=	−6.79E+02	4.88E−02	=	−6.79E+02	2.86E−02	+	−6.79E+02	2.99E−02	=	**−6.79E+02**	**3.26E−02**
F9	−5.26E+02	1.79E+00	+	**−5.46E+02**	**5.64E+00**	=	−5.36E+02	2.83E+00	+	−5.34E+02	1.60E+00	+	−5.42E+02	9.79E+00
F10	−4.84E+02	6.39E+00	+	−4.61E+02	1.58E+01	+	1.35E+03	1.40E+02	+	−4.92E+02	1.96E+00	+	**−4.94E+02**	**3.35E+00**
F11	−2.18E+01	1.75E+01	+	−3.16E+02	1.51E+01	+	2.28E+02	3.08E+01	+	−2.66E+02	8.82E+00	+	**−3.32E+02**	**2.08E+01**
F12	1.06E+02	1.33E+01	+	−5.41E+01	7.49E+01	+	6.01E+02	4.44E+01	+	5.53E+01	1.82E+01	+	**−2.05E+02**	**2.84E+01**
F13	2.08E+02	1.92E+01	+	2.04E+02	8.08E+01	+	9.33E+02	5.65E+01	+	1.67E+02	1.89E+01	+	**1.43E+01**	**4.05E+01**
F14	1.18E+04	4.28E+02	+	**2.81E+03**	**6.02E+02**	−	8.17E+03	7.76E+02	+	5.56E+03	4.62E+02	=	5.67E+03	6.98E+02
F15	1.44E+04	3.86E+02	+	1.41E+04	4.58E+02	+	**9.52E+03**	**6.42E+02**	=	1.35E+04	4.30E+02	+	9.59E+03	8.65E+02
F16	2.04E+02	3.23E−01	+	2.03E+02	4.14E−01	+	**2.00E+02**	**1.01E−02**	−	2.04E+02	3.44E−01	+	2.00E+02	3.42E−02
F17	7.41E+02	1.46E+01	+	4.90E+02	2.90E+01	+	9.58E+02	4.68E+01	+	5.04E+02	9.59E+00	+	**4.06E+02**	**1.63E+01**
F18	8.60E+02	1.29E+01	+	9.13E+02	4.60E+01	+	1.04E+03	4.79E+01	+	8.30E+02	2.03E+01	+	**5.74E+02**	**2.06E+01**
F19	5.36E+02	1.66E+00	+	**5.12E+02**	**3.07E+00**	−	2.06E+04	3.98E+03	+	5.17E+02	8.93E−01	−	5.18E+02	4.16E+00
F20	6.23E+02	2.14E−01	+	6.23E+02	3.47E−01	+	6.25E+02	2.18E−01	+	6.23E+02	3.44E−01	+	**6.22E+02**	**5.25E−01**
F21	**1.09E+03**	**3.73E+02**	=	1.49E+03	4.30E+02	=	3.95E+03	5.84E+01	+	1.11E+03	3.82E+02	=	1.30E+03	4.31E+02
F22	1.31E+04	4.29E+02	+	**4.02E+03**	**6.52E+02**	−	1.42E+04	6.11E+02	+	6.87E+03	2.61E+02	−	7.57E+03	1.22E+03
F23	1.51E+04	4.37E+02	+	1.51E+04	7.35E+02	+	**1.36E+04**	**4.47E+02**	−	1.46E+04	4.23E+02	+	1.42E+04	4.98E+02
F24	1.37E+03	1.24E+01	+	1.34E+03	1.19E+01	+	1.91E+03	1.47E+02	+	1.37E+03	4.62E+00	+	**1.28E+03**	**1.15E+01**
F25	1.45E+03	2.89E+01	+	1.44E+03	1.52E+01	+	1.76E+03	2.71E+01	+	1.49E+03	4.55E+00	+	**1.41E+03**	**1.65E+01**
F26	1.66E+03	7.39E+01	+	1.63E+03	4.56E+01	=	1.67E+03	8.73E+01	+	**1.56E+03**	**1.03E+02**	=	1.60E+03	9.18E+01
F27	3.49E+03	4.63E+01	+	2.99E+03	1.52E+02	+	4.18E+03	1.33E+02	+	3.29E+03	3.68E+01	+	**2.72E+03**	**2.81E+02**
F28	1.80E+03	4.17E−01	−	2.58E+03	1.44E+03	+	1.02E+04	4.20E+02	+	**1.80E+03**	**2.53E−02**	−	2.00E+03	7.75E+02

5.3. Runtime

To analyze the computational cost, the runtime of OEA-DPG is compared with that of its base optimizers. The difference of runtime among the algorithms is similar in 10, 30, and 50 dimensions on CEC2013. Due to the page limit, the results in 30 dimensions are selected for comparison. Figure 2 presents the average runtime of DE, PSO, GSA, and OEA-DPG. In Fig. 2, the horizontal axis represents the function, and the vertical axis represents the average runtime in seconds.

As shown in Fig. 2, it is clear that OEA consumes more time than its base optimizers due to the extra calculation in exchange iterations. However, the runtime of OEA-DPG is competitive with that of DE, PSO, and GSA except for F9, F16, and the composite functions. The reason is that there are only seven exchange iterations for the individual exchange in OEA-DPG. Each optimizer runs independently in the other iterations. The runtime of OEA-DPG is large on the composite functions due to the large runtime of DE and PSO, which demonstrates that the computational cost of extra calculation in OEA-DPG is low.

5.4. Analysis of ensemble strategies

Several ensemble strategies are designed in OEA to promote its performance. To analyze the influence of the search mechanism in OEA, this paper compares heterogeneous ensembles with homogeneous ensembles. The ensemble of DE, DE and DE (OEA-DDD), the ensemble of PSO, PSO and PSO (OEA-PPP), and the ensemble of GSA, GSA and GSA (OEA-GGG) are compared with OEA-DPG. The average, standard deviation, and Wilcoxon rank-sum test obtained by OEA-DPG and homogeneous ensembles are compared in Table 4. Due to the page limit, the results on CEC2013 testbed in 30 dimensions are selected for comparison.

As can be seen from Table 4, OEA-DPG is superior to OEA-PPP and OEA-GGG on almost all functions, and OEA-DPG is better than or similar to OEA-DDD on the majority of functions. OEA-DDD performs well on the unimodal functions because of the strong exploitation ability of DE. Compared with the base optimizer in Table 2, the homogeneous ensemble of multiple optimizers has not improved the performance obviously. The reason is that the search mechanisms of base optimizers are identical in the homogeneous ensemble. Due to the combination of different and complementary search mechanisms, OEA-DPG is better than OEA-DDD, OEA-PPP and OEA-GGG on 13, 24 and 26 functions, while OEA-DPG is only worse than OEA-DDD, OEA-PPP and OEA-GGG on 11, 2 and 0 function(s).

In an exchange iteration, the exchange number of each optimizer is adaptively assigned according to its fitness in OEA. To analyze the influence of the adaptive exchange number, OEA-DPG with a fixed exchange number (OEA-DPG-F) is compared. In OEA-DPG-F, the fixed exchange number of exchange individuals is 20. Table 5 displays the comparison result of OEA-DPG and OEA-DPG-F on CEC2013 testbed in 30 dimensions.

As can be clearly seen from Table 5 that OEA-DPG is better than OEA-DPG-F on 11 functions, and OEA-DPG is similar to OEA-DPG-F on 17 functions. It is worthwhile to mention that there is no function on which OEA-DPG is worse than OEA-DPG-F. These results are mainly due to the fact that the adaptive exchange number can maintain a balance between exchange individuals and preserved individuals. When a fixed exchange number is assigned to each optimizer, the weak optimizer cannot exchange more individuals with the other optimizers, and the strong optimizer cannot preserve more good individuals, which decreases the global search ability. Hence, the performance of OEA-DPG is higher than or similar to that of OEA-DPG-F on all functions, which conforms the effectiveness of the adaptive exchange number.

5.5. Image registration problem

To further investigate the performance of OEA, the algorithm is applied to solve image registration problem, which is a fundamental and crucial issue in remote sensing image processing [68]. Mutual information (MI) is a commonly used similarity measure in image registration [69]. The larger the MI, the better the registration [70]. According to the information theoretic notion of entropy, MI of images A and B can be computed as

$$I(A,B) = H(A) + H(B) - H(A,B) \tag{7}$$

where $H(A)$ and $H(B)$ are the marginal entropies of images A and B, respectively and $H(A,B)$ is their joint entropy. These can be denoted as

$$H(A) = -\sum_{a} P_A(a)\log_2 P_A(a) \tag{8}$$

$$H(B) = -\sum_{b} P_B(b)\log_2 P_B(b) \tag{9}$$

$$H(A,B) = -\sum_{a,b} P_{AB}(a,b)\log_2 P_{AB}(a,b) \tag{10}$$

Table 4
OEA-DPG against OEA-DDD, OEA-PPP, and OEA-GGG on CEC2013 in 30 dimensions

Function	OEA-DDD			OEA-PPP			OEA-GGG			OEA-DPG	
	AVE	STD		AVE	STD		AVE	STD		AVE	STD
F1	**−1.40E+03**	**1.53E−12**	−	−1.40E+03	2.78E−01	+	3.87E+03	1.41E+03	+	−1.40E+03	9.79E−12
F2	**1.73E+06**	**9.25E+05**	−	2.37E+07	8.54E+06	+	8.54E+07	6.20E+07	+	3.52E+06	2.01E+06
F3	**2.17E+06**	**3.97E+06**	−	1.64E+09	1.19E+09	+	4.42E+14	7.18E+14	+	1.46E+07	1.56E+07
F4	**1.84E+04**	**4.01E+03**	−	7.63E+04	2.26E+04	+	6.07E+04	6.65E+03	+	2.34E+04	6.03E+03
F5	**−1.00E+03**	**1.70E−08**	−	−9.99E+02	6.58E−01	+	3.66E+02	4.35E+02	+	−1.00E+03	5.69E−05
F6	**−8.81E+02**	**4.68E+00**	−	−8.72E+02	2.09E+00	+	8.99E+01	3.23E+02	+	−8.78E+02	4.46E+00
F7	**−7.96E+02**	**2.57E+00**	−	−7.21E+02	2.27E+01	+	2.71E+04	3.17E+04	+	−7.86E+02	9.75E+00
F8	−6.79E+02	4.64E−02	=	−6.79E+02	5.41E−02	+	−6.79E+02	6.95E−02	+	**−6.79E+02**	**5.71E−02**
F9	−5.64E+02	6.61E+00	+	−5.76E+02	3.47E+00	=	−5.59E+02	2.83E+00	+	**−5.77E+02**	**5.46E+00**
F10	**−5.00E+02**	**2.46E−02**	−	−4.73E+02	1.35E+01	+	6.74E+02	3.53E+02	+	−5.00E+02	3.41E−01
F11	−2.92E+02	1.74E+01	+	−3.53E+02	9.75E+00	+	3.49E+01	6.66E+01	+	**−3.79E+02**	**1.14E+01**
F12	−1.18E+02	1.33E+01	+	−1.93E+02	2.98E+01	+	3.19E+02	9.08E+01	+	**−2.63E+02**	**2.38E+01**
F13	−7.64E+00	9.85E+00	+	−1.35E+00	3.46E+01	+	4.92E+02	7.75E+01	+	**−1.20E+02**	**3.57E+01**
F14	4.21E+03	4.93E+02	+	**1.75E+03**	**4.33E+02**	−	3.77E+03	5.20E+02	+	2.16E+03	4.58E+02
F15	7.60E+03	2.41E+02	+	5.58E+03	1.18E+03	+	4.72E+03	6.89E+02	=	**4.38E+03**	**5.47E+02**
F16	2.03E+02	4.03E−01	+	2.03E+02	8.39E−01	+	2.04E+02	7.23E−0[illegible]	+	**2.00E+02**	**5.11E−02**
F17	4.46E+02	1.28E+01	+	4.42E+02	2.14E+01	+	7.16E+02	6.77E+01	+	**3.45E+02**	**4.31E+00**
F18	6.15E+02	1.54E+01	+	6.86E+02	3.68E+01	+	8.91E+02	4.85E+01	+	**4.72E+02**	**1.44E+01**
F19	5.14E+02	1.43E+00	+	5.09E+02	2.76E+00	+	4.22E+03	3.[illegible]6E+03	+	**5.05E+02**	**1.69E+00**
F20	6.13E+02	2.81E−01	+	6.13E+02	4.45E−01	+	6.15E+02	1.09E−01	+	**6.12E+02**	**4.85E−01**
F21	**9.47E+02**	**5.07E+01**	−	1.01E+03	9.69E+01	+	2.30E+03	2.19E+02	+	9.76E+02	6.55E+01
F22	5.37E+03	5.08E+02	+	2.86E+03	4.90E+02	=	6.64E+03	8.96E+02	+	**2.84E+03**	**4.06E+02**
F23	8.42E+03	3.16E+02	+	**6.80E+03**	**9.54E+02**	−	7.80E+03	4.28E+02	=	7.50E+03	5.89E+02
F24	**1.21E+03**	**9.63E+00**	−	1.26E+03	8.66E+00	+	1.50E+03	9.05E+01	+	1.23E+03	1.25E+01
F25	1.35E+03	7.11E+00	=	1.36E+03	7.68E+00	+	1.[illegible]4E+03	1.73E+01	+	**1.35E+03**	**7.15E+00**
F26	**1.42E+03**	**4.71E+01**	−	1.51E+03	7.35E+01	+	1.62E+03	7.21E+01	+	1.44E+03	6.18E+01
F27	**1.92E+03**	**1.18E+02**	=	2.19E+03	9.76E+01	+	2.60E+03	8.68E+01	+	1.96E+03	1.43E+02
F28	1.70E+03	4.29E−05	=	1.74E+03	4.24E+01	+	6.25E+03	3.94E+02	+	**1.70E+03**	**2.03E−04**

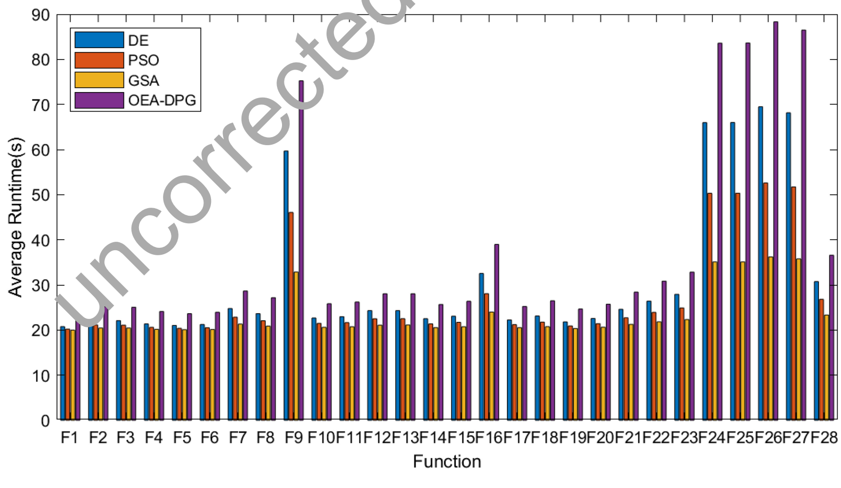

Fig. 2. Runtime comparison of DE, PSO, GSA, and OEA-DPG.

where $a \in A$, $b \in B$, $P_A(a)$ and $P_B(b)$ are the marginal probability distributions of images A and B, respectively, and $P_{AB}(a, b)$ is the joint probability distribution of images A and B [71].

The rigid transformation model is considered in this study due to its wide applicability. The translations of the x-axis and y-axis are denoted as t_x and t_y, respectively. The rotation is denoted as θ. Then the rigid transformation model can be formulated as

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{11}$$

Images registration based on MI is essentially an

Table 5
OEA-DPG against OEA-DPG-F on CEC2013 in 30 dimensions

Function	OEA-DPG-F			OEA-DPG	
	AVE	STD		AVE	STD
F1	−1.40E+03	2.27E−11	+	**−1.40E+03**	**9.79E−12**
F2	3.71E+06	1.49E+06	=	**3.52E+06**	**2.01E+06**
F3	**1.20E+07**	**1.69E+07**	=	1.46E+07	1.56E+07
F4	2.50E+04	5.84E+03	=	**2.34E+04**	**6.03E+03**
F5	−1.00E+03	2.71E−06	=	**−1.00E+03**	**5.69E−05**
F6	−8.78E+02	4.42E+00	=	**−8.78E+02**	**4.46E+00**
F7	**−7.89E+02**	**7.68E+00**	=	−7.86E+02	9.75E+00
F8	−6.79E+02	6.76E−02	+	**−6.79E+02**	**5.71E−02**
F9	**−5.78E+02**	**5.30E+00**	=	−5.76E+02	5.46E+00
F10	−5.00E+02	2.73E−01	=	**−5.00E+02**	**3.41E−01**
F11	−3.63E+02	1.73E+01	+	**−3.79E+02**	**1.14E+01**
F12	−2.36E+02	2.81E+01	+	**−2.63E+02**	**2.38E+01**
F13	−9.84E+01	3.57E+01	+	**−1.20E+02**	**3.57E+01**
F14	2.42E+03	5.45E+02	+	**2.16E+03**	**4.58E+02**
F15	4.43E+03	5.44E+02	=	**4.38E+03**	**5.47E+02**
F16	2.00E+02	5.87E−02	+	**2.00E+02**	**5.11E−02**
F17	3.52E+02	9.47E+00	+	**3.45E+02**	**4.31E+00**
F18	4.86E+02	2.45E+01	+	**4.72E+02**	**1.[illegible]4E+01**
F19	5.06E+02	3.26E+00	=	**5.05E+02**	**1.69E+00**
F20	6.12E+02	4.94E−01	=	**6.12E+02**	**4.85E−01**
F21	1.01E+03	1.02E+02	=	**9.76E+02**	**6.55E+01**
F22	3.61E+03	7.61E+02	+	**2.98E+03**	**4.06E+02**
F23	**7.40E+03**	**7.06E+02**	=	7.50E+03	5.89E+02
F24	1.23E+03	1.11E+01	=	**1.23E+03**	**1.25E+01**
F25	1.35E+03	6.62E+00	=	**1.35E+03**	**7.15E+00**
F26	**1.42E+03**	**5.10E+01**	=	1.44E+03	6.18E+01
F27	1.96E+03	1.72E+02	=	**1.96E+03**	**1.43E+02**
F28	1.70E+03	1.82E−03	+	**1.70E+03**	**2.03E−04**

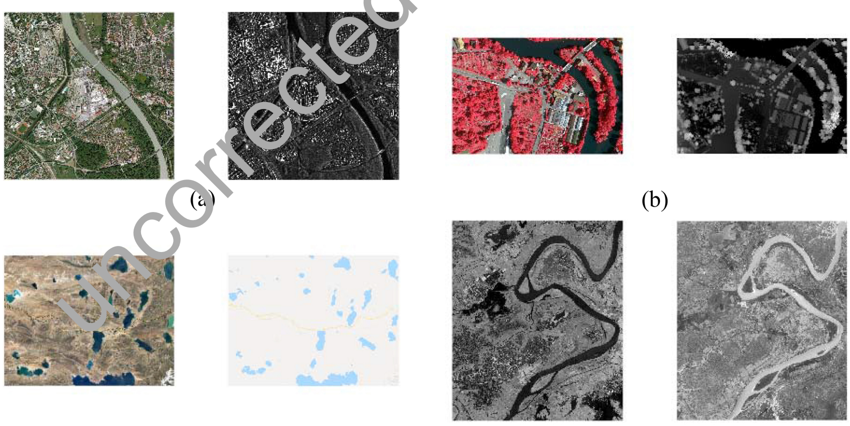

Fig. 3. Remote sensing image pairs. (a) visible-SAR. (b) LiDAR-visible. (c) image-map. (d) infrared-visible.

optimization problem of searching for the optimal parameters t_x, t_y, and θ. The multi-modal remote sensing images are used to test the algorithms, which are shown in Fig. 3. Four types of multi-modal remote sensing images are selected as experimental sets, including visible-synthetic aperture radar (SAR), light detection and ranging (LiDAR)-visible, image-map, and infrared-visible.

As shown in Fig. 3, for each image pair, the image on the left is the reference image, and the image on the right is the sensed image. There are obvious intensity, translation and rotation changes between the reference and sensed images. The images are captured by different sensors, from different places, at different time, or

Table 6
MI and RMSE comparison of DE, PSO, GSA, EPSDE, and OEA-DPG on image registration problem

Image pair	DE		PSO		GSA		EPSDE		OEA-DPG	
	MI	RMSE	MI	RMSE	MI	RMSE	MI	RMSE	MI	RMSE
a	0.1608	2.0244	0.1607	2.0533	0.1591	2.6129	0.1608	1.9678	**0.1610**	**1.6433**
b	0.4153	1.4929	0.3963	2.4615	0.4121	1.6200	0.4142	1.5485	**0.4153**	**1.4927**
c	**0.2274**	**1.5519**	0.2180	2.4443	0.2266	1.5539	0.2273	1.5528	0.2273	1.5525
d	0.2066	1.3480	0.1858	2.0916	0.2048	1.3942	0.2066	1.3477	**0.2067**	**1.3439**

from different viewpoints, which can test the efficiency and robustness of the proposed algorithm comprehensively.

The root mean square error (RMSE) of check points is used to evaluate the registration accuracy quantitatively. In general, the check points are determined manually. Specifically, for each image pair, 40–50 evenly distributed check points with subpixel accuracy between the reference and sensed images are selected [72]. The smaller the RMSE, the higher the registration accuracy.

The upper and lower boundaries of the transformation parameters t_x, t_y, and θ are set to [-100, -100, -100; 100, 100, 100]. When the value of MI is larger than 0.8, the image registration is considered to be satisfactory, and hence the iteration is stopped. Since the registration of remote sensing images is very time-consuming, the algorithms are run once on each image pair. Comparison results of the algorithms on image registration problem are presented in Table 6.

It can be seen from Table 6 that RMSE of OEA-DPG is smaller than 2 pixels on each image pair, which demonstrates that OEA-DPG handles translation and rotation changes well and achieves satisfactory registration. OEA-DPG is superior to the other algorithms on image pairs a, b, and d. This is mainly attributed to the fact that OEA-DPG has stronger global search ability and obtains better transformation parameters. However, DE outperforms OEA-DPG on image pair c. No algorithm outperforms the others on each image pair, which is in accord with NFL theorem. Although OEA-DPG is outperformed, it still obtains competitive results. Thus, OEA-DPG is more suitable for solving real-world optimization problems.

6. Conclusions

An optimizer ensemble where any population-based optimization algorithm can be integrated is proposed in this study. Multiple optimizers share information by exchanging individuals with the learning table. Each optimizer exchanges information in exchange iterations and runs independently in the other iterations. The output is obtained by the voting approach that selects the highest ranked solution. The proposed ensemble benefits from the optimizer ensemble strategies, such as the learning table, the heterogeneous search mechanism, and the voting approach. The high performance of OEA is confirmed by the empirical results on CEC2013 benchmark and image registration problem.

OEA is significantly different from other optimization algorithms. Other optimization algorithms mostly simulate the swarm intelligence behavior or evolutionary process. Nevertheless, OEA is inspired by ensemble learning that is a machine learning paradigm. Most hybrid optimization algorithms combine two or three different optimizers, while more optimizers can be integrated into the ensemble in OEA.

The important feature that makes OEA unique from other ensembles of algorithms is that OEA can be applied to any population-based optimization algorithm, while other ensembles can only be applied to evolution-based algorithm or swarm-based algorithm. In most ensembles, each optimizer exchanges information in all iterations. However, in OEA, each optimizer exchanges information only in exchange iterations and runs independently in the other iterations. Furthermore, different from the point-point mode of information sharing in other ensembles, the information exchange between the learning table and optimizers is a master-slave mode in OEA.

In the future, the following directions will be investigated:

1) Although OEA performs well in most cases, the performance of OEA algorithm mainly depends on the selected optimizers. When the base optimizers are improperly selected, the performance of OEA is poor. It is suggested that OEA combines optimizers that are distinct and complementary. Future work needs to be done to construct efficient OEA.
2) Since OEA has shown impressive performance in various optimization problems, OEA will be applied to more real-word optimization problems, such as computer aided design (CAD), image segmentation, and video processing [73–79].

3) The optimizer ensemble will benefit from the integration with deep learning methods [80–82]. Trained by the data in the previous iterations, a deep network can generate good solutions for optimizers in the exchange iteration, which is helpful to enhance the performance of OEA. However, training a deep network is usually a very time-consuming process [83–85], which needs to be improved in OEA.
4) Since the proposed ensemble is compatible with any population-based optimization algorithm [86–90], OEA will be applied to multi-objective optimization algorithms. To evaluate each optimizer, a weighted sum fitness function with a different weight vector will be constructed in the ensemble of multi-objective optimization algorithms.

Acknowledgments

This work is supported by the National Key Research and Development Program of China under Grant 2018YFB0505003, and the National Natural Science Foundation of China under Grants 41871368 and 41601352.

References

[1] Palacios J.J., González-Rodríguez I., Vela C.R., and Puente J. Satisfying flexible due dates in fuzzy job shop by means of hybrid evolutionary algorithms, *Integrated Computer-Aided Engineering* **26**(1) (2019), 65–84.
[2] Wang Q., Liu H., Yuan J., and Chen L. Optimizing the energy-spectrum efficiency of cellular systems by evolutionary multi-objective algorithm, *Integrated Computer-Aided Engineering* **26**(2) (2019), 207–220.
[3] Cui L., Li G., Zhu Z., Lin Q., Wong K., Chen J., Lu N., and Lu J. Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism, *Information Sciences* **422** (2018), 122–143.
[4] Holland J.H. Genetic algorithms, *Scientific American* **267**(1) (1992), 66–73.
[5] Eberhart R., and Kennedy J. A new optimizer using particle swarm theory, in: *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39–43.
[6] Zhou Y., He F., Hou N., and Qiu Y. Parallel ant colony optimization on multi-core SIMD CPUs, *Future Generation Computer Systems* **79**(2) (2018), 473–487.
[7] Pan L., He C., Tian Y., Su Y., and Zhang X. A region division based diversity maintaining approach for many-objective optimization, *Integrated Computer-Aided Engineering* **24**(3) (2017), 279–296.
[8] Mirjalili S., and Lewis A. The whale optimization algorithm, *Advances in Engineering Software* **95** (2016), 51–67.
[9] Kyriklidis C., and Dounias G. Evolutionary computation for resource leveling optimization in project management, *Integrated Computer-Aided Engineering* **23**(2) (2016), 173–184.
[10] Rostami S., and Neri F. Covariance matrix adaptation pareto archived evolution strategy with hypervolume-sorted adaptive grid algorithm, *Integrated Computer-Aided Engineering* **23**(4) (2016), 313–329.
[11] Mencía R., Sierra M.R., Mencía C., and Varela R. Genetic algorithms for the scheduling problem with arbitrary precedence relations and skilled operators, *Integrated Computer-Aided Engineering* **23**(3) (2016), 269–285.
[12] Jia L., Wang Y., and Fan L. Multiobjective bilevel optimization for production-distribution planning problems using hybrid genetic algorithm, *Integrated Computer-Aided Engineering* **21**(1) (2014), 77–90.
[13] Martínez-Ballesteros M., Bacardit J., Troncoso A., and Riquelme J.C. Enhancing the scalability of a genetic algorithm to discover quantitative association rules in large-scale datasets, *Integrated Computer-Aided Engineering* **22**(1) (2015), 21–39.
[14] Pillon P., Pedrino E.C., Roda V.O., and Nicoletti M.C. A hardware oriented ad-hoc computer-based method for binary structuring element decomposition based on genetic algorithms, *Integrated Computer-Aided Engineering* **23**(4) (2016), 369–383.
[15] Qin A.K., Huang V.L., and Suganthan P.N. Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE Transactions on Evolutionary Computation* **13**(2) (2009), 398–417.
[16] Cheng J., Zhang G., Caraffini F., and Neri F. Multicriteria adaptive differential evolution for global numerical optimization, *Integrated Computer-Aided Engineering* **22**(2) (2015), 103–107.
[17] Montana D.J. Strongly typed genetic programming, *Evolutionary Computation* **3**(2) (1995), 199–230.
[18] Yao X., Liu Y., and Lin G. Evolutionary programming made faster, *IEEE Transactions on Evolutionary Computation* **3**(2) (1999), 82–102.
[19] Alexandridis A., Paizis E., Chondrodima E., and Stogiannos M. A particle swarm optimization approach in printed circuit board thermal design, *Integrated Computer-Aided Engineering* **24**(2) (2017), 143–155.
[20] Boulkaibet I., Mthembu L., De Lima Neto F., and Marwala T. Finite element model updating using fish school search and volitive particle swarm optimization, *Integrated Computer-Aided Engineering* **22**(4) (2015), 361–376.
[21] Dorigo M., and Di Caro G. Ant colony optimization: a new meta-heuristic, in: *Proceedings of the 1999 Congress on Evolutionary Computation*, IEEE, 1999, pp. 1470–1477.
[22] Karaboga D., and Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *Journal of Global Optimization* **39**(3) (2007), 459–471.
[23] Mehrabian A.R., and Lucas C. A novel numerical optimization algorithm inspired from weed colonization, *Ecological Informatics* **1**(4) (2006), 355–366.
[24] Wang G., Deb S., Gandomi A.H., Zhang Z., and Alavi A.H. Chaotic cuckoo search, *Soft Computing* **20**(9) (2016), 3349–3362.
[25] Pan Q., Sang H., Duan J., and Gao L. An improved fruit fly optimization algorithm for continuous function optimization problems, *Knowledge-Based Systems* **62** (2014), 69–83.
[26] Siddique N., and Adeli H. Harmony search algorithm and its variants, *International Journal of Pattern Recognition and Ar-*

tificial Intelligence **29**(8) (2015), 1539001.

[27] Yang X. A New Metaheuristic Bat-Inspired Algorithm, in: *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 65–74.

[28] Iglesias A., Galvez A., and Collantes M. Multilayer embedded bat algorithm for B-spline curve reconstruction, *Integrated Computer-Aided Engineering* **24**(4) (2017), 385–399.

[29] Rashedi E., Nezamabadi-Pour H., and Saryazdi S. GSA: a gravitational search algorithm, *Information Sciences* **179**(13) (2009), 2232–2248.

[30] Kaveh A., and Khayatazad M. A new meta-heuristic method: ray optimization, *Computers & Structures* **112** (2012), 283–294.

[31] Hatamlou A. Black hole: a new heuristic optimization approach for data clustering, *Information Sciences* **222** (2013), 175–184.

[32] Kaveh A., and Talatahari S. A novel heuristic optimization method: charged system search, *Acta Mechanica* **213**(3) (2010), 267–289.

[33] Siddique N., and Adeli H. Spiral dynamics algorithm, *International Journal on Artificial Intelligence Tools* **23**(6) (2014), 1430001.

[34] Siddique N., and Adeli H. Water drop algorithms, *International Journal on Artificial Intelligence Tools* **23**(6) (2014), 1430002.

[35] Alatas B. ACROA: artificial chemical reaction optimization algorithm for global optimization, *Expert Systems with Applications* **38**(10) (2011), 13170–13180.

[36] Wolpert D.H., and Macready W.G. No free lunch theorems for optimization, *IEEE Transactions on Evolutionary Computation* **1**(1) (1997), 67–82.

[37] Caraffini F., Neri F., and Epitropakis M. HyperSPAM: a study on hyper-heuristic coordination strategies in the continuous domain, *Information Sciences* **477** (2019), 186–202.

[38] Liu Y., and Yao X. Ensemble learning via negative correlation, *Neural Networks* **12**(10) (1999), 1399–1404.

[39] Fernández A., Carmona C.J., José del Jesus M., and Herrera F. A pareto-based ensemble with feature and instance selection for learning from multi-class imbalanced datasets, *International Journal of Neural Systems* **27**(6) (2017), 1750028, PMID: 28633551.

[40] Dietterich T.G. Ensemble methods in machine learning, in: *Multiple Classifier Systems*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 1–15.

[41] G D.T. Machine-learning research, *AI Magazine* **18**(4) (1997), 97–106.

[42] Yan X., He F., and Chen Y. A novel hardware/software partitioning method based on position disturbed particle swarm optimization with invasive weed optimization, *Journal of Computer Science and Technology* **32**(2) (2017), 340–355.

[43] Li H., He F., and Yan X. IBEA-SVM: an indicator-based evolutionary algorithm based on pre-selection with classification guided by SVM, *Applied Mathematics-A Journal of Chinese Universities* **34**(1) (2019), 1–26.

[44] Deng L., Lu G., Shao Y., Fei M., and Hu H. A novel camera calibration technique based on differential evolution particle swarm optimization algorithm, *Neurocomputing* **174** (2016), 456–465.

[45] Yan X., He F., Hou N., and Ai H. An efficient particle swarm optimization for large-scale hardware/software co-design system, *International Journal of Cooperative Information Systems* **27**(1) (2018), 1741001.

[46] Neri F., and Cotta C. Memetic algorithms and memetic computing optimization: a literature review, *Swarm and Evolutionary Computation* **2** (2012), 1–14.

[47] Ouyang Y., and Yin H. Multi-step time series forecasting with an ensemble of varied length mixture models, *International Journal of Neural Systems* **28**(4) (2018), 1750053, PMID: 29297261.

[48] Mencía C., Sierra M.R., Mencía R., and Varela R. Evolutionary one-machine scheduling in the context of electric vehicles charging, *Integrated Computer-Aided Engineering* **26**(1) (2019), 49–63.

[49] Caraffini F., Neri F., and Picinali L. An analysis on separability for Memetic Computing automatic design, *Information Sciences* **265** (2014), 1–22.

[50] Mallipeddi R., Mallipeddi S., and Suganthan P.N. Ensemble strategies with adaptive evolutionary programming, *Information Sciences* **180**(9) (2010), 1571–1581.

[51] Wang Y., and Li B. Two-stage based ensemble optimization for large-scale global optimization, in: *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.

[52] Qu B.Y., and Suganthan P.N. Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods, *Engineering Optimization* **43**(4) (2011), 403–416.

[53] Zhao S., Suganthan P.N., and Zhang Q. Decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes, *IEEE Transactions on Evolutionary Computation* **16**(3) (2012), 442–446.

[54] Yu E.L., and Suganthan P.N. Ensemble of niching algorithms, *Information Sciences* **180** (2010), 2815–2833.

[55] Tasgetiren M.F., Suganthan P.N., and Pan Q. An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem, *Applied Mathematics and Computation* **215**(9) (2010), 3356–3368.

[56] Mallipeddi R., and Suganthan P. Differential evolution algorithm with ensemble of populations for global numerical optimization, *Opsearch* **46**(2) (2009), 184–213.

[57] Mallipeddi R., and Suganthan P.N. Ensemble differential evolution algorithm for CEC2011 problems, in: *2011 IEEE Congress of Evolutionary Computation (CEC)*, 2011, pp. 1557–1564.

[58] Zhang G., Rong H., Neri F., and Pérez-Jiménez M.J. An optimization spiking neural p system for approximately solving combinatorial optimization problems, *International Journal of Neural Systems* **24**(5) (2014), 1440006, PMID: 24875789.

[59] Iacca G., Caraffini F., and Neri F. Multi-strategy coevolving aging particle optimization, *International Journal of Neural Systems* **24**(1) (2014), 1450008, PMID: 24344695.

[60] Sagi O., and Rokach L. Ensemble learning: a survey, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**(4) (2018), 1249.

[61] Freund Y., and Schapire R.E. A decision-theoretic generalization of on-line learning and an application to boosting, *Journal of Computer and System Sciences* **55**(1) (1997), 119–139.

[62] Bauer E., and Kohavi R. An empirical comparison of voting classification algorithms: bagging, boosting, and variants, *Machine Learning* **36**(1) (1999), 105–139.

[63] Breiman L. Bagging predictors, *Machine Learning* **24**(2) (1996), 123–140.

[64] Wang G., Sun J., Ma J., Xu K., and Gu J. Sentiment classification: the contribution of ensemble learning, *Decision Support Systems* **57** (2014), 77–93.

[65] Zhou Z., Wu J., and Tang W. Ensembling neural networks: many could be better than all, *Artificial Intelligence* **137** (2002), 239–263.

[66] Gupta A., Ong Y., and Feng L. Multifactorial evolution: toward evolutionary multitasking, *IEEE Transactions on Evolutionary Computation* **20**(3) (2016), 343–357.

[67] Liang J., Qu B., Suganthan P., and Hernández-Díaz A.G. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report* **201212** (2013).

[68] Li J., Hu Q., Ai M., and Zhong R. Robust feature matching via support-line voting and affine-invariant ratios, *ISPRS Journal of Photogrammetry and Remote Sensing* **132** (2017), 61–76.

[69] Li Q., and Ji H. Multimodality image registration using local linear embedding and hybrid entropy, *Neurocomputing* **111** (2013), 34–42.

[70] Lu W., Chen M., Olivera G.H., Ruchala K.J., and Mackie T.R. Fast free-form deformable registration via calculus of variations, *Physics in Medicine and Biology* **49**(14) (2004), 3067–3087.

[71] Wachowiak M.P., Smolikova R., Zheng Y., Zurada J.M., and Elmaghraby A.S. An approach to multimodal biomedical image registration utilizing particle swarm optimization, *IEEE Transactions on Evolutionary Computation* **8**(3) (2004), 289–301.

[72] Ye Y., Shan J., Bruzzone L., and Shen L. Robust registration of multimodal remote sensing images based on structural similarity, *IEEE Transactions on Geoscience and Remote Sensing* **55**(5) (2017), 2941–2958.

[73] Zhang D., He F., Han S., Zou L., Wu Y., and Chen Y. An efficient approach to directly compute the exact Hausdorff distance for 3D point sets, *Integrated Computer Aided Engineering* **24**(3) (2017), 261–277.

[74] Zhang D., He F., Han S., and Li X. Quantitative optimization of interoperability during feature-based data exchange, *Integrated Computer-Aided Engineering* **23**(1) (2016), 31–51.

[75] Wu Y., He F., Zhang D., and Li X. Service-oriented feature-based data exchange for cloud-based design and manufacturing, *IEEE Transactions on Services Computing* **11**(2) (2018), 341–353.

[76] Lv X., He F., Cai W., and Cheng Y. Supporting selective undo of string-wise operations for collaborative editing systems, *Future Generation Computer Systems* **82** (2018), 41–62.

[77] Lv X., He F., Cheng Y., and Yiqi W. A novel CRDT-based synchronization method for real-time collaborative CAD Systems, *Advanced Engineering Informatics* **38** (2018), 381–391.

[78] Yu H., He F., and Pan Y. A novel region-based active contour model via local patch similarity measure for image segmentation, *Multimedia Tools and Applications* **77**(18) (2018), 24097–24119.

[79] Lv X., He F., Cai W., and Cheng Y. An optimized RGA supporting selective undo for collaborative text editing systems, *Journal of Parallel and Distributed Computing* **132** (2019), 310–330.

[80] Antoniades A., Spyrou L., Martin-Lopez D., Valentin A., Alarcon G., Sanei S., and Took C.C. Deep neural architectures for mapping scalp to intracranial EEG, *International Journal of Neural Systems* **28**(8) (2018), 1850009, PMID: 29631503.

[81] Rafiei M.H., and Adeli H. A novel unsupervised deep learning model for global and local health condition assessment of structures, *Engineering Structures* **156** (2018), 598–607.

[82] Pan Y., He F., and Yu H. A novel enhanced collaborative autoencoder with knowledge distillation for top-n recommender systems, *Neurocomputing* **332** (2019), 137–148.

[83] Xue Y., and Li Y. A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects, *Computer-Aided Civil and Infrastructure Engineering* **33**(8) (2018), 638–654.

[84] Gao Y., and Mosalam K.M. Deep transfer learning for image-based structural damage recognition, *Computer-Aided Civil and Infrastructure Engineering* **33**(9) (2018), 748–768.

[85] Vera-Olmos F., Pardo E., Melero H., and Malpica N. DeepEye: deep convolutional network for pupil detection in real environments, *Integrated Computer-Aided Engineering* **26**(1) (2019), 85–95.

[86] Rostami S., Neri F., and Epitropakis M. Progressive preference articulation for decision making in multi-objective optimisation problems, *Integrated Computer-Aided Engineering* **24**(4) (2017), 315–335.

[87] Yang Z., Emmerich M., Bäck T., and Kok J. Multi-objective inventory routing with uncertain demand using population-based metaheuristics, *Integrated Computer-Aided Engineering* **23**(3) (2016), 205–220.

[88] Valenzuela O., Jiang X., Carrillo A., and Rojas I. Multi-objective genetic algorithms to find most relevant volumes of the brain related to alzheimer's disease and mild cognitive impairment, *International Journal of Neural Systems* **28**(9) (2018), 1850022, PMID: 29914313.

[89] Su Y., Wu Y., Ji W., and Shen S. Shape generation of grid structures by inverse hanging method coupled with multiobjective optimization, *Computer-Aided Civil and Infrastructure Engineering* **33**(6) (2018), 498–509.

[90] Liu J., Wang Y., Fan N., Wei S., and Tong W. A convergence-diversity balanced fitness evaluation mechanism for decomposition-based many-objective optimization algorithm, *Integrated Computer-Aided Engineering* **26**(2) (2019), 1–26.